Contents

CHAPTER 1

Notation and conventions

A dataset as a collection of d-tuples (a d-tuple is an ordered list of d elements). Tuples differ from vectors, because we can always add and subtract vectors, but we cannot necessarily add or subtract tuples. There are always N items in any dataset. There are always d elements in each tuple in a dataset. The number of elements will be the same for every tuple in any given tuple. Sometimes we may not know the value of some elements in some tuples.

We use the same notation for a tuple and for a vector. Most of our data will be vectors. We write a vector in bold, so $\mathbf{x}$ could represent a vector or a tuple (the context will make it obvious which is intended).

The entire data set is $\{\mathbf{x}\}$. When we need to refer to the i'th data item, we write $\mathbf{x}_i$. Assume we have N data items, and we wish to make a new dataset out of them; we write the dataset made out of these items as $\{\mathbf{x}_i\}$ (the i is to suggest you are taking a set of items and making a dataset out of them). If we need to refer to the j'th component of a vector $\mathbf{x}_i$, we will write $x_i^{(j)}$ (notice this isn't in bold, because it is a component not a vector, and the j is in parentheses because it isn't a power). Vectors are always column vectors.

When I write $\{kx\}$, I mean the dataset created by taking each element of the dataset $\{x\}$ and multiplying by k; and when I write $\{x + c\}$, I mean the dataset created by taking each element of the dataset $\{x\}$ and adding c.

Terms:

- mean $(\{x\})$ is the mean of the dataset $\{x\}$ (definition 2.1, page 19).
- std $(\{x\})$ is the standard deviation of the dataset $\{x\}$ (definition 2.2, page 21).
- var $(\{x\})$ is the standard deviation of the dataset $\{x\}$ (definition 2.3, page 25).
- median $(\{x\})$ is the standard deviation of the dataset $\{x\}$ (definition 2.4, page 27).
- percentile$(\{x\}, k)$ is the $k\%$ percentile of the dataset $\{x\}$ (definition 2.5, page 28).
- iqr$\{x\}$ is the interquartile range of the dataset $\{x\}$ (definition 2.7, page 29).
- $\{\hat{x}\}$ is the dataset $\{x\}$, transformed to standard coordinates (definition 2.8, page 34).
- Standard normal data is defined in definition 2.9, (page 34).
- Normal data is defined in definition 2.10, (page 34).
- corr $(\{(x, y)\})$ is the correlation between two components x and y of a dataset (definition 3.1, page 60).

- $\emptyset$ is the empty set.
- Ω is the set of all possible outcomes of an experiment.
- Sets are written as $\mathcal{A}$.
- $\mathcal{A}^c$ is the complement of the set $\mathcal{A}$ (i.e. $\Omega - \mathcal{A}$).
- $\mathcal{E}$ is an event (page 411).
- $P(\{\mathcal{E}\})$ is the probability of event $\mathcal{E}$ (page 411).
- $P(\{\mathcal{E}\}|\{\mathcal{F}\})$ is the probability of event $\mathcal{E}$, conditioned on event $\mathcal{F}$ (page 411).
- $p(x)$ is the probability that random variable X will take the value x; also written $P(\{X = x\})$ (page 411).
- $p(x, y)$ is the probability that random variable X will take the value x and random variable Y will take the value y; also written $P(\{X = x\} \cap \{Y = y\})$ (page 411).
- $\underset{x}{\text{argmax}}\ f(x)$ means the value of x that maximises $f(x)$.
- $\underset{x}{\text{argmin}}\ f(x)$ means the value of x that minimises $f(x)$.
- $\max_i(f(x_i))$ means the largest value that f takes on the different elements of the dataset $\{x_i\}$.
- $\hat{\theta}$ is an estimated value of a parameter θ.

1.0.1 Background Information

Cards: A standard deck of playing cards contains 52 cards. These cards are divided into four suits. The suits are: spades and clubs (which are black); and hearts and diamonds (which are red). Each suit contains 13 cards: Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack (sometimes called Knave), Queen and King. It is common to call Jack, Queen and King *court cards*.

Dice: If you look hard enough, you can obtain dice with many different numbers of sides (though I've never seen a three sided die). We adopt the convention that the sides of an N sided die are labeled with the numbers $1 \ldots N$, and that no number is used twice. Most dice are like this.

Fairness: Each face of a fair coin or die has the same probability of landing upmost in a flip or roll.

Roulette: A roulette wheel has a collection of slots. There are 36 slots numbered with the digits $1 \ldots 36$, and then one, two or even three slots numbered with zero. There are no other slots. A ball is thrown at the wheel when it is spinning, and it bounces around and eventually falls into a slot. If the wheel is properly balanced, the ball has the same probability of falling into each slot. The number of the slot the ball falls into is said to "come up". There are a variety of bets available.

1.1 ACKNOWLEDGEMENTS

Typos spotted by: Han Chen (numerous!), Henry Lin (numerous!), Eric Huber, Brian Lunt, Yusuf Sobh, Scott Walters, — Your Name Here — Jian Peng and Paris Smaragdis taught courses from versions of these notes, and improved them by detailed comments, suggestions and typo lists. TA's for this course have helped improve the notes. Thanks to Zicheng Liao, Michael Sittig, Nikita Spirin, Saurabh Singh, Daphne Tsatsoulis, Henry Lin, Karthik Ramaswamy.

PART ONE

DESCRIBING DATASETS

CHAPTER 2

First Tools for Looking at Data

The single most important question for a working scientist — perhaps the single most useful question anyone can ask — is: "what's going on here?" Answering this question requires creative use of different ways to make pictures of datasets, to summarize them, and to expose whatever structure might be there. This is an activity that is sometimes known as "Descriptive Statistics". There isn't any fixed recipe for understanding a dataset, but there is a rich variety of tools we can use to get insights.

2.1 DATASETS

A dataset is a collection of descriptions of different instances of the same phenomenon. These descriptions could take a variety of forms, but it is important that they are descriptions of the same thing. For example, my grandfather collected the daily rainfall in his garden for many years; we could collect the height of each person in a room; or the number of children in each family on a block; or whether 10 classmates would prefer to be "rich" or "famous". There could be more than one description recorded for each item. For example, when he recorded the contents of the rain gauge each morning, my grandfather could have recorded (say) the temperature and barometric pressure. As another example, one might record the height, weight, blood pressure and body temperature of every patient visiting a doctor's office.

The descriptions in a dataset can take a variety of forms. A description could be **categorical**, meaning that each data item can take a small set of prescribed values. For example, we might record whether each of 100 passers-by preferred to be "Rich" or "Famous". As another example, we could record whether the passers-by are "Male" or "Female". Categorical data could be **ordinal**, meaning that we can tell whether one data item is larger than another. For example, a dataset giving the number of children in a family for some set of families is categorical, because it uses only non-negative integers, but it is also ordinal, because we can tell whether one family is larger than another.

Some ordinal categorical data appears not to be numerical, but can be assigned a number in a reasonably sensible fashion. For example, many readers will recall being asked by a doctor to rate their pain on a scale of 1 to 10 — a question that is usually relatively easy to answer, but is quite strange when you think about it carefully. As another example, we could ask a set of users to rate the usability of an interface in a range from "very bad" to "very good", and then record that using -2 for "very bad", -1 for "bad", 0 for "neutral", 1 for "good", and 2 for "very good".

Many interesting datasets involve **continuous** variables (like, for example, height or weight or body temperature) when you could reasonably expect to encounter any value in a particular range. For example, we might have the heights of

all people in a particular room; or the rainfall at a particular place for each day of the year; or the number of children in each family on a list.

You should think of a dataset as a collection of d-tuples (a d-tuple is an ordered list of d elements). Tuples differ from vectors, because we can always add and subtract vectors, but we cannot necessarily add or subtract tuples. We will always write N for the number of tuples in the dataset, and d for the number of elements in each tuple. The number of elements will be the same for every tuple, though sometimes we may not know the value of some elements in some tuples (which means we must figure out how to predict their values, which we will do much later).

Index	net worth
1	100, 360
2	109, 770
3	96, 860
4	97, 860
5	108, 930
6	124, 330
7	101, 300
8	112, 710
9	106, 740
10	120, 170

Index	Taste score	Index	Taste score
1	12.3	11	34.9
2	20.9	12	57.2
3	39	13	0.7
4	47.9	14	25.9
5	5.6	15	54.9
6	25.9	16	40.9
7	37.3	17	15.9
8	21.9	18	6.4
9	18.1	19	18
10	21	20	38.9

TABLE 2.1: *On the* **left**, *net worths of people you meet in a bar, in US $; I made this data up, using some information from the US Census. The index column, which tells you which data item is being referred to, is usually not displayed in a table because you can usually assume that the first line is the first item, and so on. On the* **right**, *the taste score (I'm not making this up; higher is better) for 20 different cheeses. This data is real (i.e. not made up), and it comes from http://lib.stat.cmu.edu/DASL/Datafiles/Cheese.html.*

Each element of a tuple has its own type. Some elements might be categorical. For example, one dataset we shall see several times has entries for Gender; Grade; Age; Race; Urban/Rural; School; Goals; Grades; Sports; Looks; and Money for 478 children, so $d = 11$ and $N = 478$. In this dataset, each entry is categorical data. Clearly, these tuples are not vectors because one cannot add or subtract (say) Gender, or add Age to Grades.

Most of our data will be vectors. We use the same notation for a tuple and for a vector. We write a vector in bold, so $\mathbf{x}$ could represent a vector or a tuple (the context will make it obvious which is intended).

The entire data set is $\{\mathbf{x}\}$. When we need to refer to the i'th data item, we write $\mathbf{x}_i$. Assume we have N data items, and we wish to make a new dataset out of them; we write the dataset made out of these items as $\{\mathbf{x}_i\}$ (the i is to suggest you are taking a set of items and making a dataset out of them).

In this chapter, we will work mainly with continuous data. We will see a variety of methods for plotting and summarizing 1-tuples. We can build these plots from a dataset of d-tuples by extracting the r'th element of each d-tuple.

All through the book, we will see many datasets downloaded from various web sources, because people are so generous about publishing interesting datasets on the web. In the next chapter, we will look at 2-dimensional data, and we look at high dimensional data in chapter 10.

2.2 WHAT'S HAPPENING? - PLOTTING DATA

The very simplest way to present or visualize a dataset is to produce a table. Tables can be helpful, but aren't much use for large datasets, because it is difficult to get any sense of what the data means from a table. As a continuous example, table 2.1 gives a table of the net worth of a set of people you might meet in a bar (I made this data up). You can scan the table and have a rough sense of what is going on; net worths are quite close to $ 100, 000, and there aren't any very big or very small numbers. This sort of information might be useful, for example, in choosing a bar.

People would like to measure, record, and reason about an extraordinary variety of phenomena. Apparently, one can score the goodness of the flavor of cheese with a number (bigger is better); table 2.1 gives a score for each of thirty cheeses (I did not make up this data, but downloaded it from http://lib.stat.cmu.edu/DASL/Datafiles/Cheese.html). You should notice that a few cheeses have very high scores, and most have moderate scores. It's difficult to draw more significant conclusions from the table, though.

Gender	Goal	Gender	Goal
boy	Sports	girl	Sports
boy	Popular	girl	Grades
girl	Popular	boy	Popular
girl	Popular	boy	Popular
girl	Popular	boy	Popular
girl	Popular	girl	Grades
girl	Popular	girl	Sports
girl	Grades	girl	Popular
girl	Sports	girl	Grades
girl	Sports	girl	Sports

TABLE 2.2: *Chase and Dunner (?) collected data on what students thought made other students popular. As part of this effort, they collected information on (a) the gender and (b) the goal of students. This table gives the gender ("boy" or "girl") and the goal (to make good grades —"Grades"; to be popular — "Popular"; or to be good at sports — "Sports"). The table gives this information for the first 20 of 478 students; the rest can be found at http://lib.stat.cmu.edu/DASL/Datafiles/PopularKids.html. This data is clearly categorical, and not ordinal.*

Table 2.2 shows a table for a set of categorical data. Psychologists collected data from students in grades 4-6 in three school districts to understand what factors students thought made other students popular. This fascinating data set can be found at http://lib.stat.cmu.edu/DASL/Datafiles/PopularKids.html, and was prepared by Chase and Dunner (**?**). Among other things, for each student they asked whether the student's goal was to make good grades ("Grades", for short); to be

popular ("Popular"); or to be good at sports ("Sports"). They have this information for 478 students, so a table would be very hard to read. Table 2.2 shows the gender and the goal for the first 20 students in this group. It's rather harder to draw any serious conclusion from this data, because the full table would be so big. We need a more effective tool than eyeballing the table.

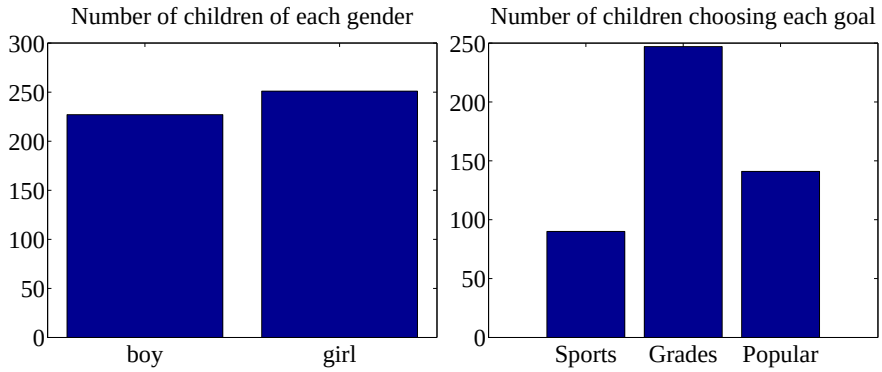

FIGURE 2.1: *On the* **left**, *a bar chart of the number of children of each gender in the Chase and Dunner study (). Notice that there are about the same number of boys and girls (the bars are about the same height). On the* **right**, *a bar chart of the number of children selecting each of three goals. You can tell, at a glance, that different goals are more or less popular by looking at the height of the bars.*

2.2.1 Bar Charts

A **bar chart** is a set of bars, one per category, where the height of each bar is proportional to the number of items in that category. A glance at a bar chart often exposes important structure in data, for example, which categories are common, and which are rare. Bar charts are particularly useful for categorical data. Figure 2.1 shows such bar charts for the genders and the goals in the student dataset of Chase and Dunner (). You can see at a glance that there are about as many boys as girls, and that there are more students who think grades are important than students who think sports or popularity is important. You couldn't draw either conclusion from Table 2.2, because I showed only the first 20 items; but a 478 item table is very difficult to read.

2.2.2 Histograms

Data is continuous when a data item could take any value in some range or set of ranges. In turn, this means that we can reasonably expect a continuous dataset contains few or no pairs of items that have *exactly* the same value. Drawing a bar chart in the obvious way — one bar per value — produces a mess of unit height bars, and seldom leads to a good plot. Instead, we would like to have fewer bars, each representing more data items. We need a procedure to decide which data items count in which bar.

A simple generalization of a bar chart is a **histogram**. We divide the range

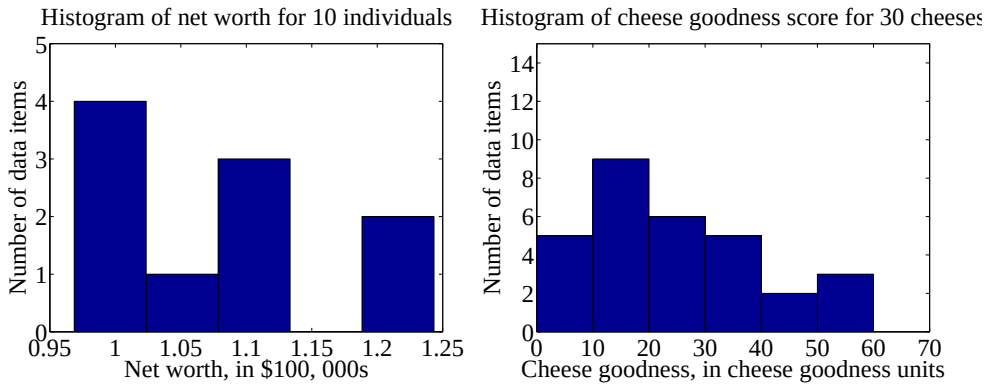

FIGURE 2.2: *On the* **left**, *a histogram of net worths from the dataset described in the text and shown in table 2.1. On the* **right**, *a histogram of cheese goodness scores from the dataset described in the text and shown in table 2.1.*

of the data into intervals, which do not need to be equal in length. We think of each interval as having an associated pigeonhole, and choose one pigeonhole for each data item. We then build a set of boxes, one per interval. Each box sits on its interval on the horizontal axis, and its height is determined by the number of data items in the corresponding pigeonhole. In the simplest histogram, the intervals that form the bases of the boxes are equally sized. In this case, the height of the box is given by the number of data items in the box.

Figure 2.2 shows a histogram of the data in table 2.1. There are five bars — by my choice; I could have plotted ten bars — and the height of each bar gives the number of data items that fall into its interval. For example, there is one net worth in the range between $102, 500 and $107, 500. Notice that one bar is invisible, because there is no data in that range. This picture suggests conclusions consistent with the ones we had from eyeballing the table — the net worths tend to be quite similar, and around $100, 000.

Figure 2.2 also shows a histogram of the data in table 2.1. There are six bars (0-10, 10-20, and so on), and the height of each bar gives the number of data items that fall into its interval — so that, for example, there are 9 cheeses in this dataset whose score is greater than or equal to 10 and less than 20. You can also use the bars to estimate other properties. So, for example, there are 14 cheeses whose score is less than 20, and 3 cheeses with a score of 50 or greater. This picture is much more helpful than the table; you can see at a glance that quite a lot of cheeses have relatively low scores, and few have high scores.

2.2.3 How to Make Histograms

Usually, one makes a histogram by finding the appropriate command or routine in your programming environment. I use Matlab and R, depending on what I feel like.
TODO: Resolve where programming examples live.
It is useful to understand the procedures used to make and plot histograms.

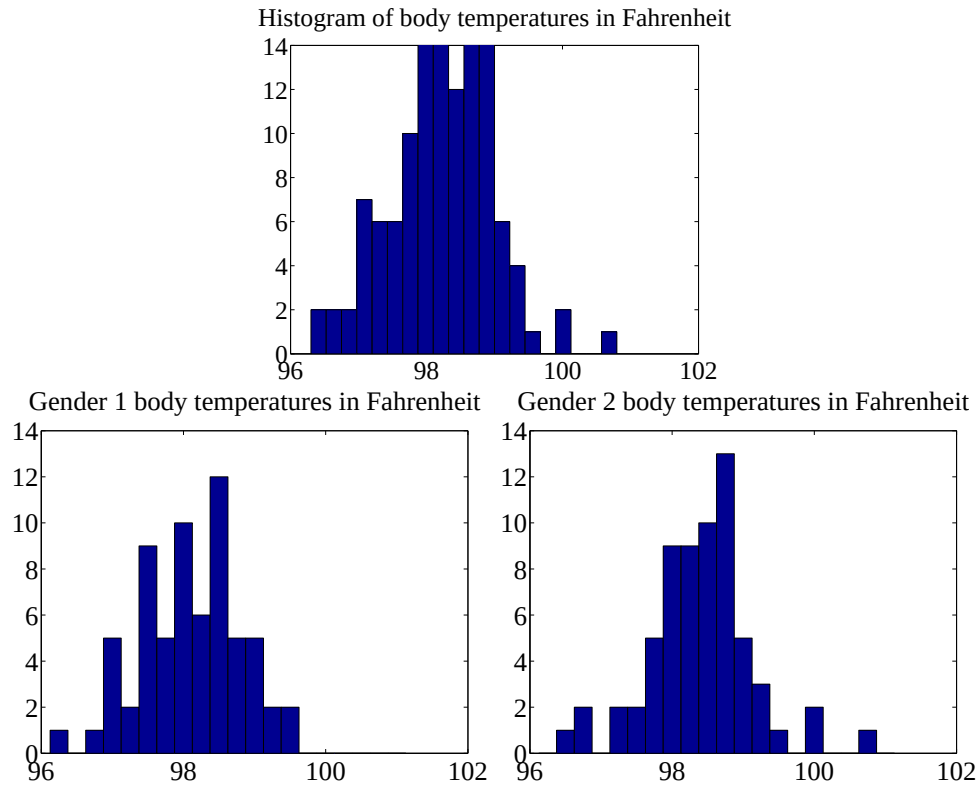

FIGURE 2.3: *On* **top**, *a histogram of body temperatures, from the dataset published at http://www2.stetson.edu/~jrasp/data.htm. These seem to be clustered fairly tightly around one value. The* **bottom row** *shows histograms for each gender (I don't know which is which). It looks as though one gender runs slightly cooler than the other.*

Histograms with Even Intervals: The easiest histogram to build uses equally sized intervals. Write x_i for the i'th number in the dataset, $x_{\min}$ for the smallest value, and $x_{\max}$ for the largest value. We divide the range between the smallest and largest values into n intervals of even width $(x_{\max} - x_{\min})/n$. In this case, the height of each box is given by the number of items in that interval. We could represent the histogram with an n-dimensional vector of counts. Each entry represents the count of the number of data items that lie in that interval. Notice we need to be careful to ensure that each point in the range of values is claimed by exactly one interval. For example, we could have intervals of $[0-1)$ and $[1-2)$, or we could have intervals of $(0-1]$ and $(1-2]$. We could *not* have intervals of $[0-1]$ and $[1-2]$, because then a data item with the value 1 would appear in two boxes. Similarly, we could not have intervals of $(0-1)$ and $(1-2)$, because then a data item with the value 1 would not appear in any box.

Histograms with Uneven Intervals: For a histogram with even intervals, it is natural that the height of each box is the number of data items in that box.

But a histogram with even intervals can have empty boxes (see figure 2.2). In this case, it can be more informative to have some larger intervals to ensure that each interval has some data items in it. But how high should we plot the box? Imagine taking two consecutive intervals in a histogram with even intervals, and fusing them. It is natural that the height of the fused box should be the average height of the two boxes. This observation gives us a rule.

Write dx for the width of the intervals; n_1 for the height of the box over the first interval (which is the number of elements in the first box); and n_2 for the height of the box over the second interval. The height of the fused box will be $(n_1 + n_2)/2$. Now the *area* of the first box is $n_1 dx$; of the second box is $n_2 dx$; and of the fused box is $(n_1 + n_2)dx$. For each of these boxes, the *area* of the box is proportional to the number of elements in the box. This gives the correct rule: plot boxes such that the area of the box is proportional to the number of elements in the box.

2.2.4 Conditional Histograms

Most people believe that normal body temperature is 98.4^o in Fahrenheit. If you take other people's temperatures often (for example, you might have children), you know that some individuals tend to run a little warmer or a little cooler than this number. I found data giving the body temperature of a set of individuals at http://www2.stetson.edu/~jrasp/data.htm. As you can see from the histogram (figure 2.3), the body temperatures cluster around a small set of numbers. But what causes the variation?

One possibility is gender. We can investigate this possibility by comparing a histogram of temperatures for males with histogram of temperatures for females. The dataset gives genders as 1 or 2 - I don't know which is male and which female. Histograms that plot only part of a dataset are sometimes called **conditional histograms** or **class-conditional histograms**, because each histogram is conditioned on something. In this case, each histogram uses only data that comes from a particular gender. Figure 2.3 gives the class conditional histograms. It does seem like individuals of one gender run a little cooler than individuals of the other. Being certain takes considerably more work than looking at these histograms, because the difference might be caused by an unlucky choice of subjects. But the histograms suggests that this work might be worth doing.

2.3 SUMMARIZING 1D DATA

For the rest of this chapter, we will assume that data items take values that are continuous real numbers. Furthermore, we will assume that values can be added, subtracted, and multiplied by constants in a meaningful way. Human heights are one example of such data; you can add two heights, and interpret the result as a height (perhaps one person is standing on the head of the other). You can subtract one height from another, and the result is meaningful. You can multiply a height by a constant — say, $1/2$ — and interpret the result (A is half as high as B).

2.3.1 The Mean

One simple and effective summary of a set of data is its **mean**. This is sometimes known as the **average** of the data.

Definition: 2.1 *Mean*

Assume we have a dataset $\{x\}$ of N data items, $x_1, \ldots, x_N$. Their mean is

$$\text{mean}\left(\{x\}\right) = \frac{1}{N}\sum_{i=1}^{i=N} x_i.$$

For example, assume you're in a bar, in a group of ten people who like to talk about money. They're average people, and their net worth is given in table 2.1 (you can choose who you want to be in this story). The mean of this data is \$107, 903.

Properties of the Mean The mean has several important properties you should remember:

- Scaling data scales the mean: or $\text{mean}\left(\{kx_i\}\right) = k\text{mean}\left(\{x_i\}\right)$.
- Translating data translates the mean: or $\text{mean}\left(\{x_i + c\}\right) = \text{mean}\left(\{x_i\}\right) + c$.
- The sum of signed differences from the mean is zero. This means that

$$\sum_{i=1}^{N}(x_i - \text{mean}\left(\{x_i\}\right)) = 0.$$

- Choose the number μ such that the sum of squared distances of data points to μ is minimized. That number is the mean. In notation

$$\underset{\mu}{\arg\min} \sum_{i}(x_i - \mu)^2 = \text{mean}\left(\{x_i\}\right)$$

These properties are easy to prove (and so easy to remember). I have broken these out into a box of useful facts below, to emphasize them. All but one proof is relegated to the exercises.

Property 2.1: The Average Squared Distance to the Mean is Minimized

Proposition: $\underset{\mu}{arg\ min}\ \sum_i (x_i - \mu)^2 = \mathsf{mean}\,(\{x\})$

Proof: Choose the number μ such that the sum of squared distances of data points to μ is minimized. That number is the mean. In notation:

$$\underset{\mu}{\arg\min}\ \sum_i (x_i - \mu)^2 = \mathsf{mean}\,(\{x\})$$

We can show this by actually minimizing the expression. We must have that the derivative of the expression we are minimizing is zero at the value of μ we are seeking. So we have

$$\begin{aligned} \frac{d}{d\mu}\sum_{i=1}^{N}(x_i - \mu)^2 &= \sum_{i=1}^{N} 2(x_i - \mu) \\ &= 2\sum_{i=1}^{N}(x_i - \mu) \\ &= 0 \end{aligned}$$

so that $2N\mathsf{mean}\,(\{x\}) - 2N\mu = 0$, which means that $\mu = \mathsf{mean}\,(\{x\})$.

Now this result means that the mean is the single number that is closest to all the data items. The mean tells you where the overall blob of data lies. For this reason, it is often referred to as a **location parameter**. If you choose to summarize the dataset with a number that is as close as possible to each data item, the mean is the number to choose. The mean is also a guide to what new values will look like, if you have no other information. For example, in the case of the bar, a new person walks in, and I must guess that person's net worth. Then the mean is the best guess, because it is closest to all the data items we have already seen. In the case of the bar, if a new person walked into this bar, and you had to guess that person's net worth, you should choose \$107, 903.

Useful Facts: 2.1 *Properties of the mean*

- $\mathsf{mean}\left(\{kx_i\}\right) = k\mathsf{mean}\left(\{x_i\}\right)$.
- $\mathsf{mean}\left(\{x_i + c\}\right) = \mathsf{mean}\left(\{x_i\}\right) + c$.
- $[\sum_{i=1}^{N}(x_i - \mathsf{mean}\left(\{x_i\}\right)) = 0$.
- $\underset{\mu}{\arg\min} \sum_i (x_i - \mu)^2 = \mathsf{mean}\left(\{x_i\}\right)$

The mean is a location parameter; it tells you where the data lies along a number line.

2.3.2 Standard Deviation and Variance

We would also like to know the extent to which data items are close to the mean. This information is given by the **standard deviation**, which is the root mean square of the offsets of data from the mean.

Definition: 2.2 *Standard deviation*

Assume we have a dataset $\{x\}$ of N data items, $x_1, \ldots, x_N$. The standard deviation of this dataset is is:

$$\mathsf{std}\left(\{x_i\}\right) = \sqrt{\frac{1}{N}\sum_{i=1}^{i=N}(x_i - \mathsf{mean}\left(\{x\}\right))^2} = \sqrt{\mathsf{mean}\left(\{(x_i - \mathsf{mean}\left(\{x\}\right))^2\}\right)}.$$

You should think of the standard deviation as a scale. It measures the size of the average deviation from the mean for a dataset, or how wide the spread of data is. For this reason, it is often referred to as a **scale parameter**. When the standard deviation of a dataset is large, there are many items with values much larger than, or much smaller than, the mean. When the standard deviation is small, most data items have values close to the mean. This means it is helpful to talk about how many standard devations away from the mean a particular data item is. Saying that data item x_j is "within k standard deviations from the mean" means that

$$\mathsf{abs}\left(x_j - \mathsf{mean}\left(\{x\}\right)\right) \leq k\mathsf{std}\left(\{x_i\}\right).$$

Similarly, saying that data item x_j is "more than k standard deviations from the mean" means that

$$\mathsf{abs}\left(x_i - \mathsf{mean}\left(\{x\}\right)\right) > k\mathsf{std}\left(\{x\}\right).$$

As I will show below, there must be some data at least one standard deviation away from the mean, and there can be very few data items that are many standard deviations away from the mean.

Properties of the Standard Deviation
Standard deviation has very important properties:

- Translating data does not change the standard deviation, i.e. $\mathsf{std}\left(\{x_i + c\}\right) = \mathsf{std}\left(\{x_i\}\right)$.
- Scaling data scales the standard deviation, i.e. $\mathsf{std}\left(\{kx_i\}\right) = k\mathsf{std}\left(\{x_i\}\right)$.
- For any dataset, there can be only a few items that are many standard deviations away from the mean. In particular, assume we have N data items, x_i, whose standard deviation is σ. Then there are at most $\frac{1}{k^2}$ data points lying k or more standard deviations away from the mean.
- For any dataset, there must be at least one data item that is at least one standard deviation away from the mean.

The first two properties are easy to prove, and are relegated to the exercises. I prove the others below. Again, for emphasis, I have broken these properties out in a box below.

Property 2.2: For any dataset, it is hard for data items to get many standard deviations away from the mean.

Proposition: *Assume we have a dataset $\{x\}$ of N data items, $x_1, \ldots, x_N$. Assume the standard deviation of this dataset is* $\mathsf{std}(\{x\}) = \sigma$. *Then there are at most $\frac{1}{k^2}$ data points lying k or more standard deviations away from the mean.*

Proof: Assume the mean is zero. There is no loss of generality here, because translating data translates the mean, but doesn't change the standard deviation. Now we must construct a dataset with the largest possible fraction r of data points lying k or more standard deviations from the mean. To achieve this, our data should have $N(1-r)$ data points each with the value 0, because these contribute 0 to the standard deviation. It should have Nr data points with the value $k\sigma$; if they are further from zero than this, each will contribute more to the standard deviation, so the fraction of such points will be fewer. Because

$$\mathsf{std}(\{x\}) = \sigma = \sqrt{\frac{\sum_i x_i^2}{N}}$$

we have that, for this rather specially constructed dataset,

$$\sigma = \sqrt{\frac{Nrk^2\sigma^2}{N}}$$

so that

$$r = \frac{1}{k^2}.$$

We constructed the dataset so that r would be as large as possible, so

$$r \leq \frac{1}{k^2}$$

for any kind of data at all.

This bound (box 2.2) is true for *any kind of data*. This bound implies that, for example, at most 100% of *any* dataset could be one standard deviation away from the mean, 25% of *any* dataset is 2 standard deviations away from the mean and at most 11% of *any* dataset could be 3 standard deviations away from the mean. But the configuration of data that achieves this bound is very unusual. This means the bound tends to wildly *overstate* how much data is far from the mean for most practical datasets. Most data has more random structure, meaning that we expect to see very much *less* data far from the mean than the bound predicts. For example, much data can reasonably be modelled as coming from a normal distribution (a topic we'll go into later). For such data, we expect that about 68% of the data is within one standard deviation of the mean, 95% is within two standard deviations of the mean, and 99.7% is within three standard deviations of the mean, and the percentage of data that is within ten standard deviations of the mean is essentially indistinguishable from 100%. This kind of behavior is quite

common; the crucial point about the standard deviation is that you won't see much data that lies many standard deviations from the mean, because you can't.

Property 2.3: For any dataset, there must be at least one data item that is at least one standard deviation away from the mean.

Proposition:

$$(\mathsf{std}\,(\{x\}))^2 \leq \max_i (x_i - \mathsf{mean}\,(\{x\}))^2.$$

Proof: You can see this by looking at the expression for standard deviation. We have

$$\mathsf{std}\,(\{x\}) = \sqrt{\frac{1}{N}\sum_{i=1}^{i=N}(x_i - \mathsf{mean}\,(\{x\}))^2}.$$

Now, this means that

$$N(\mathsf{std}\,(\{x\}))^2 = \sum_{i=1}^{i=N}(x_i - \mathsf{mean}\,(\{x\}))^2.$$

But

$$\sum_{i=1}^{i=N}(x_i - \mathsf{mean}\,(\{x\}))^2 \leq N \max_i (x_i - \mathsf{mean}\,(\{x\}))^2$$

so

$$(\mathsf{std}\,(\{x\}))^2 \leq \max_i (x_i - \mathsf{mean}\,(\{x\}))^2.$$

Boxes 2.2 and 2.3 mean that the standard deviation is quite informative. Very little data is many standard deviations away from the mean; similarly, at least some of the data should be one or more standard deviations away from the mean. So the standard deviation tells us how data points are scattered about the mean.

Useful Facts: 2.2 *Properties of standard deviation*

- $\mathsf{std}\left(\{x + c\}\right) = \mathsf{std}\left(\{x\}\right)$.
- $\mathsf{std}\left(\{kx\}\right) = k\mathsf{std}\left(\{x\}\right)$.
- For N data items, x_i, whose standard deviation is σ, there are at most $\frac{1}{k^2}$ data points lying k or more standard deviations away from the mean.
- $(\mathsf{std}\left(\{x\}\right))^2 \leq \max_i (x_i - \mathsf{mean}\left(\{x\}\right))^2$.

The standard deviation is often referred to as a scale parameter; it tells you how broadly the data spreads about the mean.

There is an ambiguity that comes up often here because two (very slightly) different numbers are called the standard deviation of a dataset. One — the one we use in this chapter — is an estimate of the scale of the data, as we describe it. The other differs from our expression very slightly; one computes

$$\sqrt{\frac{\sum_i (x_i - \mathsf{mean}\left(\{x\}\right))^2}{N - 1}}$$

(notice the $N - 1$ for our N). If N is large, this number is basically the same as the number we compute, but for smaller N there is a difference that can be significant. Irritatingly, this number is also called the standard deviation; even more irritatingly, we will have to deal with it, but not yet. I mention it now because you may look up terms I have used, find this definition, and wonder whether I know what I'm talking about. In this case, I do (although I would say that).

The confusion arises because sometimes the datasets we see are actually samples of larger datasets. For example, in some circumstances you could think of the net worth dataset as a sample of all the net worths in the USA. In such cases, we are often interested in the standard deviation *of the underlying dataset that was sampled* (rather than of the dataset of samples that you have). The second number is a slightly better way to estimate this standard deviation than the definition we have been working with. Don't worry - the N in our expressions is the right thing to use for what we're doing.

2.3.3 Variance

It turns out that thinking in terms of the square of the standard deviation, which is known as the **variance**, will allow us to generalize our summaries to apply to higher dimensional data.

Definition: 2.3 *Variance*

Assume we have a dataset $\{x\}$ of N data items, $x_1, \ldots, x_N$. where $N > 1$. Their variance is:

$$\mathsf{var}\left(\{x\}\right) = \frac{1}{N}\left(\sum_{i=1}^{i=N}(x_i - \mathsf{mean}\left(\{x\}\right))^2\right) = \mathsf{mean}\left(\left\{(x_i - \mathsf{mean}\left(\{x\}\right))^2\right\}\right).$$

One good way to think of the variance is as the mean-square error you would incur if you replaced each data item with the mean. Another is that it is the square of the standard deviation.

Properties of the Variance The properties of the variance follow from the fact that it is the square of the standard deviation. I have broken these out in a box, for emphasis.

Useful Facts: 2.3 *Properties of variance*

- $\mathsf{var}\left(\{x + c\}\right) = \mathsf{var}\left(\{x\}\right)$.
- $\mathsf{var}\left(\{kx\}\right) = k^2\mathsf{var}\left(\{x\}\right)$.

While one could restate the other two properties of the standard deviation in terms of the variance, it isn't really natural to do so. The standard deviation is in the same units as the original data, and should be thought of as a scale. Because the variance is the square of the standard deviation, it isn't a natural scale (unless you take its square root!).

2.3.4 The Median

One problem with the mean is that it can be affected strongly by extreme values. Go back to the bar example, of section 2.3.1. Now Warren Buffett (or Bill Gates, or your favorite billionaire) walks in. What happened to the average net worth?

Assume your billionaire has net worth \$ 1, 000, 000, 000. Then the mean net worth suddenly has become

$$\frac{10 \times \$107,903 + \$1,000,000,000}{11} = \$91,007,184$$

But this mean isn't a very helpful summary of the people in the bar. It is probably more useful to think of the net worth data as ten people together with one billionaire. The billionaire is known as an **outlier**.

One way to get outliers is that a small number of data items are very different, due to minor effects you don't want to model. Another is that the data was misrecorded, or mistranscribed. Another possibility is that there is just too much variation in the data to summarize it well. For example, a small number of extremely wealthy people could change the average net worth of US residents dramatically, as the example shows. An alternative to using a mean is to use a **median**.

Definition: 2.4 *Median*

The median of a set of data points is obtained by sorting the data points, and finding the point halfway along the list. If the list is of even length, it's usual to average the two numbers on either side of the middle. We write

$$\mathsf{median}\left(\{x_i\}\right)$$

for the operator that returns the median.

For example,

$$\mathsf{median}\left(\{3,5,7\}\right) = 5,$$

$$\mathsf{median}\left(\{3,4,5,6,7\}\right) = 5,$$

and

$$\mathsf{median}\left(\{3,4,5,6\}\right) = 4.5.$$

For much, but not all, data, you can expect that roughly half the data is smaller than the median, and roughly half is larger than the median. Sometimes this property fails. For example,

$$\mathsf{median}\left(\{1,2,2,2,2,2,2,2,3\}\right) = 2.$$

With this definition, the median of our list of net worths is $\$107,835$. If we insert the billionaire, the median becomes $\$108,930$. Notice by how little the number has changed — it remains an effective summary of the data.

Properties of the median You can think of the median of a dataset as giving the "middle" or "center" value. It is another way of estimating where the dataset lies on a number line (and so is another location parameter). This means it is rather like the mean, which also gives a (slightly differently defined) "middle" or "center" value. The mean has the important properties that if you translate the dataset, the mean translates, and if you scale the dataset, the mean scales. The median has these properties, too, which I have broken out in a box. Each is easily proved, and proofs are relegated to the exercises.

Useful Facts: 2.4 *Properties of the median*

- $\mathsf{median}\,(\{x + c\}) = \mathsf{median}\,(\{x\}) + c$.
- $\mathsf{median}\,(\{kx\}) = k\mathsf{median}\,(\{x\})$.

2.3.5 Interquartile Range

Outliers are a nuisance in all sorts of ways. Plotting the histogram of the net worth data with the billionaire included will be tricky. Either you leave the billionaire out of the plot, or all the histogram bars are tiny. Visualizing this plot shows outliers can affect standard deviations severely, too. For our net worth data, the standard deviation without the billionaire is \$9265, but if we put the billionaire in there, it is $\$3.014 \times 10^8$. When the billionaire is in the dataset, the mean is about $91M\$$ and the standard deviation is about $300M\$$ — so all but one of the data items lie about a third of a standard deviation away from the mean on the small side. The other data item (the billionaire) is about three standard deviations away from the mean on the large side. In this case, the standard deviation has done its work of informing us that there are huge changes in the data, but isn't really helpful as a description of the data.

The problem is this: describing the net worth data with billionaire as a having a mean of $\$9.101 \times 10^7$ with a standard deviation of $\$3.014 \times 10^8$ isn't really helpful. Instead, the data really should be seen as a clump of values that are near $\$100,000$ and moderately close to one another, and one massive number (the billionaire outlier).

One thing we could do is simply remove the billionaire and compute mean and standard deviation. This isn't always easy to do, because it's often less obvious which points are outliers. An alternative is to follow the strategy we did when we used the median. Find a summary that describes scale, but is less affected by outliers than the standard deviation. This is the **interquartile range**; to define it, we need to define percentiles and quartiles, which are useful anyway.

Definition: 2.5 *Percentile*

The k'th percentile is the value such that $k\%$ of the data is less than or equal to that value. We write $\mathsf{percentile}(\{x\}, k)$ for the k'th percentile of dataset $\{x\}$.

Definition: 2.6 *Quartiles*

The first quartile of the data is the value such that 25% of the data is less than or equal to that value (i.e. $\mathsf{percentile}(\{x\}, 25)$). The second quartile of the data is the value such that 50% of the data is less than or equal to that value, which is usually the median (i.e. $\mathsf{percentile}(\{x\}, 50)$). The third quartile of the data is the value such that 75% of the data is less than or equal to that value (i.e. $\mathsf{percentile}(\{x\}, 75)$).

Definition: 2.7 *Interquartile Range*

The interquartile range of a dataset $\{x\}$ is $\mathsf{iqr}\{x\} = \mathsf{percentile}(\{x\}, 75) - \mathsf{percentile}(\{x\}, 25)$.

Like the standard deviation, the interquartile range gives an estimate of how widely the data is spread out. But it is quite well-behaved in the presence of outliers. For our net worth data without the billionaire, the interquartile range is \$12350; with the billionaire, it is \$17710.

Properties of the interquartile range You can think of the interquartile range of a dataset as giving an estimate of the scale of the difference from the mean. This means it is rather like the standard deviation, which also gives a (slightly differently defined) scale. The standard deviation has the important properties that if you translate the dataset, the standard deviation translates, and if you scale the dataset, the standard deviation scales. The interquartile range has these properties, too, which I have broken out into a box. Each is easily proved, and proofs are relegated to the exercises.

Useful Facts: 2.5 *Properties of the interquartile range*

- $\mathsf{iqr}\{x + c\} = \mathsf{iqr}\{x\}$.
- $\mathsf{iqr}\{kx\} = k^2\mathsf{iqr}\{x\}$.

For most datasets, interquartile ranges tend to be somewhat larger than standard deviations. This isn't really a problem. Each is a method for estimating the scale of the data — the range of values above and below the mean that you are likely to see. It is neither here nor there if one method yields slightly larger estimates than another, as long as you don't compare estimates across methods.

2.3.6 Using Summaries Sensibly

One should be careful how one summarizes data. For example, the statement that "the average US family has 2.6 children" invites mockery (the example is from Andrew Vickers' book *What is a p-value anyway?*), because you can't have fractions of a child — no family has 2.6 children. A more accurate way to say things might be "the average of the number of children in a US family is 2.6", but this is clumsy. What is going wrong here is the 2.6 is a mean, but the number of children in a family is a categorical variable. Reporting the mean of a categorical variable is often a bad idea, because you may never encounter this value (the 2.6 children). For a categorical variable, giving the median value and perhaps the interquartile range often makes much more sense than reporting the mean.

For continuous variables, reporting the mean is reasonable because you could expect to encounter a data item with this value, even if you haven't seen one in the particular data set you have. It is sensible to look at both mean and median; if they're significantly different, then there is probably something going on that is worth understanding. You'd want to plot the data using the methods of the next section before you decided what to report.

You should also be careful about how precisely numbers are reported (equivalently, the number of significant figures). Numerical and statistical software will produce very large numbers of digits freely, but not all are always useful. This is a particular nuisance in the case of the mean, because you might add many numbers, then divide by a large number; in this case, you will get many digits, but some might not be meaningful. For example, Vickers (ibid) describes a paper reporting the mean length of pregnancy as 32.833 weeks. That fifth digit suggests we know the mean length of pregnancy to about 0.001 weeks, or roughly 10 minutes. Neither medical interviewing nor people's memory for past events is that detailed. Furthermore, when you interview them about embarrassing topics, people quite often lie. There is no prospect of knowing this number with this precision.

People regularly report silly numbers of digits because it is easy to miss the harm caused by doing so. But the harm is there: you are implying to other people, and to yourself, that you know something more accurately than you do. At some point, someone may suffer for it.

2.4 PLOTS AND SUMMARIES

Knowing the mean, standard deviation, median and interquartile range of a dataset gives us some information about what its histogram might look like. In fact, the summaries give us a language in which to describe a variety of characteristic properties of histograms that are worth knowing about (Section 2.4.1). Quite remarkably, many different datasets have histograms that have about the same shape (Section 2.4.2). For such data, we know roughly what percentage of data items are how far from the mean.

Complex datasets can be difficult to interpret with histograms alone, because it is hard to compare many histograms by eye. Section 2.4.3 describes a clever plot of various summaries of datasets that makes it easier to compare many cases.

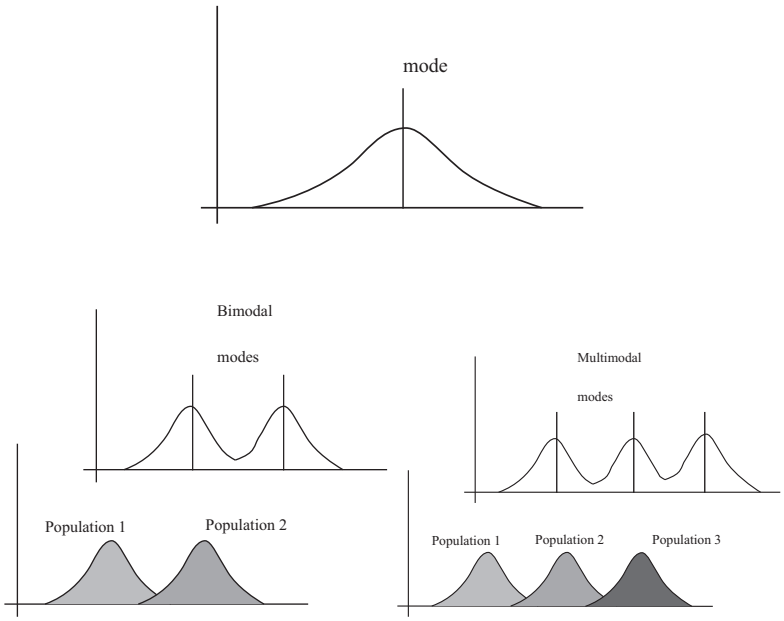

FIGURE 2.4: *Many histograms are unimodal, like the example on the* **top***; there is one peak, or mode. Some are bimodal (two peaks;* **bottom left***) or even multimodal (two or more peaks;* **bottom right***). One common reason (but not the only reason) is that there are actually two populations being conflated in the histograms. For example, measuring adult heights might result in a bimodal histogram, if male and female heights were slightly different. As another example, measuring the weight of dogs might result in a multimodal histogram if you did not distinguish between breeds (eg chihauhau, terrier, german shepherd, pyranean mountain dog, etc.).*

2.4.1 Some Properties of Histograms

The **tails** of a histogram are the relatively uncommon values that are significantly larger (resp. smaller) than the value at the peak (which is sometimes called the **mode**). A histogram is **unimodal** if there is only one peak; if there are more than one, it is **multimodal**, with the special term **bimodal** sometimes being used for the case where there are two peaks (Figure 2.4). The histograms we have seen have been relatively symmetric, where the left and right tails are about as long as one another. Another way to think about this is that values a lot larger than the mean are about as common as values a lot smaller than the mean. Not all data is symmetric. In some datasets, one or another tail is longer (figure 2.5). This effect is called **skew**.

Skew appears often in real data. SOCR (the Statistics Online Computational Resource) publishes a number of datasets. Here we discuss a dataset of citations to faculty publications. For each of five UCLA faculty members, SOCR collected the number of times each of the papers they had authored had been cited by other authors (data at http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_

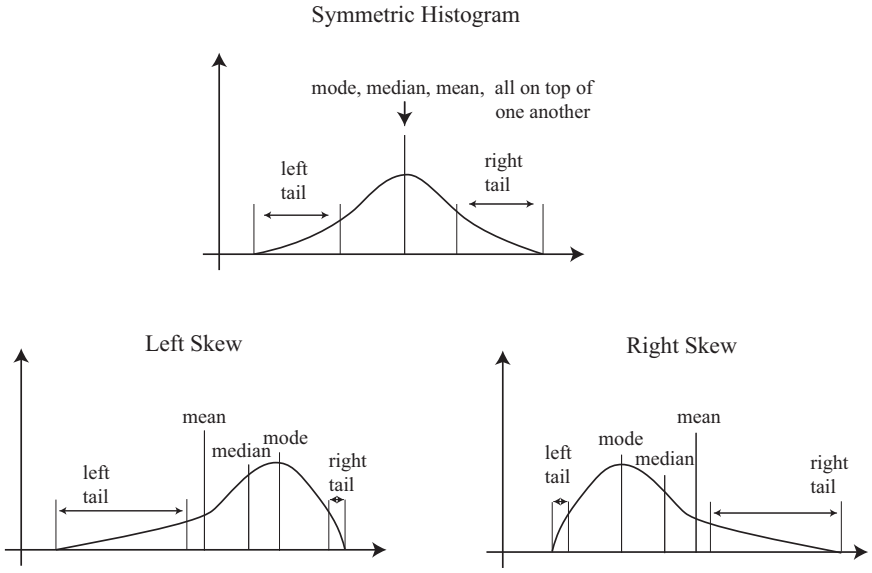

FIGURE 2.5: *On the* **top**, *an example of a symmetric histogram, showing its tails (relatively uncommon values that are significantly larger or smaller than the peak or mode).* **Lower left**, *a sketch of a left-skewed histogram. Here there are few large values, but some very small values that occur with significant frequency. We say the left tail is "long", and that the histogram is left skewed. You may find this confusing, because the main bump is to the right — one way to remember this is that the left tail has been stretched.* **Lower right**, *a sketch of a right-skewed histogram. Here there are few small values, but some very large values that occur with significant frequency. We say the right tail is "long", and that the histogram is right skewed.*

072108_H_Index_Pubs). Generally, a small number of papers get many citations, and many papers get few citations. We see this pattern in the histograms of citation numbers (figure 2.6). These are very different from (say) the body temperature pictures. In the citation histograms, there are many data items that have very few citations, and few that have many citations. This means that the right tail of the histogram is longer, so the histogram is skewed to the right.

One way to check for skewness is to look at the histogram; another is to compare mean and median (though this is not foolproof). For the first citation histogram, the mean is 24.7 and the median is 7.5; for the second, the mean is 24.4, and the median is 11. In each case, the mean is a lot bigger than the median. Recall the definition of the median (form a ranked list of the data points, and find the point halfway along the list). For much data, the result is larger than about half of the data set and smaller than about half the dataset. So if the median is quite small compared to the mean, then there are many small data items and a small number of data items that are large — the right tail is longer, so the histogram is skewed to the right.

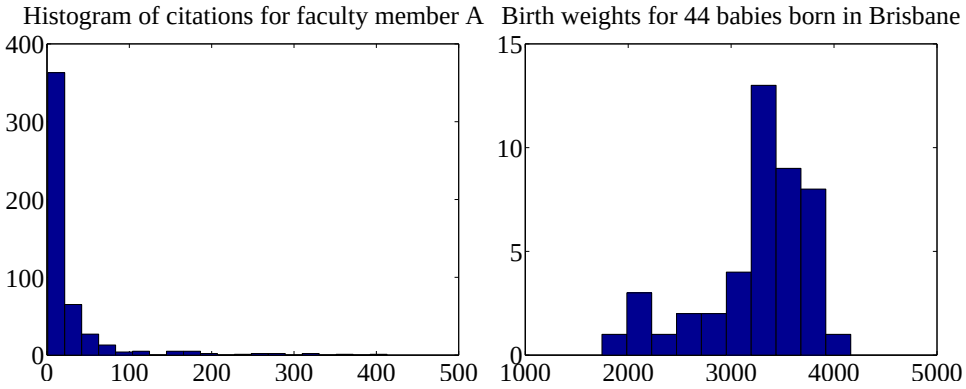

FIGURE 2.6: *On the* **left**, *a histogram of citations for a faculty member, from data at http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_072108_H_Index_Pubs. Very few publications have many citations, and many publications have few. This means the histogram is strongly right-skewed. On the* **right**, *a histogram of birth weights for 44 babies borne in Brisbane in 1997. This histogram looks slightly left-skewed.*

Left-skewed data also occurs; figure 2.6 shows a histogram of the birth weights of 44 babies born in Brisbane, in 1997 (from http://www.amstat.org/publications/jse/jse_data_archive.htm). This data appears to be somewhat left-skewed, as birth weights can be a lot smaller than the mean, but tend not to be much larger than the mean.

Skewed data is often, but not always, the result of constraints. For example, good obstetrical practice tries to ensure that very large birth weights are rare (birth is typically induced before the baby gets too heavy), but it may be quite hard to avoid some small birth weights. This could could skew birth weights to the left (because large babies will get born, but will not be as heavy as they could be if obstetricians had not interfered). Similarly, income data can be skewed to the right by the fact that income is always positive. Test mark data is often skewed — whether to right or left depends on the circumstances — by the fact that there is a largest possible mark and a smallest possible mark.

2.4.2 Standard Coordinates and Normal Data

It is useful to look at lots of histograms, because it is often possible to get some useful insights about data. However, in their current form, histograms are hard to compare. This is because each is in a different set of units. A histogram for length data will consist of boxes whose horizontal units are, say, metres; a histogram for mass data will consist of boxes whose horizontal units are in, say, kilograms. Furthermore, these histograms typically span different ranges.

We can make histograms comparable by (a) estimating the "location" of the plot on the horizontal axis and (b) estimating the "scale" of the plot. The location is given by the mean, and the scale by the standard deviation. We could then normalize the data by subtracting the location (mean) and dividing by the standard

deviation (scale). The resulting values are unitless, and have zero mean. They are often known as **standard coordinates**.

Definition: 2.8 *Standard coordinates*

Assume we have a dataset $\{x\}$ of N data items, $x_1, \ldots, x_N$. We represent these data items in standard coordinates by computing

$$\hat{x}_i = \frac{(x_i - \mathsf{mean}\,(\{x\}))}{\mathsf{std}\,(\{x\})}.$$

We write $\{\hat{x}\}$ for a dataset that happens to be in standard coordinates.

Standard coordinates have some important properties. Assume we have N data items. Write x_i for the i'th data item, and $\hat{x}_i$ for the i'th data item in standard coordinates (I sometimes refer to these as "normalized data items"). Then we have

$$\mathsf{mean}\,(\{\hat{x}\}) = 0.$$

We also have that

$$\mathsf{std}\,(\{\hat{x}\}) = 1.$$

An extremely important fact about data is that, for many kinds of data, histograms of these standard coordinates look the same. Many completely different datasets produce a histogram that, in standard coordinates, has a very specific appearance. It is symmetric and unimodal, and it looks like a bump. If there were enough data points and the histogram boxes were small enough, the histogram would look like the curve in figure 2.7. This phenomenon is so important that data of this form has a special name.

Definition: 2.9 *Standard normal data*

Data is **standard normal data** if, when we have a great deal of data, the histogram of the data in standard coordinates is a close approximation to the **standard normal curve**. This curve is given by

$$y(x) = \frac{1}{\sqrt{2\pi}} e^{\left(-x^2/2\right)}$$

(which is shown in figure 2.7).

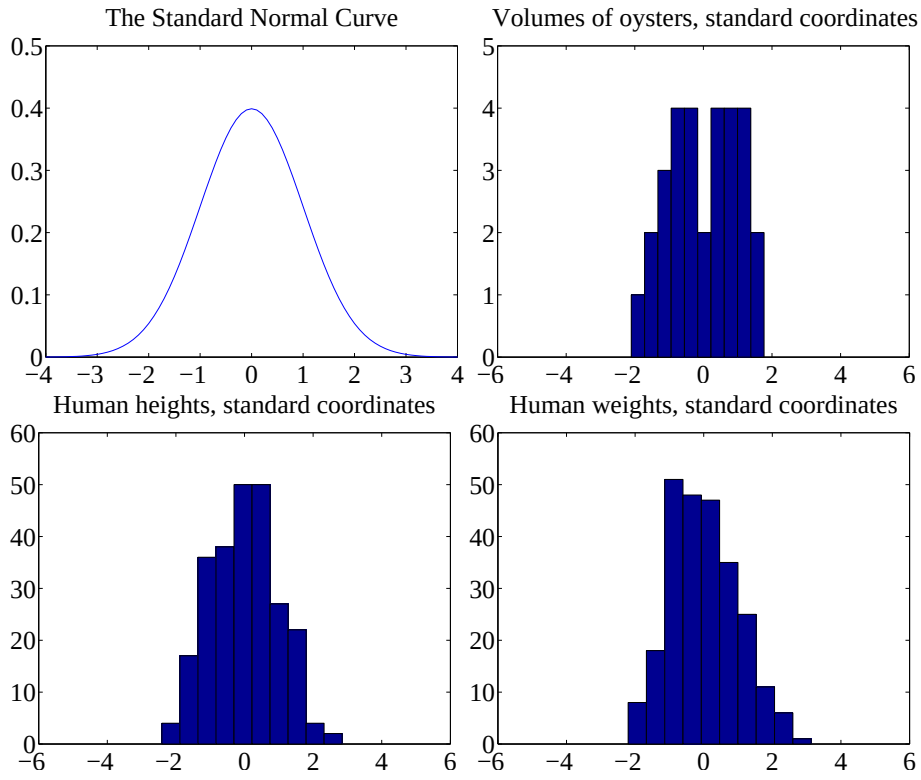

FIGURE 2.7: *Data is standard normal data when its histogram takes a stylized, bell-shaped form, plotted above. One usually requires a lot of data and very small histogram boxes for this form to be reproduced closely. Nonetheless, the histogram for normal data is unimodal (has a single bump) and is symmetric; the tails fall off fairly fast, and there are few data items that are many standard deviations from the mean. Many quite different data sets have histograms that are similar to the normal curve; I show three such datasets here.*

Definition: 2.10 *Normal data*

Data is **normal data** if, when we subtract the mean and divide by the standard deviation (i.e. compute standard coordinates), it becomes standard normal data.

It is not always easy to tell whether data is normal or not, and there are a variety of tests one can use, which we discuss later. However, there are many examples of normal data. Figure 2.7 shows a diverse variety of data sets, plotted as histograms in standard coordinates. These include: the volumes of 30 oysters (from http://www.amstat.org/publications/jse/jse_data_archive.htm; look for 30oys-

ters.dat.txt); human heights (from http://www2.stetson.edu/~jrasp/data.htm; look for bodyfat.xls, and notice that I removed two outliers); and human weights (from http://www2.stetson.edu/~jrasp/data.htm; look for bodyfat.xls, again, I removed two outliers).

Properties of normal data For the moment, assume we know that a dataset is normal. Then we expect it to have the following properties:

- If we normalize it, its histogram will be close to the standard normal curve. This means, among other things, that the data is not significantly skewed.
- About 68% of the data lie within one standard deviation of the mean. We will prove this later.
- About 95% of the data lie within two standard deviations of the mean. We will prove this later.
- About 99% of the data lie within three standard deviations of the mean. We will prove this later.

In turn, these properties imply that data that contains outliers (points many standard deviations away from the mean) is not normal. This is usually a very safe assumption. It is quite common to model a dataset by excluding a small number of outliers, then modelling the remaining data as normal. For example, if I exclude two outliers from the height and weight data from http://www2.stetson.edu/~jrasp/data.htm, the data looks pretty close to normal.

2.4.3 Boxplots

It is usually hard to compare multiple histograms by eye. One problem with comparing histograms is the amount of space they take up on a plot, because each histogram involves multiple vertical bars. This means it is hard to plot multiple overlapping histograms cleanly. If you plot each one on a separate figure, you have to handle a large number of separate figures; either you print them too small to see enough detail, or you have to keep flipping over pages.

A **boxplot** is a way to plot data that simplifies comparison. A boxplot displays a dataset as a vertical picture. There is a vertical box whose height corresponds to the interquartile range of the data (the width is just to make the figure easy to interpret). Then there is a horizontal line for the median; and the behavior of the rest of the data is indicated with whiskers and/or outlier markers. This means that each dataset makes is represented by a vertical structure, making it easy to show multiple datasets on one plot *and interpret the plot* (Figure 2.8).

To build a boxplot, we first plot a box that runs from the first to the third quartile. We then show the median with a horizontal line. We then decide which data items should be outliers. A variety of rules are possible; for the plots I show, I used the rule that data items that are larger than $q_3 + 1.5(q_3 - q_1)$ or smaller than $q_1 - 1.5(q_3 - q_1)$, are outliers. This criterion looks for data items that are more than one and a half interquartile ranges above the third quartile, or more than one and a half interquartile ranges below the first quartile.

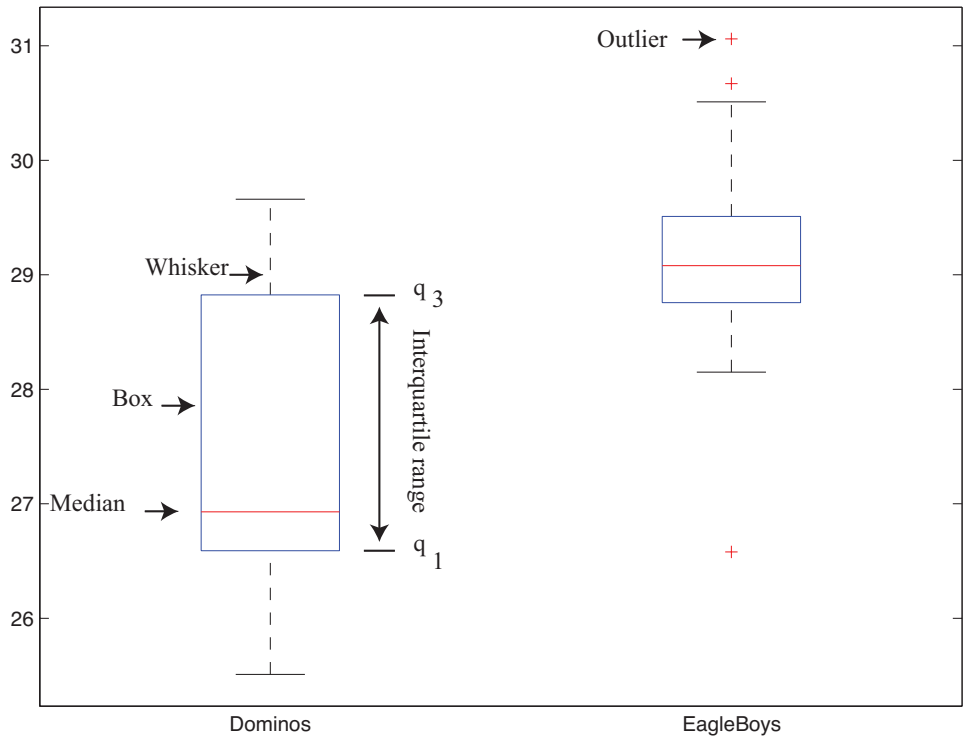

FIGURE 2.8: *A boxplot showing the box, the median, the whiskers and two outliers. Notice that we can compare the two datasets rather easily; the next section explains the comparison.*

Once we have identified outliers, we plot these with a special symbol (crosses in the plots I show). We then plot whiskers, which show the range of non-outlier data. We draw a whisker from q_1 to the smallest data item that is not an outlier, and from q_3 to the largest data item that is not an outlier. While all this sounds complicated, any reasonable programming environment will have a function that will do it for you. Figure 2.8 shows an example boxplot. Notice that the rich graphical structure means it is quite straightforward to compare two histograms.

2.5 WHOSE IS BIGGER? INVESTIGATING AUSTRALIAN PIZZAS

At http://www.amstat.org/publications/jse/jse_data_archive.htm), you will find a dataset giving the diameter of pizzas, measured in Australia (search for the word "pizza"). This website also gives the backstory for this dataset. Apparently, EagleBoys pizza claims that their pizzas are always bigger than Dominos pizzas, and published a set of measurements to support this claim (the measurements were available at http://www.eagleboys.com.au/realsizepizza as of Feb 2012, but seem not to be there anymore).

Whose pizzas are bigger? and why? A histogram of all the pizza sizes appears

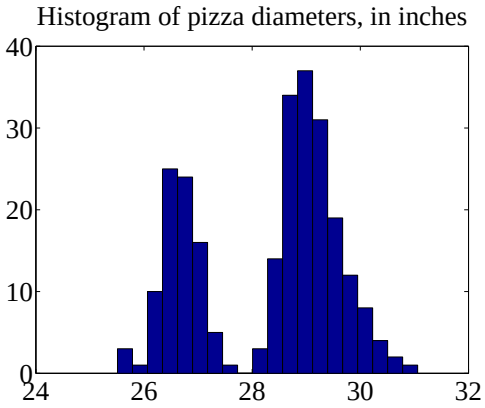

FIGURE 2.9: *A histogram of pizza diameters from the dataset described in the text. Notice that there seem to be two populations.*

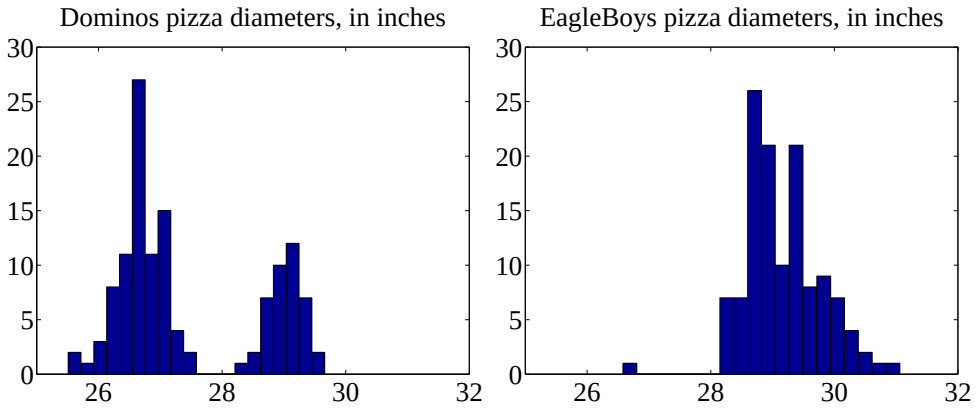

FIGURE 2.10: *On the* **left**, *the class-conditional histogram of Dominos pizza diameters from the pizza data set; on the* **right**, *the class-conditional histogram of EagleBoys pizza diameters. Notice that EagleBoys pizzas seem to follow the pattern we expect — the diameters are clustered tightly around a mean, and there is a small standard deviation — but Dominos pizzas do not seem to be like that. There is more to understand about this data.*

in figure 2.9. We would not expect every pizza produced by a restaurant to have exactly the same diameter, but the diameters are probably pretty close to one another, and pretty close to some standard value. This would suggest that we'd expect to see a histogram which looks like a single, rather narrow, bump about a mean. This is not what we see in figure 2.9 — instead, there are two bumps, which suggests two populations of pizzas. This isn't particularly surprising, because we know that some pizzas come from EagleBoys and some from Dominos.

If you look more closely at the data in the dataset, you will notice that each data item is tagged with the company it comes from. We can now easily plot

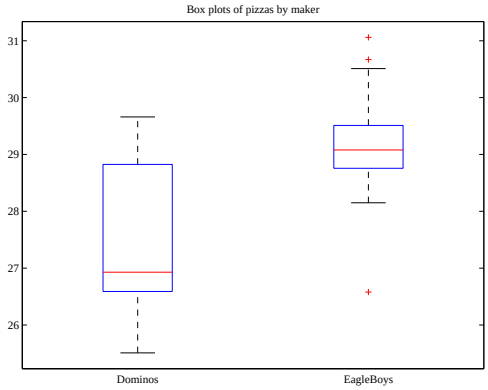

FIGURE 2.11: *Boxplots of the pizza data, comparing EagleBoys and Dominos pizza. There are several curiosities here: why is the range for Dominos so large (25.5-29)? EagleBoys has a smaller range, but has several substantial outliers; why? One would expect pizza manufacturers to try and control diameter fairly closely, because pizzas that are too small present risks (annoying customers; publicity; hostile advertising) and pizzas that are too large should affect profits.*

conditional histograms, conditioning on the company that the pizza came from. These appear in figure 2.10. Notice that EagleBoys pizzas seem to follow the pattern we expect — the diameters are clustered tightly around one value — but Dominos pizzas do not seem to be like that. This is reflected in a boxplot (figure 2.11), which shows the range of Dominos pizza sizes is surprisingly large, and that EagleBoys pizza sizes have several large outliers. There is more to understand about this data. The dataset contains labels for the type of crust and the type of topping — perhaps these properties affect the size of the pizza?

EagleBoys produces DeepPan, MidCrust and ThinCrust pizzas, and Dominos produces DeepPan, ClassicCrust and ThinNCrispy pizzas. This may have something to do with the observed patterns, but comparing six histograms by eye is unattractive. A boxplot is the right way to compare these cases (figure 2.12). The boxplot gives some more insight into the data. Dominos thin crust appear to have a narrow range of diameters (with several outliers), where the median pizza is rather larger than either the deep pan or the classic crust pizza. EagleBoys pizzas all have a range of diameters that is (a) rather similar across the types and (b) rather a lot like the Dominos thin crust. There are outliers, but few for each type.

Another possibility is that the variation in size is explained by the topping. We can compare types and toppings by producing a set of conditional boxplots (i.e. the diameters for each type and each topping). This leads to rather a lot of boxes (figure 2.13), but they're still easy to compare by eye. The main difficulty is that the labels on the plot have to be shortened. I made labels using the first letter from the manufacturer ("D" or "E"); the first letter from the crust type (previous paragraph); and the first and last letter of the topping. Toppings for Dominos are: Hawaiian; Supreme; BBQMeatlovers. For EagleBoys, toppings are: Hawaiian; SuperSupremo; and BBQMeatlovers. This gives the labels: 'DCBs'; (Dominos; Clas-

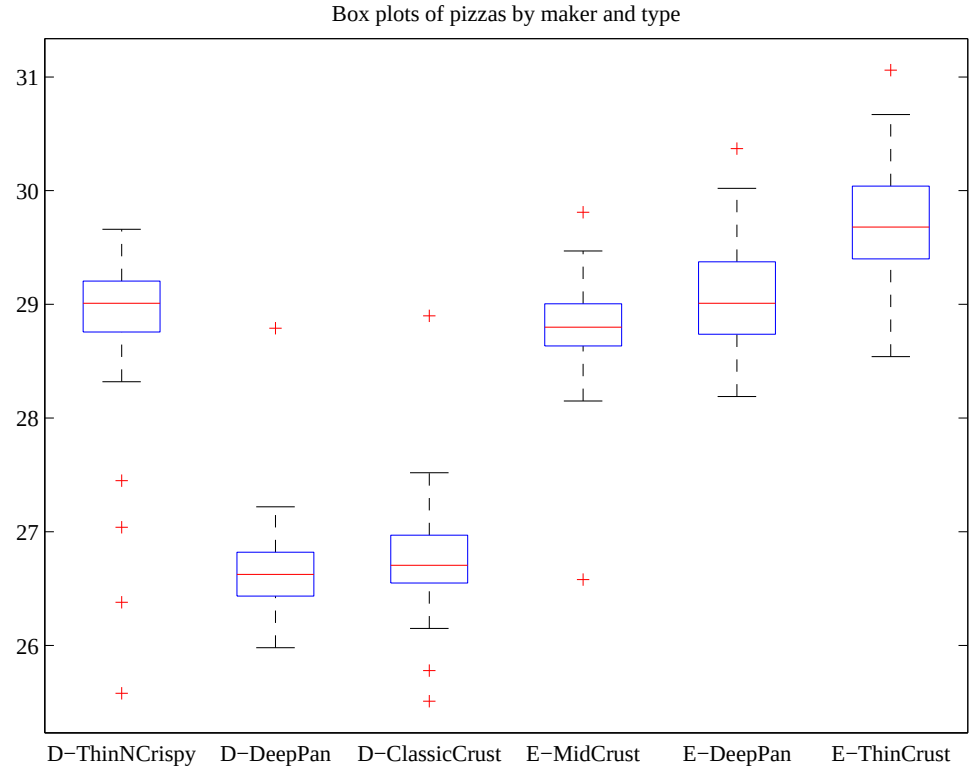

FIGURE 2.12: *Boxplots for the pizza data, broken out by type (thin crust, etc.).*

sicCrust; BBQMeatlovers); 'DCHn'; 'DCSe'; 'DDBs'; 'DDHn'; 'DDSe'; 'DTBs'; 'DTHn'; 'DTSe'; 'EDBs'; 'EDHn'; 'EDSo'; 'EMBs'; 'EMHn'; 'EMSo'; 'ETBs'; 'ETHn'; 'ETSo'. Figure 2.13 suggests that the topping isn't what is important, but the crust (group the boxplots by eye).

What could be going on here? One possible explanation is that Eagleboys have tighter control over the size of the final pizza. One way this could happen is that all EagleBoys pizzas start the same size and shrink the same amount in baking, whereas all Dominos pizzas start a standard diameter, but different Dominos crusts shrink differently in baking. Another way is that Dominos makes different size crusts for different types, but that the cooks sometimes get confused. Yet another possibility is that Dominos controls portions by the mass of dough (so thin crust diameters tend to be larger), but Eagleboys controls by the diameter of the crust.

You should notice that this is more than just a fun story. If you were a manager at a pizza firm, you'd need to make choices about how to control costs. Labor costs, rent, and portion control (i.e. how much pizza, topping, etc. a customer gets for their money) are the main thing to worry about. If the same kind of pizza has a wide range of diameters, you have a problem, because some customers are getting too much (which affects your profit) or too little (which means they might call someone

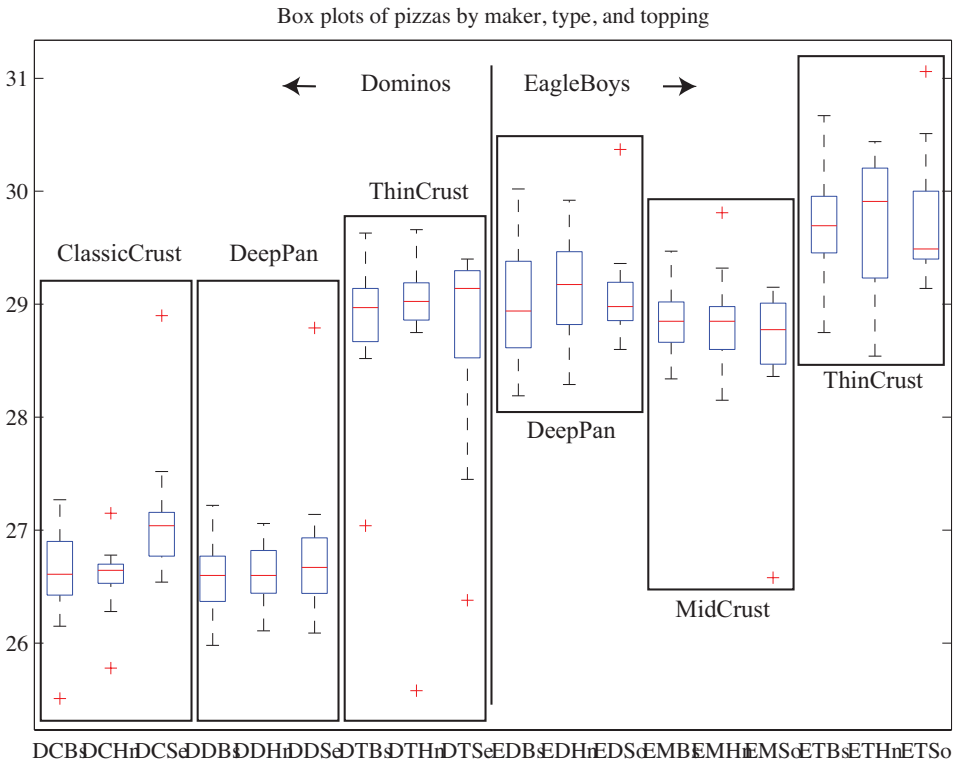

FIGURE 2.13: *The pizzas are now broken up by topping as well as crust type (look at the source for the meaning of the names). I have separated Dominos from Eagleboys with a vertical line, and grouped each crust type with a box. It looks as though the issue is not the type of topping, but the crust. Eagleboys seems to have tighter control over the size of the final pizza.*

else next time). But making more regular pizzas might require more skilled (and so more expensive) labor. The fact that Dominos and EagleBoys seem to be following different strategies successfully suggests that more than one strategy might work. But you can't choose if you don't know what's happening. As I said at the start, "what's going on here?" is perhaps the single most useful question anyone can ask.

2.6 YOU SHOULD

2.6.1 remember these definitions:

2.6.2 remember these terms:

2.6.3 remember these facts:

2.6.4 be able to:

- Plot a bar chart for a dataset.
- Plot a histogram for a dataset.

- Tell whether the histogram is skewed or not, and in which direction.
- Plot and interpret conditional histograms.
- Compute basic summaries for a dataset, including mean, median, standard deviation and interquartile range.
- Plot a box plot for one or several datasets.
- Interpret a box plot.
- Use histograms, summaries and box plots to investigate datasets.

PROBLEMS

2.1. Show that $\mathsf{mean}(\{kx\}) = k\mathsf{mean}(\{x\})$ by substituting into the definition.
2.2. Show that $\mathsf{mean}(\{x + c\}) = \mathsf{mean}(\{x\}) + c$ by substituting into the definition.
2.3. Show that $\sum_{i=1}^{N}(x_i - \mathsf{mean}(\{x\})) = 0$ by substituting into the definition.
2.4. Show that $\mathsf{std}(\{x + c\}) = \mathsf{std}(\{x\})$ by substituting into the definition (you'll need to recall the properties of the mean to do this).
2.5. Show that $\mathsf{std}(\{kx\}) = k\mathsf{std}(\{x\})$ by substituting into the definition (you'll need to recall the properties of the mean to do this).
2.6. Show that $\mathsf{median}(\{x + c\}) = \mathsf{median}(\{x\}) + c$ by substituting into the definition.
2.7. Show that $\mathsf{median}(\{kx\}) = k\mathsf{median}(\{x\})$ by substituting into the definition.
2.8. Show that $\mathsf{iqr}\{x + c\} = \mathsf{iqr}\{x\}$ by substituting into the definition.
2.9. Show that $\mathsf{iqr}\{kx\} = k\mathsf{iqr}\{x\}$ by substituting into the definition.

PROGRAMMING EXERCISES

2.10. You can find a data set showing the number of barrels of oil produced, per year for the years 1880-1984 at http://lib.stat.cmu.edu/DASL/Datafiles/Oilproduction.html. Is a mean a useful summary of this dataset? Why?
2.11. You can find a dataset giving the cost (in 1976 US dollars), number of megawatts, and year of construction of a set of nuclear power plants at http://lib.stat.cmu.edu/DASL/Datafiles/NuclearPlants.html.
(a) Are there outliers in this data?
(b) What is the mean cost of a power plant? What is the standard deviation?
(c) What is the mean cost per megawatt? What is the standard deviation?
(d) Plot a histogram of the cost per megawatt. Is it skewed? Why?
2.12. You can find a dataset giving the sodium content and calorie content of three types of hot dog at http://lib.stat.cmu.edu/DASL/Datafiles/Hotdogs.html. The types are Beef, Poultry, and Meat (a rather disturbingly vague label). Use class-conditional histograms to compare these three types of hot dog with respect to sodium content and calories.
2.13. You will find a dataset giving (among other things) the number of 3 or more syllable words in advertising copy appearing in magazines at http://lib.stat.cmu.edu/DASL/Datafiles/magadsdat.html. The magazines are grouped by the education level of their readers; the groups are 1, 2, and 3 (the variable is called GRP in the data).
(a) Use a boxplot to compare the number of three or more syllable words for the ads in magazines in these three groups. What do you see?
(b) Use a boxplot to compare the number of sentences appearing in the ads in magazines in these three groups. What do you see?
2.14. You can find a dataset recording a variety of properties of secondary school students in Portugal at http://archive.ics.uci.edu/ml/datasets/STUDENT+ALCOHOL+CONSUMPTION. This dataset was collected by P. Cortez and A. Silva, and is hosted by the UC Irvine Machine Learning Repository. There are two datasets; one for students in a math course, and another for students in a Portugese language course.
(a) Use plots of conditional histograms to investigate whether math students drink more alcohol during the week than Portugese language students.
(b) Use plots of conditional histograms to investigate whether students from small families drink more alcohol at the weekend than those from large families.

(c) Each of the variables school, sex, famsize and romantic has two possible values. This means that if we characterize students by the values of these variables, there are sixteen possible types of student. Use box plots to investigate which of these types drinks more alcohol in total.

2.15. You can find a dataset recording a variety of properties of Taiwanese credit card holders at http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients. This dataset was collected by I-Cheng Yeh, and is hosted by the UC Irvine Machine Learning Repository. There is a variable indicating whether a holder defaulted or not, and a variety of other variables.

(a) Use plots of conditional histograms to investigate whether people who default have more debt (use the variable X1 for debt) than those who don't default.

(b) Use box plots to investigate whether gender, education or marital status has any effect on the amount of debt (again, use X1 for debt).

CHAPTER 3

Looking at Relationships

We think of a dataset as a collection of d-tuples (a d-tuple is an ordered list of d elements). For example, the Chase and Dunner dataset had entries for Gender; Grade; Age; Race; Urban/Rural; School; Goals; Grades; Sports; Looks; and Money (so it consisted of 11-tuples). The previous chapter explored methods to visualize and summarize a set of values obtained by extracting a single element from each tuple. For example, I could visualize the heights or the weights of a population (as in Figure 2.7). But I could say nothing about the relationship between the height and weight. In this chapter, we will look at methods to visualize and summarize the relationships between pairs of elements of a dataset.

3.1 PLOTTING 2D DATA

We take a dataset, choose two different entries, and extract the corresponding elements from each tuple. The result is a dataset consisting of 2-tuples, and we think of this as a two dimensional dataset. The first step is to plot this dataset in a way that reveals relationships. The topic of how best to plot data fills many books, and we can only scratch the surface here. Categorical data can be particularly tricky, because there are a variety of choices we can make, and the usefulness of each tends to depend on the dataset and to some extent on one's cleverness in graphic design (section 3.1.1).

For some continuous data, we can plot the one entry as a function of the other (so, for example, our tuples might consist of the date and the number of robberies; or the year and the price of lynx pelts; and so on, section 3.1.2).

Mostly, we use a simple device, called a scatter plot. Using and thinking about scatter plots will reveal a great deal about the relationships between our data items (section 3.1.3).

3.1.1 Categorical Data, Counts, and Charts

Categorical data is a bit special. Assume we have a dataset with several categorical descriptions of each data item. One way to plot this data is to think of it as belonging to a richer set of categories. Assume the dataset has categorical descriptions, which are not ordinal. Then we can construct a new set of categories by looking at each of the cases for each of the descriptions. For example, in the Chase and Dunner data of table 2.2, our new categories would be: "boy-sports"; "girl-sports"; "boy-popular"; "girl-popular"; "boy-grades"; and "girl-grades". A large set of categories like this can result in a poor bar chart, though, because there may be too many bars to group the bars successfully. Figure 3.1 shows such a bar chart. Notice that it is hard to group categories by eye to compare; for example, you can see that slightly more girls think grades are important than boys do, but to do so you need to compare two bars that are separated by two other bars. An alternative is a **pie**

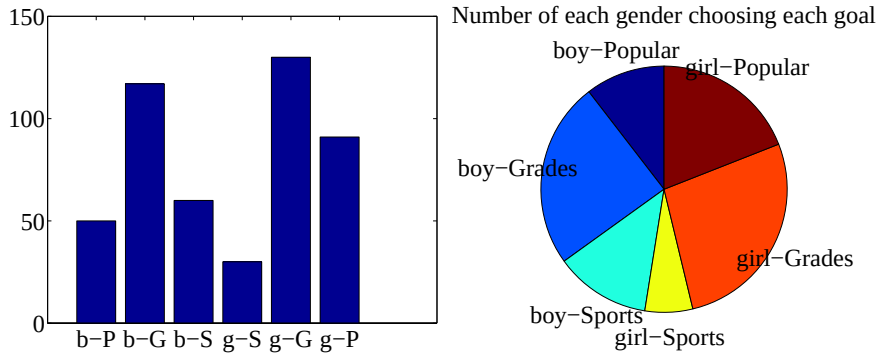

FIGURE 3.1: *I sorted the children in the Chase and Dunner study into six categories (two genders by three goals), and counted the number of children that fell into each cell. I then produced the bar chart on the* **left***, which shows the number of children of each gender, selecting each goal. On the* **right***, a pie chart of this information. I have organized the pie chart so it is easy to compare boys and girls by eye — start at the top; going down on the left side are boy goals, and on the right side are girl goals. Comparing the size of the corresponding wedges allows you to tell what goals boys (resp. girls) identify with more or less often.*

chart, where a circle is divided into sections whose angle is proportional to the size of the data item. You can think of the circle as a pie, and each section as a slice of pie. Figure 3.1 shows a pie chart, where each section is proportional to the number of students in its category. In this case, I've used my judgement to lay the categories out in a way that makes comparisons easy. I'm not aware of any tight algorithm for doing this, though.

Pie charts have problems, because it is hard to judge small differences in area accurately by eye. For example, from the pie chart in figure 3.1, it's hard to tell that the "boy-sports" category is slightly bigger than the "boy-popular" category (try it; check using the bar chart). For either kind of chart, it is quite important to think about *what* you plot. For example, the plot of figure 3.1 shows the total number of respondents, and if you refer to figure 2.1, you will notice that there are slightly more girls in the study. Is the *percentage* of boys who think grades are important smaller (or larger) than the *percentage* of girls who think so? you can't tell from these plots, and you'd have to plot the percentages instead.

An alternative is to use a **stacked bar chart**. You can (say) regard the data as of two types, "Boys" and "Girls". Within those types, there are subtypes ("Popularity", "Grades" and "Sport"). The height of the bar is given by the number of elements in the type, and the bar is divided into sections corresponding to the number of elements of that subtype. Alternatively, if you want the plot to show relative frequencies, the bars could all be the same height, but the shading corresponds to the fraction of elements of that subtype. This is all much harder to say than to see or to do (Figure 3.2).

An alternative to a pie chart that is very useful for two dimensional data is a **heat map**. This is a method of displaying a matrix as an image. Each entry of

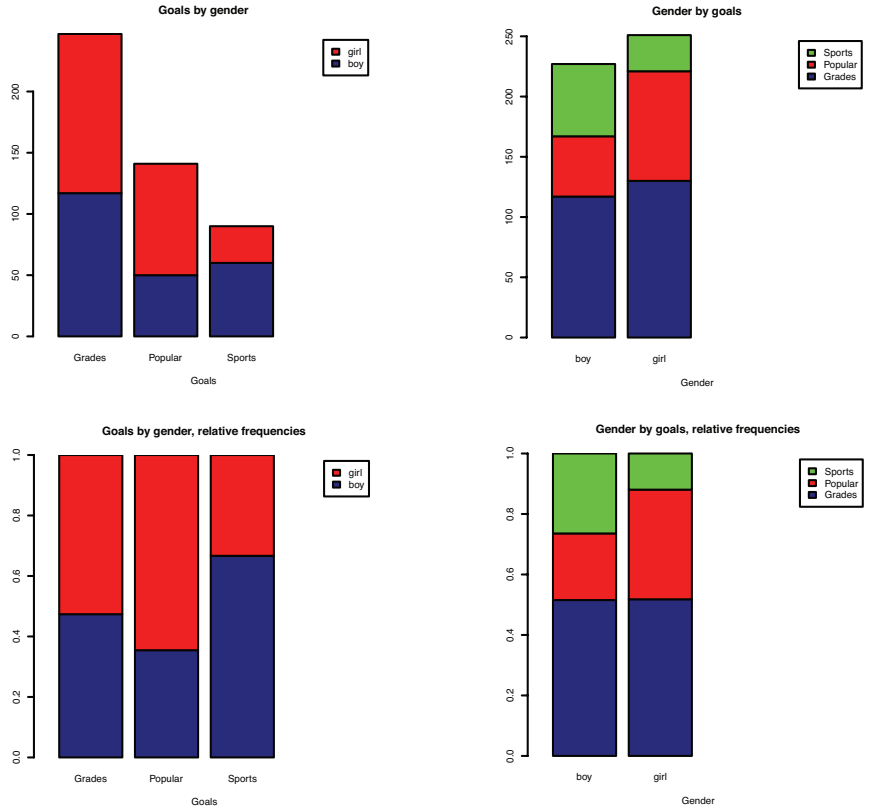

FIGURE 3.2: *These bar charts use stacked bars. In the* **top row**, *the overall height of the bar is given by the number of elements of that type but each different subtype is identified by shading, so you can tell by eye, for example, how many of the "Grades" in the study were "Boys". This layout makes it hard to tell what* fraction *of, say, "Boys" aspire to "Popularity". In the* **bottom row**, *all bars have the same height, but the shading of the bar identifies the fraction of that type that has a corresponding subtype. This means you can tell by eye what fraction of "Girls" aspire to "Sports".*

the matrix is mapped to a color, and the matrix is represented as an image. For the Chase and Dunner study, I constructed a matrix where each row corresponds to a choice of "sports", "grades", or "popular", and each column corresponds to a choice of "boy" or "girl". Each entry contains the count of data items of that type. Zero values are represented as white; the largest values as red; and as the value increases, we use an increasingly saturated pink. This plot is shown in figure 3.3

If the categorical data is ordinal, the ordering offers some hints for making a good plot. For example, imagine we are building a user interface. We build an initial version, and collect some users, asking each to rate the interface on scales for "ease of use" (-2, -1, 0, 1, 2, running from bad to good) and "enjoyability" (again, -2, -1, 0, 1, 2, running from bad to good). It is natural to build a 5x5 table, where

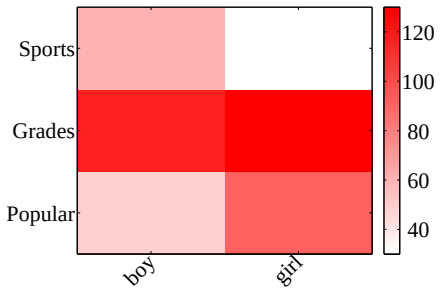

FIGURE 3.3: *A heat map of the Chase and Dunner data. The color of each cell corresponds to the count of the number of elements of that type. The colorbar at the side gives the correspondence between color and count. You can see at a glance that the number of boys and girls who prefer grades is about the same; that about the same number of boys prefer sports and popularity, with sports showing a mild lead; and that more girls prefer popularity to sports.*

	-2	-1	0	1	2
-2	24	5	0	0	1
-1	6	12	3	0	0
0	2	4	13	6	0
1	0	0	3	13	2
2	0	0	0	1	5

TABLE 3.1: *I simulated data representing user evaluations of a user interface. Each cell in the table on the* **left** *contains the count of users rating "ease of use" (horizontal, on a scale of -2 -very bad- to 2 -very good) vs. "enjoyability" (vertical, same scale). Users who found the interface hard to use did not like using it either. While this data is categorical, it's also ordinal, so that the order of the cells is determined. It wouldn't make sense, for example, to reorder the columns of the table or the rows of the table.*

each cell represents a pair of "ease of use" and "enjoyability" values. We then count the number of users in each cell, and build graphical representations of this table. One natural representation is a **3D bar chart**, where each bar sits on its cell in the 2D table, and the height of the bars is given by the number of elements in the cell. Table 3.1 shows a table and figure 3.4 shows a 3D bar chart for some simulated data. The main difficulty with a 3D bar chart is that some bars are hidden behind others. This is a regular nuisance. You can improve things by using an interactive tool to rotate the chart to get a nice view, but this doesn't always work. Heatmaps don't suffer from this problem (Figure 3.4), another reason they are a good choice.

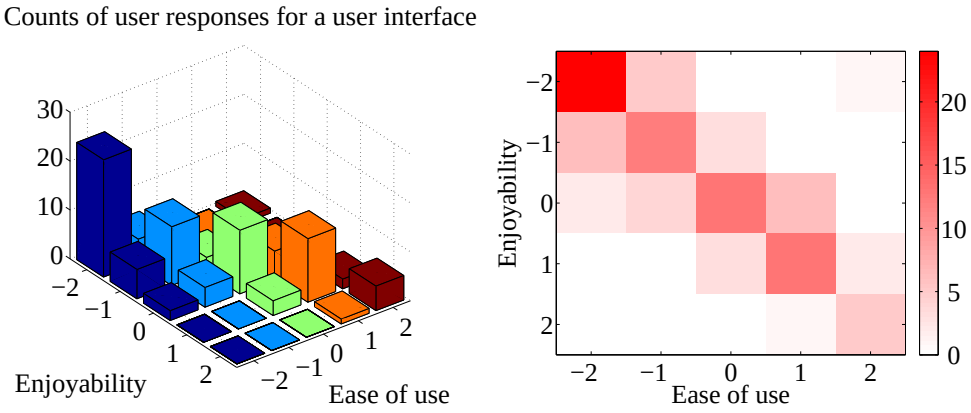

FIGURE 3.4: *On the* **left**, *a 3D bar chart of the data. The height of each bar is given by the number of users in each cell. This figure immediately reveals that users who found the interface hard to use did not like using it either. However, some of the bars at the back are hidden, so some structure might be hard to infer. On the* **right**, *a heat map of this data. Again, this figure immediately reveals that users who found the interface hard to use did not like using it either. It's more apparent that everyone disliked the interface, though, and it's clear that there is no important hidden structure.*

> **Remember this:** *There are a variety of tools for plotting categorical data. It's difficult to give strict rules for which to use when, but usually one tries to avoid pie charts (angles are hard to judge by eye) and 3D bar charts (where occlusion can hide important effects).*

3.1.2 Series

Sometimes one component of a dataset gives a natural ordering to the data. For example, we might have a dataset giving the maximum rainfall for each day of the year. We could record this either by using a two-dimensional representation, where one dimension is the number of the day and the other is the temperature, or with a convention where the i'th data item is the rainfall on the i'th day. For example, at http://lib.stat.cmu.edu/DASL/Datafiles/timeseriesdat.html, you can find four datasets indexed in this way. It is natural to plot data like this as a function of time. From this dataset, I extracted data giving the number of burglaries each month in a Chicago suburb, Hyde Park. I have plotted part this data in Figure 3.5 (I left out the data to do with treatment effects). It is natural to plot a graph of the burglaries as a function of time (in this case, the number of the month). The plot shows each data point explicitly. I also told the plotting software to draw lines joining data points, because burglaries do not all happen on a specific day.

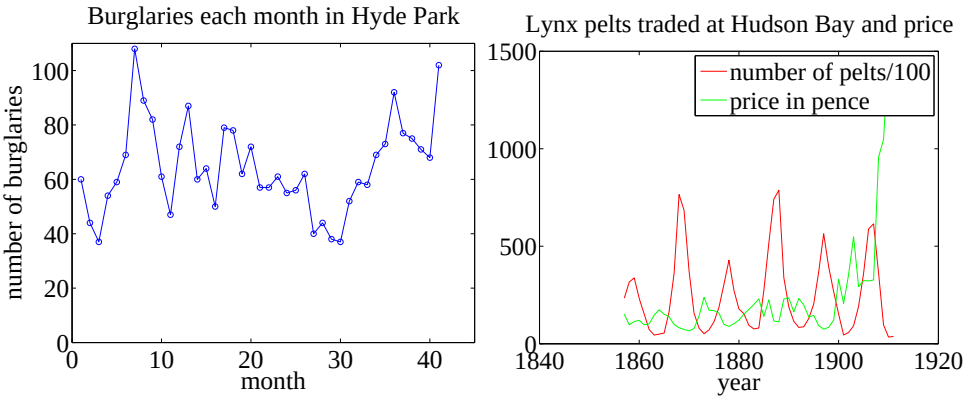

FIGURE 3.5: **Left**, *the number of burglaries in Hyde Park, by month.* **Right**, *a plot of the number of lynx pelts traded at Hudson Bay and of the price paid per pelt, as a function of the year. Notice the scale, and the legend box (the number of pelts is scaled by 100).*

The lines suggest, reasonably enough, the rate at which burglaries are happening between data points.

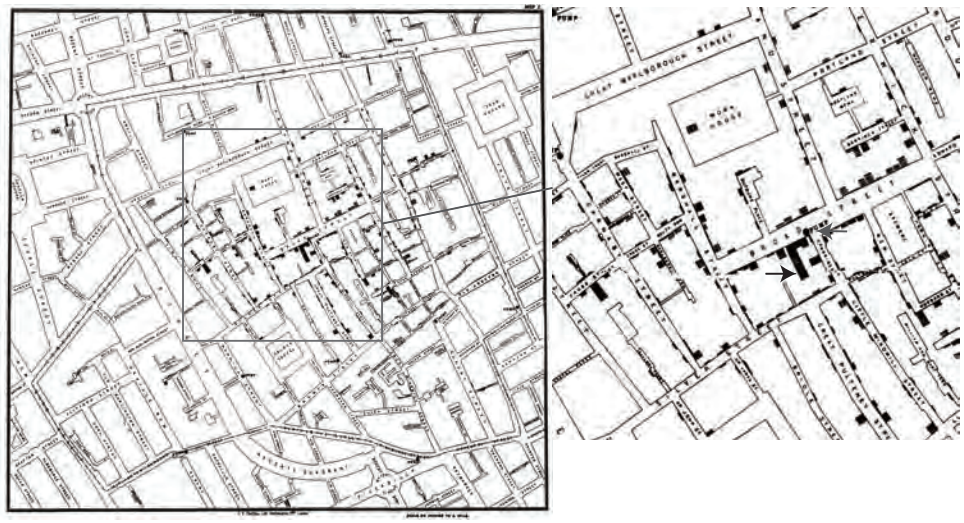

FIGURE 3.6: *Snow's scatter plot of cholera deaths on the* **left**. *Each cholera death is plotted as a small bar on the house in which the bar occurred (for example, the black arrow points to one stack of these bars, indicating many deaths, in the detail on the* **right**). *Notice the fairly clear pattern of many deaths close to the Broad street pump (grey arrow in the detail), and fewer deaths further away (where it was harder to get water from the pump).*

As another example, at http://lib.stat.cmu.edu/datasets/Andrews/ you can

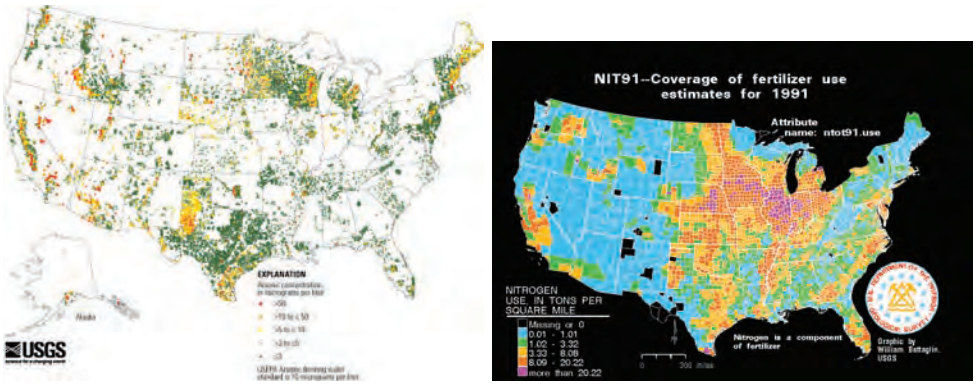

FIGURE 3.7: **Left**, *a scatter plot of arsenic levels in US groundwater, prepared by the US Geological Survey (you can find the data at* http://water.usgs.gov/GIS/metadata/usgswrd/XML/arsenic_map.xml*. Here the shape and color of each marker shows the amount of arsenic, and the spatial distribution of the markers shows where the wells were sampled.* **Right**, *the usage of Nitrogen (a component of fertilizer) by US county in 1991, prepared by the US Geological Survey (you can find the data at* http://water.usgs.gov/GIS/metadata/usgswrd/XML/nit91.xml*). In this variant of a scatter plot (which usually takes specialized software to prepare) one fills each region with a color indicating the data in that region.*

find a dataset that records the number of lynx pelts traded to the Hudson's Bay company and the price paid for each pelt. This version of the dataset appeared first in table 3.2 of *Data: a Collection of Problems from many Fields for the Student and Research Worker* by D.F. Andrews and A.M. Herzberg, published by Springer in 1985. I have plotted it in figure 3.5. The dataset is famous, because it shows a periodic behavior in the number of pelts (which is a good proxy for the number of lynx), which is interpreted as a result of predator-prey interactions. Lynx eat rabbits. When there are many rabbits, lynx kittens thrive, and soon there will be many lynx; but then they eat most of the rabbits, and starve, at which point the rabbit population rockets. You should also notice that after about 1900, prices seem to have gone up rather quickly. I don't know why this is. There is also some suggestion, as there should be, that prices are low when there are many pelts, and high when there are few.

3.1.3 Scatter Plots for Spatial Data

It isn't always natural to plot data as a function. For example, in a dataset containing the temperature and blood pressure of a set of patients, there is no reason to believe that temperature is a function of blood pressure, or the other way round. Two people could have the same temperature, and different blood pressures, or vice-versa. As another example, we could be interested in what causes people to die of cholera. We have data indicating *where* each person died in a particular outbreak. It isn't helpful to try and plot such data as a function.

The **scatter plot** is a powerful way to deal with this situation. In the first

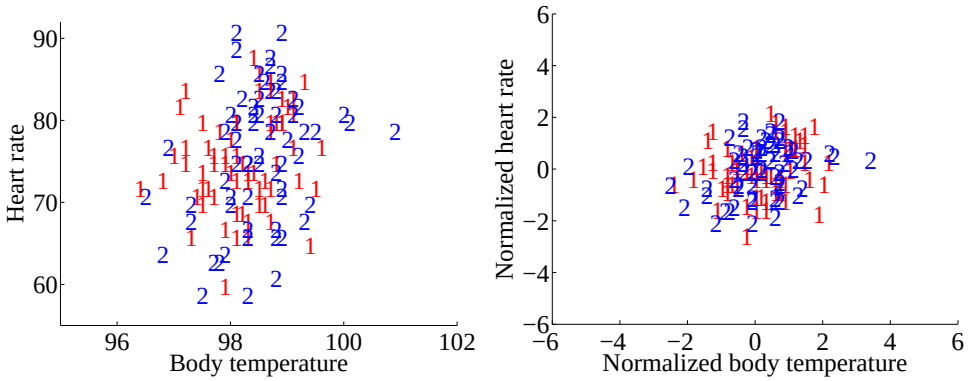

FIGURE 3.8: *A scatter plot of body temperature against heart rate, from the dataset at http://www2.stetson.edu/~jrasp/data.htm; normtemp.xls. I have separated the two genders by plotting a different symbol for each (though I don't know which gender is indicated by which letter); if you view this in color, the differences in color makes for a greater separation of the scatter. This picture suggests, but doesn't conclusively establish, that there isn't much dependence between temperature and heart rate, and any dependence between temperature and heart rate isn't affected by gender.*

instance, assume that our data points actually describe points on the a real map. Then, to make a scatter plot, we make a mark on the map at a place indicated by each data point. What the mark looks like, and how we place it, depends on the particular dataset, what we are looking for, how much we are willing to work with complex tools, and our sense of graphic design.

Figure 3.6 is an extremely famous scatter plot, due to John Snow. Snow — one of the founders of epidemiology — used a scatter plot to reason about a cholera outbreak centered on the Broad Street pump in London in 1854. At that time, the mechanism that causes cholera was not known. Snow plotted cholera deaths as little bars (more bars, more deaths) on the location of the house where the death occurred. More bars means more deaths, fewer bars means fewer deaths. There are more bars per block close to the pump, and few far away. This plot offers quite strong evidence of an association between the pump and death from cholera. Snow used this scatter plot as evidence that cholera was associated with water, and that the Broad Street pump was the source of the tainted water.

Figure 3.7 shows a scatter plot of arsenic levels in groundwater for the United States, prepared by the US Geological Survey. The data set was collected by Focazio and others in 2000; by Welch and others in 2000; and then updated by Ryker 2001. It can be found at http://water.usgs.gov/GIS/metadata/usgswrd/XML/arsenic_map.xml. One variant of a scatter plot that is particularly useful for geographic data occurs when one fills regions on a map with different colors, following the data in that region. Figure 3.7 shows the nitrogen usage by US county in 1991; again, this figure was prepared by the US Geological Survey.

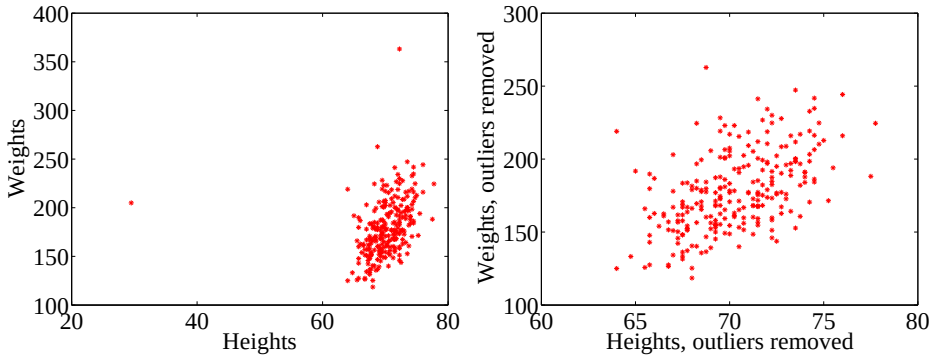

FIGURE 3.9: *A scatter plots of weight against height, from the dataset at* http://www2.stetson.edu/~jrasp/data.htm. **Left:** *Notice how two outliers dominate the picture, and to show the outliers, the rest of the data has had to be bunched up.* **Right** *shows the data with the outliers removed. The structure is now somewhat clearer.*

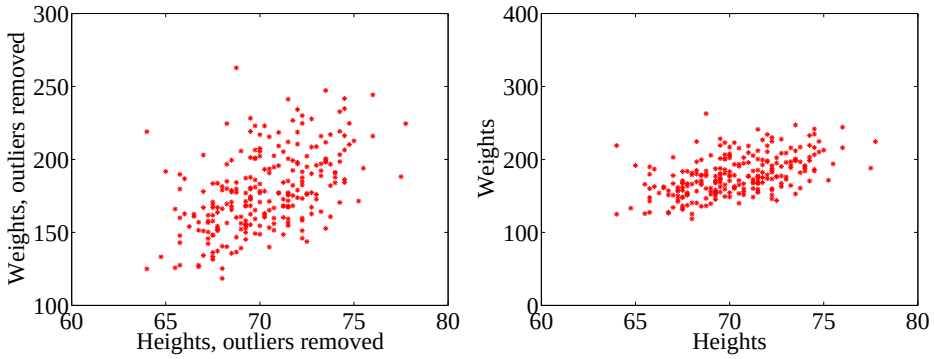

FIGURE 3.10: *Scatter plots of weight against height, from the dataset at* http://www2.stetson.edu/~jrasp/data.htm. **Left:** *data with two outliers removed, as in figure 3.9.* **Right:** *this data, rescaled slightly. Notice how the data looks less spread out. But there is no difference between the datasets. Instead, your eye is easily confused by a change of scale.*

Remember this: *Scatter plots are a most effective tool for geographic data and 2D data in general. A scatter plot should be your first step with a new 2D dataset.*

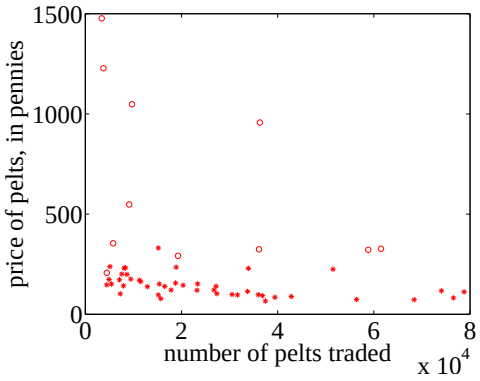

FIGURE 3.11: *A scatter plot of the price of lynx pelts against the number of pelts. I have plotted data for 1901 to the end of the series as circles, and the rest of the data as *'s. It is quite hard to draw any conclusion from this data, because the scale is confusing. Furthermore, the data from 1900 on behaves quite differently from the other data.*

3.1.4 Exposing Relationships with Scatter Plots

Scatter plots are natural for geographic data, but a scatter plot is a useful, simple tool for ferreting out associations in other kinds of data as well. Now we need some notation. Assume we have a dataset $\{x\}$ of N data items, $x_1, \ldots, x_N$. Each data item is a d dimensional vector (so its components are numbers). We wish to investigate the relationship between two components of the dataset. For example, we might be interested in the 7'th and the 13'th component of the dataset. We will produce a two-dimensional plot, one dimension for each component. It does not really matter which component is plotted on the x-coordinate and which on the y-coordinate (though it will be some pages before this is clear). But it is very difficult to write sensibly without talking about the x and y coordinates.

We will make a two-dimensional dataset out of the components that interest us. We must choose which component goes first in the resulting 2-vector. We will plot this component on the x-coordinate (and we refer to it as the x-coordinate), and to the other component as the y-coordinate. This is just to make it easier to describe what is going on; there's no important idea here. It really will not matter which is x and which is y. The two components make a dataset $\{\mathbf{x}_i\} = \{(x_i, y_i)\}$. To produce a scatter plot of this data, we plot a small shape at the location of each data item.

Such scatter plots are very revealing. For example, figure 3.8 shows a scatter plot of body temperature against heart rate for humans. In this dataset, the gender of the subject was recorded (as "1" or "2" — I don't know which is which), and so I have plotted a "1" at each data point with gender "1", and so on. Looking at the data suggests there isn't much difference between the blob of "1" labels and the blob of "2" labels, which suggests that females and males are about the same in this respect.

The scale used for a scatter plot matters. For example, plotting lengths in

meters gives a very different scatter from plotting lengths in millimeters. Figure 3.9 shows two scatter plots of weight against height. Each plot is from the same dataset, but one is scaled so as to show two outliers. Keeping these outliers means that the rest of the data looks quite concentrated, just because the axes are in large units. In the other plot, the axis scale has changed (so you can't see the outliers), but the data looks more scattered. This may or may not be a misrepresentation. Figure 3.10 compares the data with outliers removed, with the same plot on a somewhat different set of axes. One plot looks as though increasing height corresponds to increasing weight; the other looks as though it doesn't. This is purely due to deceptive scaling — each plot shows the same dataset.

Dubious data can also contribute to scaling problems. Recall that, in figure 3.5, price data before and after 1900 appeared to behave differently. Figure 3.11 shows a scatter plot of the lynx data, where I have plotted number of pelts against price. I plotted the post-1900 data as circles, and the rest as asterisks. Notice how the circles seem to form a quite different figure, which supports the suggestion that something interesting happened around 1900. We can reasonably choose to analyze data after 1900 separately from before 1900. A choice like this should be made with care. If you exclude every data point that might disagree with your hypothesis, you may miss the fact that you are wrong. Leaving out data is an essential component of many kinds of fraud. You should always reveal whether you have excluded data, and why, to allow the reader to judge the evidence.

When you look at Figure 3.11, you should notice the scatter plot does not seem to support the idea that prices go up when supply goes down. This is puzzling because it's generally a pretty reliable idea. In fact, the plot is just hard to interpret because it is poorly scaled. Scale is an important nuisance, and it's easy to get misled by scale effects.

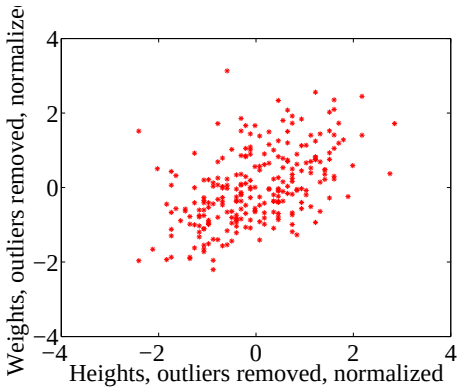

FIGURE 3.12: *A normalized scatter plot of weight against height, from the dataset at http://www2.stetson.edu/~jrasp/data.htm. Now you can see that someone who is a standard deviation taller than the mean will tend to be somewhat heavier than the mean too.*

The way to avoid the problem is to plot in standard coordinates. We can normalize without worrying about the dimension of the data — we normalize each

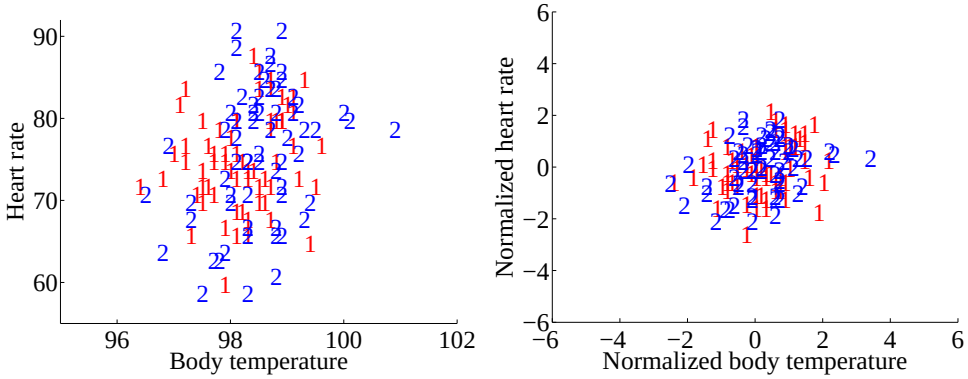

FIGURE 3.13: **Left:** *A scatter plot of body temperature against heart rate, from the dataset at http://www2.stetson.edu/~jrasp/data.htm; normtemp.xls. I have separated the two genders by plotting a different symbol for each (though I don't know which gender is indicated by which letter); if you view this in color, the differences in color makes for a greater separation of the scatter. This picture suggests, but doesn't conclusively establish, that there isn't much dependence between temperature and heart rate, and any dependence between temperature and heart rate isn't affected by gender. The scatter plot of the normalized data, in standard coordinates, on the* **right** *supports this view.*

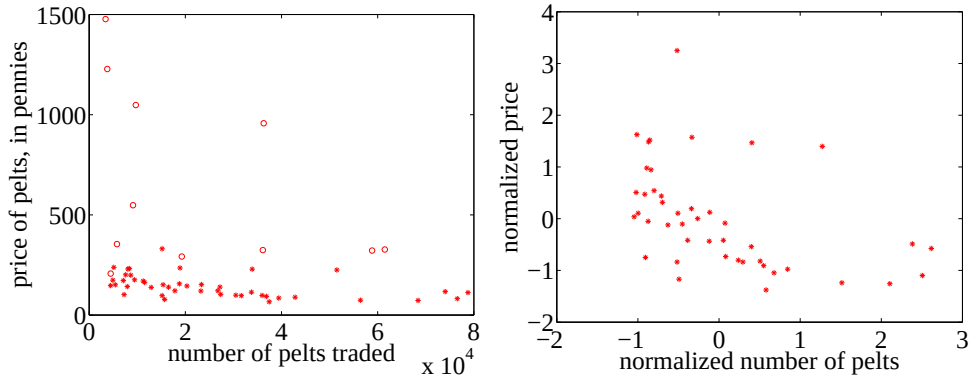

FIGURE 3.14: **Left:** *A scatter plot of the price of lynx pelts against the number of pelts (this is a repeat of figure 3.11 for reference). I have plotted data for 1901 to the end of the series as circles, and the rest of the data as *'s. It is quite hard to draw any conclusion from this data, because the scale is confusing.* **Right:** *A scatter plot of the price of pelts against the number of pelts for lynx pelts. I excluded data for 1901 to the end of the series, and then normalized both price and number of pelts. Notice that there is now a distinct trend; when there are fewer pelts, they are more expensive, and when there are more, they are cheaper.*

dimension independently by subtracting the mean of that dimension and dividing by the standard deviation of that dimension. This means we can normalize the x and y coordinates of the two-dimensional data separately. We continue to use the convention of writing the normalized x coordinate as $\hat{x}$ and the normalized y coordinate as $\hat{y}$. So, for example, we can write $\hat{x}_j = (x_j - \mathsf{mean}(\{x\}))/\mathsf{std}(\{x\}))$ for the $\hat{x}$ value of the j'th data item in normalized coordinates. Normalizing shows us the dataset on a standard scale. Once we have done this, it is quite straightforward to read off simple relationships between variables from a scatter plot.

> **Remember this:** *The plot scale can mask effects in scatter plots, and it's usually a good idea to plot in standard coordinates.*

3.2 CORRELATION

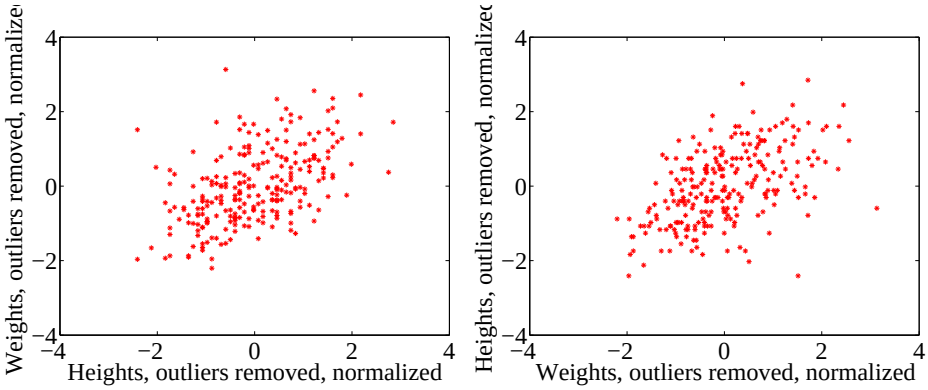

FIGURE 3.15: *On the* **left**, *a normalized scatter plot of weight (y-coordinate) against height (x-coordinate). On the* **right**, *a scatter plot of height (y-coordinate) against weight (x-coordinate). I've put these plots next to one another so you don't have to mentally rotate (which is what you should usually do).*

The simplest, and most important, relationship to look for in a scatter plot is this: when $\hat{x}$ increases, does $\hat{y}$ tend to increase, decrease, or stay the same? This is straightforward to spot in a normalized scatter plot, because each case produces a very clear shape on the scatter plot. Any relationship is called **correlation** (we will see later how to measure this), and the three cases are: positive correlation, which means that larger $\hat{x}$ values tend to appear with larger $\hat{y}$ values; zero correlation, which means no relationship; and negative correlation, which means that larger $\hat{x}$ values tend to appear with smaller $\hat{y}$ values. I have shown these cases together in one figure using a real data example (Figure 3.16), so you can compare the appearance of the plots.

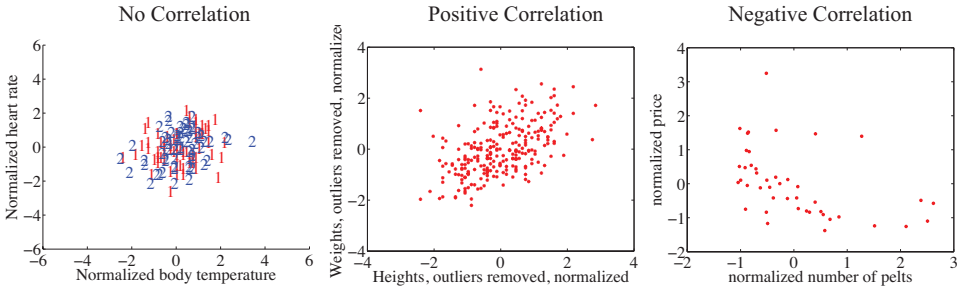

FIGURE 3.16: *The three kinds of scatter plot are less clean for real data than for our idealized examples. Here I used the body temperature vs heart rate data for the zero correlation; the height-weight data for positive correlation; and the lynx data for negative correlation. The pictures aren't idealized — real data tends to be messy — but you can still see the basic structures.*

Positive correlation occurs when larger $\hat{x}$ values tend to appear with larger $\hat{y}$ values. This means that data points with with small (i.e. negative with large magnitude) $\hat{x}$ values must have small $\hat{y}$ values, otherwise the mean of $\hat{x}$ (resp. $\hat{y}$) would be too big. In turn, this means that the scatter plot should look like a "smear" of data from the bottom left of the graph to the top right. The smear might be broad or narrow, depending on some details we'll discuss below. Figure 3.12 shows normalized scatter plots of weight against height, and of body temperature against heart rate. In the weight-height plot, you can clearly see that individuals who are higher tend to weigh more. The important word here is "tend" — taller people could be lighter, but mostly they tend not to be. Notice, also, that I did NOT say that they weighed more *because* they were taller, but only that they tend to be heavier.

Negative correlation occurs when larger $\hat{x}$ values tend to appear with smaller $\hat{y}$ values. This means that data points with with small $\hat{x}$ values must have large $\hat{y}$ values, otherwise the mean of $\hat{x}$ (resp. $\hat{y}$) would be too big. In turn, this means that the scatter plot should look like a "smear" of data from the top left of the graph to the bottom right. The smear might be broad or narrow, depending on some details we'll discuss below. Figure 3.14 shows a normalized scatter plot of the lynx pelt-price data, where I have excluded the data from 1901 on. I did so because there seemed to be some other effect operating to drive prices up, which was inconsistent with the rest of the series. This plot suggests that when there were more pelts, prices were lower, as one would expect.

Zero correlation occurs when there is no relationship. This produces a characteristic shape in a scatter plot, but it takes a moment to understand why. If there really is no relationship, then knowing $\hat{x}$ will tell you nothing about $\hat{y}$. All we know is that $\mathsf{mean}\left(\{\hat{y}\}\right) = 0$, and $\mathsf{var}\left(\{\hat{y}\}\right) = 1$. This is enough information to predict what the plot will look like. We know that $\mathsf{mean}\left(\{\hat{x}\}\right) = 0$ and $\mathsf{var}\left(\{\hat{x}\}\right) = 1$; so there will be many data points with $\hat{x}$ value close to zero, and few with a much larger or much smaller $\hat{x}$ value. The same applies to $\hat{y}$. Now consider the data points in a strip of $\hat{x}$ values. If this strip is far away from the origin, there will

be few data points in the strip, because there aren't many big $\hat{x}$ values. If there is no relationship, we don't expect to see large or small $\hat{y}$ values in this strip, because there are few data points in the strip and because large or small $\hat{y}$ values are uncommon — we see them only if there are many data points, and then seldom. So for a strip with $\hat{x}$ close to zero, we might see some $\hat{y}$ values that are far from zero because we will see many $\hat{y}$ values. For a strip with $\hat{x}$ that is far from zero, we expect to see few $\hat{y}$ values that are far from zero, because we see few points in this strip. This reasoning means the data should form a round blob, centered at the origin. In the temperature-heart rate plot of figure 3.13, it looks as though nothing of much significance is happening. The average heart rate seems to be about the same for people who run warm or who run cool. There is probably not much relationship here.

The correlation is not affected by which variable is plotted on the x-axis and which is plotted on the y-axis. Figure 3.15 compares a plot of height against weight to one of weight against height. Usually, one just does this by rotating the page, or by imagining the new picture. The left plot tells you that data points with higher height value tend to have higher weight value; the right plot tells you that data points with higher weight value tend to have higher height value — i.e. the plots tell you the same thing. It doesn't really matter which one you look at. Again, the important word is "tend" — the plot doesn't tell you anything about *why*, it just tells you that when one variable is larger the other tends to be, too.

3.2.1 The Correlation Coefficient

Consider a normalized data set of N two-dimensional vectors. We can write the i'th data point *in standard coordinates* $(\hat{x}_i, \hat{y}_i)$. We already know many important summaries of this data, because it is in standard coordinates. We have $\mathsf{mean}\left(\{\hat{x}\}\right) = 0$; $\mathsf{mean}\left(\{\hat{y}\}\right) = 0$; $\mathsf{std}\left(\{\hat{x}\}\right) = 1$; and $\mathsf{std}\left(\{\hat{y}\}\right) = 1$. Each of these summaries is itself the mean of some monomial. So $\mathsf{std}\left(\{\hat{x}\}\right)^2 = \mathsf{mean}\left(\left\{\hat{x}^2\right\}\right) = 1$; $\mathsf{std}\left(\{\hat{y}\}\right)^2 = \mathsf{mean}\left(\left\{\hat{y}^2\right\}\right)$ (the other two are easy). We can rewrite this information in terms of means of monomials, giving $\mathsf{mean}\left(\{\hat{x}\}\right) = 0$; $\mathsf{mean}\left(\{\hat{y}\}\right) = 0$; $\mathsf{mean}\left(\left\{\hat{x}^2\right\}\right) = 1$; and $\mathsf{mean}\left(\left\{\hat{y}^2\right\}\right) = 1$. There is one monomial missing here, which is $\hat{x}\hat{y}$.

The term $\mathsf{mean}\left(\{\hat{x}\hat{y}\}\right)$ captures correlation between x and y. The term is known as the **correlation coefficient** or **correlation**.

Definition: 3.1 *Correlation coefficient*

Assume we have N data items which are 2-vectors $(x_1, y_1), \ldots, (x_N, y_N)$, where $N > 1$. These could be obtained, for example, by extracting components from larger vectors. We compute the correlation coefficient by first normalizing the x and y coordinates to obtain $\hat{x}_i = \frac{(x_i - \mathsf{mean}(\{x\}))}{\mathsf{std}(x)}$, $\hat{y}_i = \frac{(y_i - \mathsf{mean}(\{y\}))}{\mathsf{std}(y)}$. The correlation coefficient is the mean value of $\hat{x}\hat{y}$, and can be computed as:

$$\mathsf{corr}\left(\{(x, y)\}\right) = \frac{\sum_i \hat{x}_i \hat{y}_i}{N}$$

Correlation is a measure of our ability to predict one value from another. The correlation coefficient takes values between -1 and 1 (we'll prove this below). If the correlation coefficient is close to 1, then we are likely to predict very well. Small correlation coefficients (under about 0.5, say, but this rather depends on what you are trying to achieve) tend not to be all that interesting, because (as we shall see) they result in rather poor predictions. Figure 3.17 gives a set of scatter plots of different real data sets with different correlation coefficients. These all come from data set of age-height-weight, which you can find at http://www2.stetson.edu/~jrasp/data.htm (look for bodyfat.xls). In each case, two outliers have been removed. Age and height are hardly correlated, as you can see from the figure. Younger people do tend to be slightly taller, and so the correlation coefficient is -0.25. You should interpret this as a small correlation. However, the variable called "adiposity" (which isn't defined, but is presumably some measure of the amount of fatty tissue) is quite strongly correlated with weight, with a correlation coefficient is 0.86. Average tissue density is quite strongly negatively correlated with adiposity, because muscle is much denser than fat, so these variables are negatively correlated — we expect high density to appear with low adiposity, and vice versa. The correlation coefficient is -0.86. Finally, density is very strongly correlated with body weight. The correlation coefficient is -0.98.

It's not always convenient or a good idea to produce scatter plots in standard coordinates (among other things, doing so hides the units of the data, which can be a nuisance). Fortunately, scaling or translating data does not change the value of the correlation coefficient (though it can change the sign if one scale is negative). This means that it's worth being able to spot correlation in a scatter plot that isn't in standard coordinates (even though correlation is always *defined* in standard coordinates). Figure 3.18 shows different correlated datasets plotted in their original units. These data sets are the same as those used in figure 3.17

Properties of the Correlation Coefficient

You should memorize the following properties of the correlation coefficient:

- The correlation coefficient is symmetric (it doesn't depend on the order of its

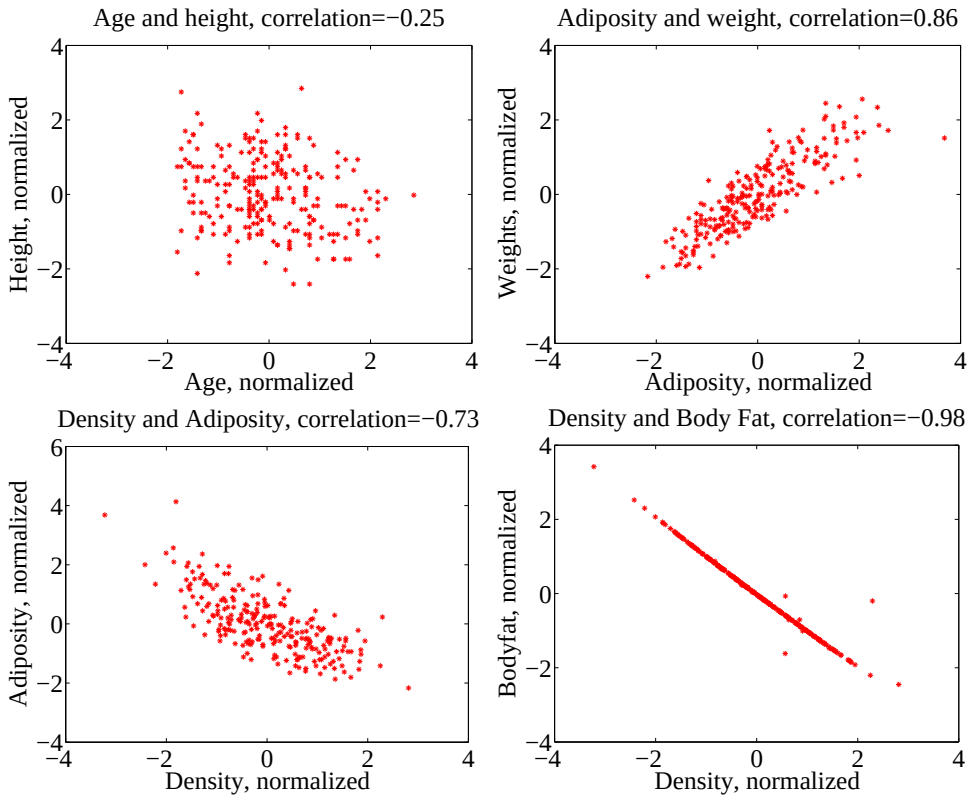

FIGURE 3.17: *Scatter plots for various pairs of variables for the age-height-weight dataset from* http://www2.stetson.edu/~jrasp/data.htm*; bodyfat.xls. In each case, two outliers have been removed, and the plots are in standard coordinates (compare to figure 3.18, which shows these data sets plotted in their original units). The legend names the variables.*

arguments), so

$$\mathsf{corr}\left(\{(x,y)\}\right) = \mathsf{corr}\left(\{(y,x)\}\right)$$

- The value of the correlation coefficient is not changed by translating the data. Scaling the data can change the sign, but not the absolute value. For constants $a \neq 0$, b, $c \neq 0$, d we have

$$\mathsf{corr}\left(\{(ax+b,cx+d)\}\right) = \text{sign}(ab)\mathsf{corr}\left(\{(x,y)\}\right)$$

- If $\hat{y}$ tends to be large (resp. small) for large (resp. small) values of $\hat{x}$, then the correlation coefficient will be positive.

- If $\hat{y}$ tends to be small (resp. large) for large (resp. small) values of $\hat{x}$, then the correlation coefficient will be negative.

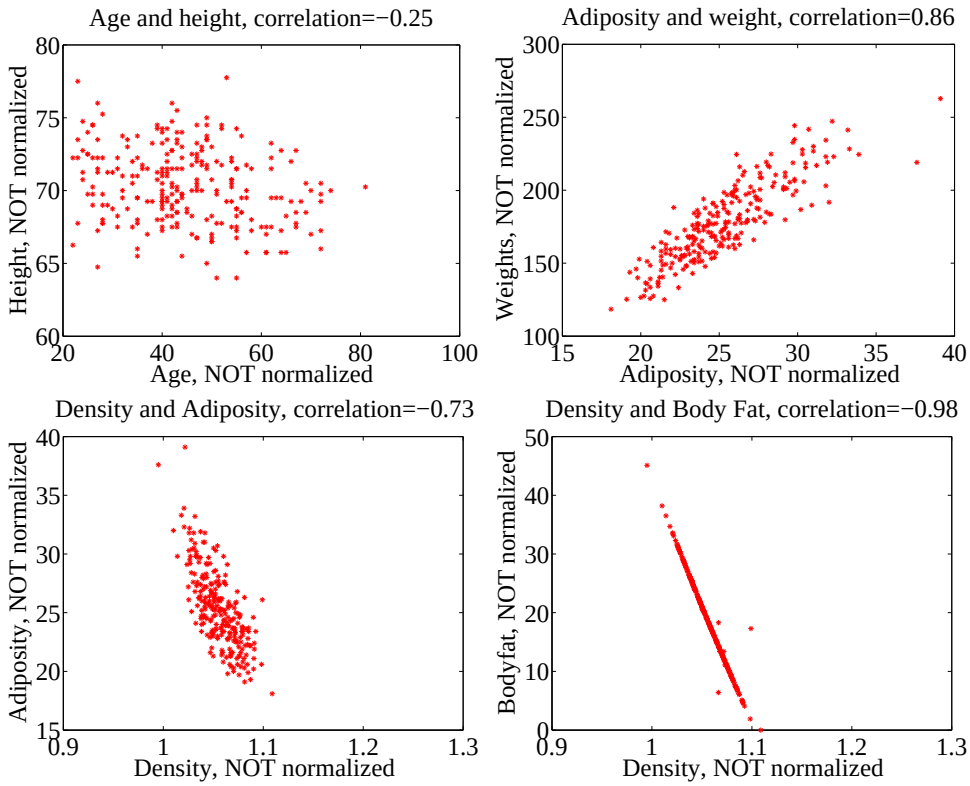

FIGURE 3.18: *Scatter plots for various pairs of variables for the age-height-weight dataset from* http://www2.stetson.edu/~jrasp/data.htm*; bodyfat.xls. In each case, two outliers have been removed, and the plots are* NOT *in standard coordinates (compare to figure 3.17, which shows these data sets plotted in normalized coordinates). The legend names the variables.*

- If $\hat{y}$ doesn't depend on $\hat{x}$, then the correlation coefficient is zero (or close to zero).
- The largest possible value is 1, which happens when $\hat{x} = \hat{y}$.
- The smallest possible value is -1, which happens when $\hat{x} = -\hat{y}$.

The first property is easy, and we relegate that to the exercises. One way to see that the correlation coefficient isn't changed by translation or scale is to notice that it is defined in standard coordinates, and scaling or translating data doesn't change those. Another way to see this is to scale and translate data, then write out the equations; notice that taking standard coordinates removes the effects of the scale and translation. In each case, notice that if the scale is negative, the sign of the correlation coefficient changes.

The property that, if $\hat{y}$ tends to be large (resp. small) for large (resp. small) values of $\hat{x}$, then the correlation coefficient will be positive, doesn't really admit

a formal statement. But it's relatively straightforward to see what's going on. Because $\mathsf{mean}\,(\{\hat{x}\}) = 0$, small values of $\mathsf{mean}\,(\{\hat{x}\})$ must be negative and large values must be positive. But $\mathsf{corr}\,(\{(x,y)\}) = \frac{\sum_i \hat{x}_i \hat{y}_i}{N}$; and for this sum to be positive, it should contain mostly positive terms. It can contain few or no hugely positive (or hugely negative) terms, because $\mathsf{std}\,(\hat{x}) = \mathsf{std}\,(\hat{y}) = 1$ so there aren't many large (or small) numbers. For the sum to contain mostly positive terms, then the sign of $\hat{x}_i$ should be the same as the sign $\hat{y}_i$ for most data items. Small changes to this argument work to show that if if $\hat{y}$ tends to be small (resp. large) for large (resp. small) values of $\hat{x}$, then the correlation coefficient will be negative.

Showing that no relationship means zero correlation requires slightly more work. Divide the scatter plot of the dataset up into thin vertical strips. There are S strips. Each strip is narrow, so the $\hat{x}$ value does not change much for the data points in a particular strip. For the s'th strip, write $N(s)$ for the number of data points in the strip, $\hat{x}(s)$ for the $\hat{x}$ value at the center of the strip, and $\overline{\hat{y}}(s)$ for the mean of the $\hat{y}$ values within that strip. Now the strips are narrow, so we can approximate all data points within a strip as having the same value of $\hat{x}$. This yields

$$\mathsf{mean}\,(\{\hat{x}\hat{y}\}) \approx \frac{1}{S} \sum_{s \in \text{strips}} \hat{x}(s) \left[N(s) \overline{\hat{y}}(s) \right]$$

(where you could replace $\approx$ with $=$ if the strips were narrow enough). Now assume that $\hat{y}(s)$ does not change from strip to strip, meaning that there is no relationship between $\hat{x}$ and $\hat{y}$ in this dataset (so the picture is like the left hand side in figure 3.16). Then each value of $\overline{\hat{y}}(s)$ is the same — we write $\overline{\hat{y}}$ — and we can rearrange to get

$$\mathsf{mean}\,(\{\hat{x}\hat{y}\}) \approx \overline{\hat{y}} \frac{1}{S} \sum_{s \in \text{strips}} \hat{x}(s).$$

Now notice that

$$0 = \mathsf{mean}\,(\{\hat{y}\}) \approx \frac{1}{S} \sum_{s \in \text{strips}} N(s) \overline{\hat{y}}(s)$$

(where again you could replace $\approx$ with $=$ if the strips were narrow enough). This means that if every strip has the same value of $\overline{\hat{y}}(s)$, then that value must be zero. In turn, if there is no relationship between $\hat{x}$ and $\hat{y}$, we must have $\mathsf{mean}\,(\{\hat{x}\hat{y}\}) = 0$.

Property 3.1: The largest possible value of the correlation is 1, and this occurs when $\hat{x}_i = \hat{y}_i$ for all i. The smallest possible value of the correlation is -1, and this occurs when $\hat{x}_i = -\hat{y}_i$ for all i.

Proposition:

$$-1 \leq \mathsf{corr}\left(\{(x, y)\}\right) \leq 1$$

Proof: Writing $\hat{x}$, $\hat{y}$ for the normalized coefficients, we have

$$\mathsf{corr}\left(\{(x, y)\}\right) = \frac{\sum_i \hat{x}_i \hat{y}_i}{N}$$

and you can think of the value as the inner product of two vectors. We write

$$\mathbf{x} = \frac{1}{\sqrt{N}}\left[\hat{x}_1, \hat{x}_2, \ldots \hat{x}_N\right] \text{ and } \mathbf{y} = \frac{1}{\sqrt{N}}\left[\hat{y}_1, \hat{y}_2, \ldots \hat{y}_N\right]$$

and we have $\mathsf{corr}\left(\{(x, y)\}\right) = \mathbf{x}^T\mathbf{y}$. Notice $\mathbf{x}^T\mathbf{x} = \mathsf{std}\left(x\right)^2 = 1$, and similarly for $\mathbf{y}$. But the inner product of two vectors is at its maximum when the two vectors are the same, and this maximum is 1. This argument is also sufficient to show that smallest possible value of the correlation is -1, and this occurs when $\hat{x}_i = -\hat{y}_i$ for all i.

3.2.2 Using Correlation to Predict

Assume we have N data items which are 2-vectors $(x_1, y_1), \ldots, (x_N, y_N)$, where $N > 1$. These could be obtained, for example, by extracting components from larger vectors. As usual, we will write $\hat{x}_i$ for x_i in normalized coordinates, and so on. Now assume that we know the correlation coefficient is r (this is an important, traditional notation). What does this mean?

One (very useful) interpretation is in terms of prediction. Assume we have a data point $(x_0, ?)$ where we know the x-coordinate, but not the y-coordinate. We can use the correlation coefficient to predict the y-coordinate. First, we transform to standard coordinates. Now we must obtain the best $\hat{y}_0$ value to predict, using the $\hat{x}_0$ value we have.

We want to construct a prediction function which gives a prediction for any value of $\hat{x}$. This predictor should behave as well as possible on our existing data. For each of the $(\hat{x}_i, \hat{y}_i)$ pairs in our data set, the predictor should take $\hat{x}_i$ and produce a result as close to $\hat{y}_i$ as possible. We can choose the predictor by looking at the errors it makes at each data point.

We write $\hat{y}_i^p$ for the value of $\hat{y}_i$ predicted at $\hat{x}_i$. The simplest form of predictor is linear. If we predict using a linear function, then we have, for some unknown a, b, that $\hat{y}_i^p = a\hat{x}_i + b$. Now think about $u_i = \hat{y}_i - \hat{y}_i^p$, which is the error in our prediction. We would like to have $\mathsf{mean}\left(\{u\}\right) = 0$ (otherwise, we could reduce the

error of the prediction just by subtracting a constant).

$$\begin{aligned}
\mathsf{mean}\left(\{u\}\right) &= \mathsf{mean}\left(\{\hat{y} - \hat{y}^p\}\right) \\
&= \mathsf{mean}\left(\{\hat{y}\}\right) - \mathsf{mean}\left(\{a\hat{x}_i + b\}\right) \\
&= \mathsf{mean}\left(\{\hat{y}\}\right) - a\mathsf{mean}\left(\{\hat{x}\}\right) + b \\
&= 0 - a0 + b \\
&= 0.
\end{aligned}$$

This means that we must have $b = 0$.

To estimate a, we need to think about $\mathsf{var}\left(\{u\}\right)$. We should like $\mathsf{var}\left(\{u\}\right)$ to be as small as possible, so that the errors are as close to zero as possible (remember, small variance means small standard deviation which means the data is close to the mean). We have

$$\begin{aligned}
\mathsf{var}\left(\{u\}\right) &= \mathsf{var}\left(\{\hat{y} - \hat{y}^p\}\right) \\
&= \mathsf{mean}\left(\left\{(\hat{y} - a\hat{x})^2\right\}\right) \quad \text{because } \mathsf{mean}\left(\{u\}\right) = 0 \\
&= \mathsf{mean}\left(\left\{(\hat{y})^2 - 2a\hat{x}\hat{y} + a^2(\hat{x})^2\right\}\right) \\
&= \mathsf{mean}\left(\left\{(\hat{y})^2\right\}\right) - 2a\mathsf{mean}\left(\{\hat{x}\hat{y}\}\right) + a^2\mathsf{mean}\left(\left\{(\hat{x})^2\right\}\right) \\
&= 1 - 2ar + a^2,
\end{aligned}$$

which we want to minimize by choice of a. At the minimum, we must have

$$\frac{d\mathsf{var}\left(\{u_i\}\right)}{da} = 0 = -2r + 2a$$

so that $a = r$ and the correct prediction is

$$\hat{y}_0^p = r\hat{x}_0$$

You can use a version of this argument to establish that if we have $(?, \hat{y}_0)$, then the best prediction for $\hat{x}_0$ (*which is in standard coordinates*) is $r\hat{y}_0$. It is important to notice that the coefficient of $\hat{y}_i$ is NOT $1/r$; you should work this example, which appears in the exercises. We now have a prediction procedure, outlined below.

Procedure: 3.1 *Predicting a value using correlation*

Assume we have N data items which are 2-vectors $(x_1, y_1), \ldots, (x_N, y_N)$, where $N > 1$. These could be obtained, for example, by extracting components from larger vectors. Assume we have an x value x_0 for which we want to give the best prediction of a y value, based on this data. The following procedure will produce a prediction:

- Transform the data set into standard coordinates, to get

$$\begin{aligned}\hat{x}_i &= \frac{1}{\mathsf{std}\,(x)}(x_i - \mathsf{mean}\,(\{x\})) \\ \hat{y}_i &= \frac{1}{\mathsf{std}\,(y)}(y_i - \mathsf{mean}\,(\{y\})) \\ \hat{x}_0 &= \frac{1}{\mathsf{std}\,(x)}(x_0 - \mathsf{mean}\,(\{x\})).\end{aligned}$$

- Compute the correlation

$$r = \mathsf{corr}\,(\{(x, y)\}) = \mathsf{mean}\,(\{\hat{x}\hat{y}\}).$$

- Predict $\hat{y}_0 = r\hat{x}_0$.
- Transform this prediction into the original coordinate system, to get

$$y_0 = \mathsf{std}\,(y) r \hat{x}_0 + \mathsf{mean}\,(\{y\})$$

Now assume we have a y value y_0, for which we want to give the best prediction of an x value, based on this data. The following procedure will produce a prediction:

- Transform the data set into standard coordinates.
- Compute the correlation.
- Predict $\hat{x}_0 = r\hat{y}_0$.
- Transform this prediction into the original coordinate system, to get

$$x_0 = \mathsf{std}\,(x) r \hat{y}_0 + \mathsf{mean}\,(\{x\})$$

There is another way of thinking about this prediction procedure, which is often helpful. Assume we need to predict a value for x_0. In normalized coordinates, our prediction is $\hat{y}^p = r\hat{x}_0$; if we revert back to the original coordinate system, the

prediction becomes

$$\frac{(y^p - \mathsf{mean}\left(\{y\}\right))}{\mathsf{std}\left(y\right)} = r\left(\frac{(x_0 - \mathsf{mean}\left(\{x\}\right))}{\mathsf{std}\left(x\right)}\right).$$

This gives a really useful rule of thumb, which I have broken out in the box below.

Procedure: 3.2 *Predicting a value using correlation: Rule of thumb - 1*

If x_0 is k standard deviations from the mean of x, then the predicted value of y will be rk standard deviations away from the mean of y, and the sign of r tells whether y increases or decreases.

An even more compact version of the rule of thumb is in the following box.

Procedure: 3.3 *Predicting a value using correlation: Rule of thumb - 2*

The predicted value of y goes up by r standard deviations when the value of x goes up by one standard deviation.

We can compute the average root mean square error that this prediction procedure will make. The square of this error must be

$$\begin{aligned}\mathsf{mean}\left(\left\{u^2\right\}\right) &= \mathsf{mean}\left(\left\{y^2\right\}\right) - 2r\mathsf{mean}\left(\{xy\}\right) + r^2\mathsf{mean}\left(\left\{x^2\right\}\right) \\ &= 1 - 2r^2 + r^2 \\ &= 1 - r^2\end{aligned}$$

so the root mean square error will be $\sqrt{1-r^2}$. This is yet another interpretation of correlation; if x and y have correlation close to one, then predictions could have very small root mean square error, and so might be very accurate. In this case, knowing one variable is about as good as knowing the other. If they have correlation close to zero, then the root mean square error in a prediction might be as large as the root mean square error in $\hat{y}$ — which means the prediction is nearly a pure guess.

The prediction argument means that we can spot correlations for data in other kinds of plots — one doesn't have to make a scatter plot. For example, if we were to observe a child's height from birth to their 10'th year (you can often find these observations in ballpen strokes, on kitchen walls), we could plot height as a function of year. If we also had their weight (less easily found), we could plot weight as a function of year, too. The prediction argument above say that, if you can predict the weight from the height (or vice versa) then they're correlated. One way to spot this is to look and see if one curve goes up when the other does (or goes down when the other goes up). You can see this effect in figure 3.5, where (before 19h00), prices go down when the number of pelts goes up, and vice versa. These two variables are negatively correlated.

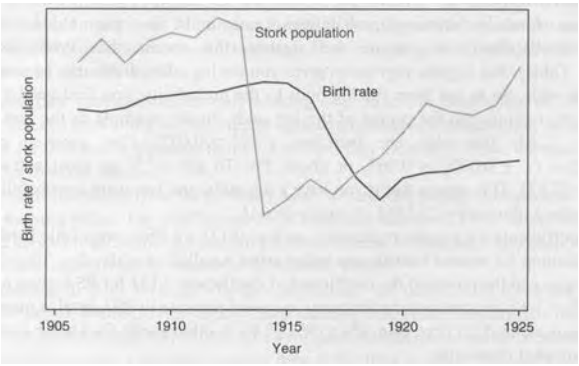

FIGURE 3.19: *This figure, from Vickers (ibid, p184) shows a plot of the stork population as a function of time, and the human birth rate as a function of time, for some years in Germany. The correlation is fairly clear; but this does not mean that reducing the number of storks means there are fewer able to bring babies. Instead, this is the impact of the first world war — a hidden or latent variable.*

3.2.3 Confusion caused by correlation

There is one very rich source of potential (often hilarious) mistakes in correlation. When two variables are correlated, they change together. If the correlation is positive, that means that, in typical data, if one is large then the other is large, and if one is small the other is small. In turn, this means that one can make a reasonable prediction of one from the other. However, correlation DOES NOT mean that changing one variable causes the other to change (sometimes known as causation).

Two variables in a dataset could be correlated for a variety of reasons. One important reason is pure accident. If you look at enough pairs of variables, you may well find a pair that appears to be correlated just because you have a small set of observations. Imagine, for example, you have a dataset consisting of only two vectors — there is a pretty good chance that there is some correlation between the coefficients. Such accidents can occur in large datasets, particularly if the dimensions are high.

Another reason variables could be correlated is that there is some causal relationship — for example, pressing the accelerator tends to make the car go faster, and so there will be some correlation between accelerator position and car acceleration. As another example, adding fertilizer does tend to make a plant grow bigger. Imagine you record the amount of fertilizer you add to each pot, and the size of the resulting potplant. There should be some correlation.

Yet another reason variables could be correlated is that there is some other background variable — often called a **latent variable** — linked causally to each of the observed variables. For example, in children (as Freedman, Pisani and Purves note in their excellent *Statistics*), shoe size is correlated with reading skills. This DOES NOT mean that making your feet grow will make you read faster, or that you can make your feet shrink by forgetting how to read. The real issue here is

the age of the child. Young children tend to have small feet, and tend to have weaker reading skills (because they've had less practice). Older children tend to have larger feet, and tend to have stronger reading skills (because they've had more practice). You can make a reasonable prediction of reading skills from foot size, because they're correlated, even though there is no direct connection.

This kind of effect can mask correlations, too. Imagine you want to study the effect of fertilizer on potplants. You collect a set of pots, put one plant in each, and add different amounts of fertilizer. After some time, you record the size of each plant. You expect to see correlation between fertilizer amount and plant size. But you might not if you had used a different species of plant in each pot. Different species of plant can react quite differently to the same fertilizer (some plants just die if over-fertilized), so the species could act as a latent variable. With an unlucky choice of the different species, you might even conclude that there was a negative correlation between fertilizer and plant size. This example illustrates why you need to take great care in setting up experiments and interpreting their results.

This sort of thing happens often, and it's an effect you should look for. Another nice example comes from Vickers (ibid). The graph, shown in Figure 3.19, shows a plot of (a) a dataset of the stork population in Europe over a period of years and (b) a dataset of the birth rate over those years. This isn't a scatter plot; instead, the data has been plotted on a graph. You can see by eye that these two datasets are quite strongly correlated . Even more disturbing, the stork population dropped somewhat before the birth rate dropped. Is this evidence that storks brought babies in Europe during those years? No (the usual arrangement seems to have applied). For a more sensible explanation, look at the dates. The war disturbed both stork and human breeding arrangements. Storks were disturbed immediately by bombs, etc., and the human birth rate dropped because men died at the front.

3.3 STERILE MALES IN WILD HORSE HERDS

Large herds of wild horses are (apparently) a nuisance, but keeping down numbers by simply shooting surplus animals would provoke outrage. One strategy that has been adopted is to sterilize males in the herd; if a herd contains sufficient sterile males, fewer foals should result. But catching stallions, sterilizing them, and reinserting them into a herd is a performance — does this strategy work?

We can get some insight by plotting data. At http://lib.stat.cmu.edu/DASL/Datafiles/WildHorses.html, you can find a dataset covering herd management in wild horses. I have plotted part of this dataset in figure 3.20. In this dataset, there are counts of all horses, sterile males, and foals made on each of a small number of days in 1986, 1987, and 1988 for each of two herds. I extracted data for one herd. I have plotted this data as a function of the count of days since the first data point, because this makes it clear that some measurements were taken at about the same time, but there are big gaps in the measurements. In this plot, the data points are shown with a marker. Joining them leads to a confusing plot because the data points vary quite strongly. However, notice that the size of the herd drifts down slowly (you could hold a ruler against the plot to see the trend), as does the number of foals, when there is a (roughly) constant number of sterile males.

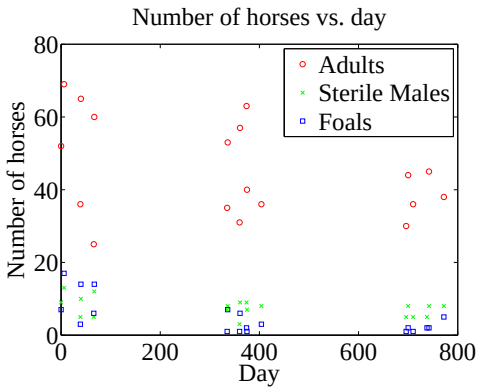

FIGURE 3.20: *A plot of the number of adult horses, sterile males, and foals in horse herds over a period of three years. The plot suggests that introducing sterile males might cause the number of foals to go down. Data from* http://lib.stat.cmu.edu/DASL/Datafiles/WildHorses.html.

Does sterilizing males result in fewer foals? This is likely hard to answer for this dataset, but we could ask whether herds with more sterile males have fewer foals. A scatter plot is a natural tool to attack this question. However, the scatter plots of figure 3.21 suggest, rather surprisingly, that when there are more sterile males there are more adults (and vice versa), and when there are more sterile males there are more foals (and vice versa). This is borne out by a correlation analysis. The correlation coefficient between foals and sterile males is 0.74, and the correlation coefficient between adults and sterile males is 0.68. You should find this very surprising — how do the horses know how many sterile males there are in the herd? You might think that this is an effect of scaling the plot, but there is a scatter plot in normalized coordinates in figure 3.21 that is entirely consistent with the conclusions suggested by the unnormalized plot. What is going on here?

The answer is revealed by the scatter plots of figure 3.22. Here, rather than plotting a '*' at each data point, I have plotted the day number of the observation. This is in days from the first observation. You can see that the whole herd is shrinking — observations where there are many adults (resp. sterile adults, foals) occur with small day numbers, and observations where there are few have large day numbers. Because the whole herd is shrinking, it is true that when there are more adults and more sterile males, there are also more foals. Alternatively, you can see the plots of figure 3.20 as a scatter plot of herd size (resp. number of foals, number of sterile males) against day number. Then it becomes clear that the whole herd is shrinking, as is the size of each group. To drive this point home, we can look at the correlation coefficient between adults and days (-0.24), between sterile adults and days (-0.37), and between foals and days (-0.61). We can use the rule of thumb in box 3.3 to interpret this. This means that every 282 days, the herd loses about three adults; about one sterile adult; and about three foals. For the herd to have a stable size, it needs to gain by birth as many foals as it loses both to growing up and to death. If the herd is losing three foals every 282 days, then if they all grow

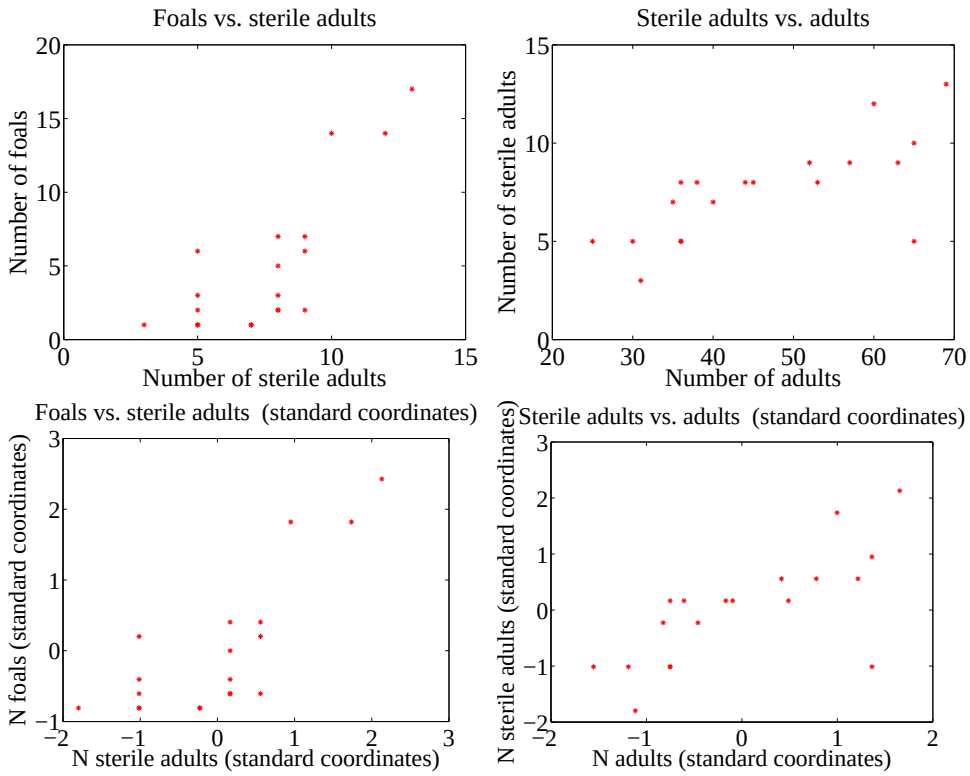

FIGURE 3.21: *Scatter plots of the number of sterile males in a horse herd against the number of adults, and the number of foals against the number of sterile males, from data of http://lib.stat.cmu.edu/DASL/Datafiles/WildHorses.html.* **Top:** *unnormalized;* **bottom:** *standard coordinates.*

up to replace the missing adults, the herd will be shrinking slightly (because it is losing four adults in this time); but if it loses foals to natural accidents, etc., then it is shrinking rather fast.

The message of this example is important. To understand a simple dataset, you might need to plot it several ways. You should make a plot, look at it and ask what it says, and then try to use another type of plot to confirm or refute what you think might be going on.

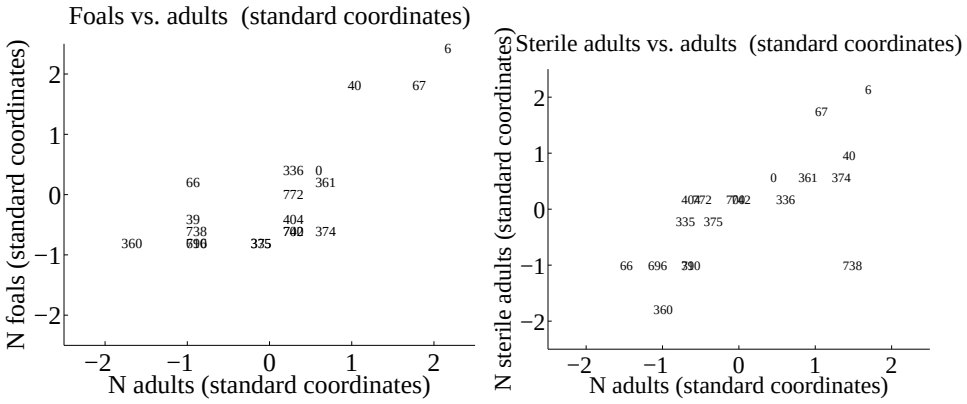

FIGURE 3.22: *Scatter plots of the number of foals vs. the number of adults and the number of adults vs. the number of sterile adults for the wild horse herd, from* http://lib.stat.cmu.edu/DASL/Datafiles/WildHorses.html*. Rather than plot data points as dots, I have plotted the* day *on which the observation was made. Notice how the herd starts large, and then shrinks.*

3.4 YOU SHOULD

3.4.1 remember these definitions:

3.4.2 remember these terms:

3.4.3 remember these facts:

3.4.4 remember these procedures:

3.4.5 be able to:

- Plot a bar chart, a heat map, and a pie chart for a categorical dataset.
- Plot a dataset as a graph, making sensible choices about markers, lines and the like.
- Plot a scatter plot for a dataset.
- Plot a normalized scatter plot for a dataset.
- Interpret the scatter plot to tell the sign of the correlation between two variables, and estimate the size of the correlation coefficient.
- Compute a correlation coefficient.
- Interpret a correlation coefficient.
- Use correlation to make predictions.

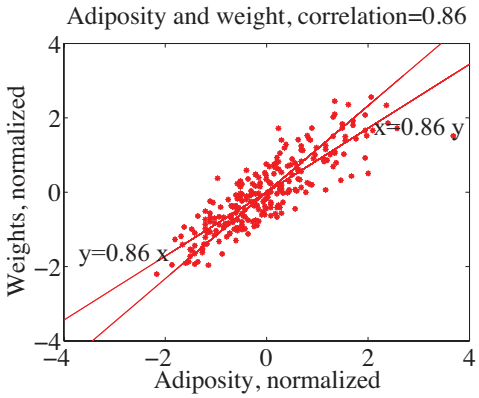

FIGURE 3.23: *This figure shows two lines, $y = 0.86x$ and $x = 0.86y$, superimposed on the normalized adiposity-weight scatter plot.*

PROBLEMS

3.1. In a population, the correlation coefficient between weight and adiposity is 0.9. The mean weight is 150lb. The standard deviation in weight is 30lb. Adiposity is measured on a scale such that the mean is 0.8, and the standard deviation is 0.1.
- **(a)** Using this information, predict the expected adiposity of a subject whose weight is 170lb
- **(b)** Using this information, predict the expected weight of a subject whose adiposity is 0.75
- **(c)** How reliable do you expect this prediction to be? Why? (your answer should be a property of correlation, not an opinion about adiposity or weight)

3.2. n a population, the correlation coefficient between family income and child IQ is 0.30. The mean family income was \$60, 000. The standard deviation in income is \$20, 000. IQ is measured on a scale such that the mean is 100, and the standard deviation is 15.
- **(a)** Using this information, predict the expected IQ of a child whose family income is \$70, 000
- **(b)** How reliable do you expect this prediction to be? Why? (your answer should be a property of correlation, not an opinion about IQ)
- **(c)** The family income now rises does the correlation predict that the child will have a higher IQ? Why?

3.3. Show that $\mathsf{corr}(\{(x, y)\}) = \mathsf{corr}(\{(y, x)\})$ by substituting into the definition.

3.4. Show that if $\hat{y}$ tends to be small (resp. large) for large (resp. small) values of $\hat{x}$, then the correlation coefficient will be negative.

3.5. We have a 2D dataset consisting of N pairs $(\hat{x}_i, \hat{y}_i)$ in normalized coordinates. This data has correlation coefficient r. We observe a new $\hat{y}$ value $\hat{y}_0$, and wish to predict the (unknown) x value. We will do so with a linear prediction, choosing a, b, to predict an $\hat{x}$ for any $\hat{y}$ using the rule $\hat{x}^p = a\hat{y}^p + b$. Write $u_i = \hat{x}_i - \hat{x}_i^p$ for the error that this rule makes on each data item.
- **(a)** We require $\mathsf{mean}(\{u\}) = 0$. Show that this means that $b = 0$.
- **(b)** We require that $\mathsf{var}(\{u\})$ is minimized. Show that this means that $a = r$.

(c) We now have a result that seems paradoxical — if I have $(\hat{x}_0, ?)$ I predict $(\hat{x}_0, r\hat{x}_0)$ and if I have $(?, y_0)$, I predict $(r\hat{y}_0, \hat{y}_0)$. Use figure 3.23 to explain why this is right. The important difference between the two lines is that lies (approximately) in the middle of each vertical span of data, and the other lies (approximately) in the middle of each horizontal span of data.

3.6. I did the programming exercise about the earth temperature below. I looked at the years 1965-2012. Write $\{(y, T)\}$ for the dataset giving the temperature (T) of the earth in year y. I computed: $\mathsf{mean}(\{y\}) = 1988.5$, $\mathsf{std}(y) = 14$, $\mathsf{mean}(\{T\}) = 0.175$, $\mathsf{std}(T) = 0.231$ and $\mathsf{corr}(\{y\})T = 0.892$. What is the best prediction using this information for the temperature in mid 2014? in mid 2028? in mid 2042?

3.7. I did the programming exercise about the earth temperature below. It is straightforward to build a dataset $\{(T, n_t)\}$ where each entry contains the temperature of the earth (T) and the number of counties where FEMA declared tornadoes n_t (for each year, you look up T and n_t, and make a data item). I computed: $\mathsf{mean}(\{T\}) = 0.175$, $\mathsf{std}(T) = 0.231$, $\mathsf{mean}(\{n_t\}) = 31.6$, $\mathsf{std}(n_t) = 30.8$, and $\mathsf{corr}(\{T\})n_t = 0.471$. What is the best prediction using this information for the number of tornadoes if the global earth temperature is 0.5? 0.6? 0.7?

PROGRAMMING EXERCISES

3.8. At http://lib.stat.cmu.edu/DASL/Datafiles/cigcancerdat.html, you will find a dataset recording per capita cigarette sales and cancer deaths per 100K population for a variety of cancers, recorded for 43 states and the District of Columbia in 1960.

(a) Plot a scatter plot of lung cancer deaths against cigarette sales, using the two letter abbreviation for each state as a marker. You should see two fairly obvious outliers. The backstory at http://lib.stat.cmu.edu/DASL/Stories/cigcancer.html suggests that the unusual sales in Nevada are generated by tourism (tourists go home, and die there) and the unusual sales in DC are generated by commuting workers (who also die at home).

(b) What is the correlation coefficient between per capita cigarette sales and lung cancer deaths per 100K population? Compute this with, and without the outliers. What effect did the outliers have? Why?

(c) What is the correlation coefficient between per capita cigarette sales and bladder cancer deaths per 100K population? Compute this with, and without the outliers. What effect did the outliers have? Why?

(d) What is the correlation coefficient between per capita cigarette sales and kidney cancer deaths per 100K population? Compute this with, and without the outliers. What effect did the outliers have? Why?

(e) What is the correlation coefficient between per capita cigarette sales and leukemia deaths per 100K population? Compute this with, and without the outliers. What effect did the outliers have? Why?

(f) You should have computed a positive correlation between cigarette sales and lung cancer deaths. Does this mean that smoking causes lung cancer? Why?

(g) You should have computed a negative correlation between cigarette sales and leukemia deaths. Does this mean that smoking cures leukemia? Why?

3.9. At http://www.cru.uea.ac.uk/cru/info/warming/gtc.csv, you can find a dataset of global temperature by year. When I accessed this, the years spanned 1880-

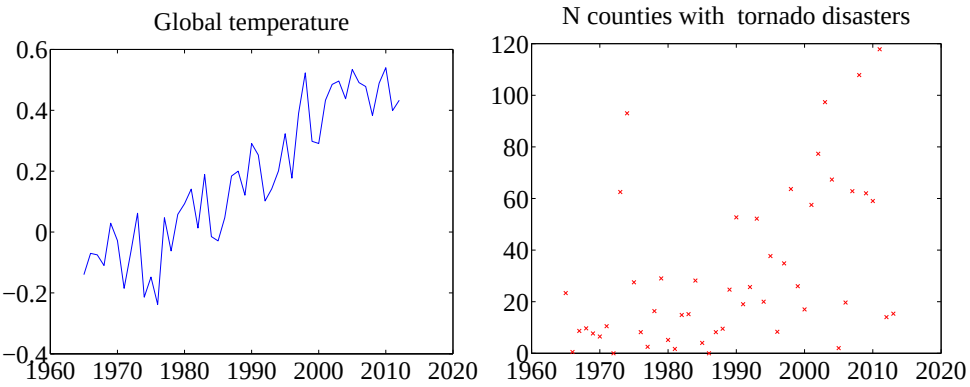

FIGURE 3.24: *Plots I prepared from* **left** *uea data on temperature and* **right** *FEMA data on tornadoes by county. These should help you tell if you're on the right track.*

2012. I don't know what units the temperatures are measured in. Keep in mind that measuring the temperature of the earth has non-trivial difficulties (you can't just insert an enormous thermometer!), and if you look at http://www.cru.uea.ac.uk/cru and http://www.cru.uea.ac.uk/cru/data/temperature/ you can see some discussion of the choices made to get these measurements. There are two kinds of data in this dataset, smoothed and unsmoothed. I used the unsmoothed data, which should be fine for our purposes. The government publishes a great deal of data at http://data.gov. From there, I found a dataset, published by the Federal Emergency Management Agency (FEMA), of all federally declared disasters (which I found at http://www.fema.gov/media-library/assets/documents/28318?id=6292). We would like to see whether weather related disasters are correlated to global temperature.

(a) The first step is preprocessing the data. The FEMA data has all sorts of information. From 1965 on, a disaster was declared per county (so you get one line in the data set for each county), but before 1965, it seems to have been by state. We divide the disasters into four types: TORNADO, FLOOD, STORM, HURRICANE. (FEMA seems to have a much richer type system). We want to know how many counties declare a disaster of each type, in each year. This is a really rough estimate of the number of people affected by the disaster. If a disaster has two types (in some rows, you will see "SEVERE STORMS, HEAVY RAINS & FLOODING" we will allocate the credit evenly between the types (i.e. for this case we would count 1/2 for STORM and 1/2 for FLOOD). You should write code that will (a) read the dataset and then (b) compute a table with the count of the number of counties where a disaster of each type has occurred for each year. This takes a bit of work. Notice you only need to deal with two columns of the data (the date it was declared, and the type). Notice also that FEMA changed the way it represented dates somewhere through the first column (they added times), which can cause problems. You can tell the type of the disaster by just using a string match routine with the four keywords. Figure 3.24 shows the plot of temperature and of number of counties where FEMA declared a tornado disaster for this data.

(b) Plot a normalized scatter plot of the number of counties where FEMA declared the disaster against temperature, for each kind.

(c) For each kind of disaster, compute the correlation coefficient between the number of counties where FEMA declared the disaster and the year. For each kind of disaster, use this correlation coefficient to predict the number of disasters of this kind for 2013. Compare this to the true number, and explain what you see.

(d) For each kind of disaster, compute the correlation coefficient between the number of counties where FEMA declared the disaster and the global temperature. For each kind of disaster, use this correlation coefficient to predict the number of disasters of this kind when the earth reaches 0.6 temperature units and 0.7 temperature units (on the absolute temperature scale).

(e) Does this data show that warming of the earth causes weather disasters? Why?

(f) Does this data suggest that more people will be affected by disasters in the US in the future? Why?

(g) Does this data suggest that the earth will be warmer in the future? Why?

3.10. If you go to https://github.com/TheUpshot/Military-Surplus-Gear, you will find data on purchases of military weapons by US police departments. This data is organized by state and county. There's a fair amount of data here, and you'll need to do some data jockeying.

(a) Prepare a plot showing how much each Illinois county spent under this program.

(b) Now look up population levels in the counties. Prepare a plot showing how much each county spent *per capita*.

(c) Prepare a graphic illustrating what the overall most popular items were — i.e., those items counties bought the most of.

(d) Prepare a graphic illustrating on what items the most money was spent — for example, was more money spent on "RIFLE,5.56 MILLIMETER" or on "MINE RESISTANT VEHICLE"?

(e) Prepare a graphic illustrating the pattern of purchases across counties for the ten overall most popular items.

(f) Can you draw any interesting conclusions?

PART TWO

PROBABILITY

CHAPTER 4

Basic ideas in probability

We will perform experiments — which could be pretty much anything, from flipping a coin, to eating too much saturated fat, to smoking, to crossing the road without looking — and reason about the outcomes (mostly bad for the examples I gave). But these outcomes are uncertain, and we need to weigh those uncertainties against one another. If I flip a coin, I could get heads or tails, and there's no reason to expect to see one more often than the other. If I eat too much saturated fat or smoke, I will very likely have problems, though I might not. If I cross the road without looking, I may be squashed by a truck or I may not. Our methods need also to account for information. If I look before I cross the road, I am much less likely to be squashed. Probability is the machinery we use to describe and account for the fact that some outcomes are more frequent than others.

4.1 EXPERIMENTS, OUTCOMES AND PROBABILITY

Imagine you repeat the same experiment numerous times. You do not necessarily expect to see the same result each time. Some results might occur more frequently than others. We account for this tendency using probability. To do so, we need to be clear about what results an experiment can have. For example, you flip a coin. We might agree that the only possible results are a head or a tail, thus ignoring the possibilities that (say) a bird swoops down and steals the coin; the coin lands and stays on edge; the coin falls between the cracks in the floor and disappears; and so on. By doing so, we have idealized the experiment.

4.1.1 Outcomes and Probability

We will formalize experiments by specifying the set of **outcomes** that we expect from the experiment. Every run of the experiment produces exactly one of the set of possible outcomes. We never see two or more outcomes from a single experiment, and we never see no outcome. The advantage of doing this is that we can count how often each outcome appears.

Definition: 4.1 *Sample space*

The sample space is the set of all outcomes, which we usually write Ω.

Worked example 4.1 *Find the lady*

We have three playing cards. One is a queen; one is a king, and one is a jack. All are shown face down, and one is chosen at random and turned up. What is the set of outcomes?

Solution: Write Q for queen, K for king, J for jack; the outcomes are $\{Q, K, J\}$

Worked example 4.2 *Find the lady, twice*

We play find the lady twice, replacing the card we have chosen. What is the sample space?

Solution: We now have $\{QQ, QK, QN, KQ, KK, KJ, JQ, JK, JJ\}$

Useful Facts: 4.1 *Sample spaces are required, and need not be finite*

You must specify a sample space to build a probability model. Sample spaces do not need to be finite.

We represent our model of how often a particular outcome will occur in a repeated experiment with a **probability**, a non-negative number. This number gives the relative frequency of the outcome of interest, when an experiment is repeated a very large number of times.

Assume that we repeat an experiment N times. Assume also that the coins, dice, whatever involved in each repetition of the experiment don't communicate with one another from experiment to experiment (or, equivalently, that experiments don't "know" about one another). We say that an outcome A has probability P if (a) outcome A occurs in about $N \times P$ of those experiments and (b) as N gets larger, the fraction of experiments where outcome A occurs will get closer to P. We write $\#(A)$ for the number of times outcome A occurs. We interpret P as

$$\lim_{N \to \infty} \frac{\#(A)}{N}.$$

We can draw two important conclusions immediately.

- For any outcome A, $0 \leq P(A) \leq 1$.
- $\sum_{A_i \in \Omega} P(A_i) = 1$.

Remember that every run of the experiment produces exactly one outcome. The probabilities add up to one because each experiment must have one of the outcomes in the sample space. Some problems can be handled by building a set of outcomes and reasoning about the probability of each outcome. This is particularly useful when the outcomes *must* have the same probability, which happens rather a lot.

Worked example 4.3 *A biased coin*

Assume we have a coin where the probability of getting heads is $P(H) = \frac{1}{3}$, and so the probability of getting tails is $P(T) = \frac{2}{3}$. We flip this coin three million times. How many times do we see heads?

Solution: $P(H) = \frac{1}{3}$, so we expect this coin will come up heads in $\frac{1}{3}$ of experiments. This means that we will very likely see very close to a million heads. Later on, we will be able to be more precise.

Remember this: *The probability of an outcome is the frequency of that outcome in a very large number of repeated experiments. The sum of probabilities over all outcomes must be one.*

4.2 EVENTS

Assume we run an experiment and get an outcome. We know what the outcome is (that's the whole point of a sample space). This means we can tell whether the outcome we get belongs to some particular known *set* of outcomes. We just look in the set and see if our outcome is there. This means that we should be able to predict the probability of a *set* of outcomes from any reasonable model of an experiment. For example, we might roll a die and ask what the probability of getting an even number is. An **event** is a set of outcomes. We would like our probability models to be able to predict the probability of events.

Assume A and B are two distinct outcomes, and write $\mathcal{E} = \{A, B\}$ for the event that contains both. We must have that $P(\mathcal{E}) = P(A) + P(B)$, because the number of times repeated experiments produce an outcome in $\mathcal{E}$ is given by the number of times we see A plus the number of times we see B. Now assume that C_i are N distinct outcomes, and $\mathcal{F}$ is the event that contains all of them, and no other outcomes. Then we must have $P(\mathcal{F}) = \sum_i P(C_i)$ (because we observe an outcome in $\mathcal{F}$ whenever we see any of the outcomes C_i).

This means we can compute the probabilities of many events without much fuss. The set of all outcomes, which we wrote Ω, must be an event. We must have $P(\Omega) = 1$ (because we said that every run of an experiment produces one outcome,

and that outcome must be in Ω). In principle, there could be no outcome, although this never happens. This means that the empty set, which we write $\varnothing$, is an event, and we have $P(\varnothing) = 0$.

4.2.1 Computing Event Probabilities by Counting Outcomes

Assume A and B are two distinct outcomes, and write $\mathcal{E} = \{A, B\}$ for the event that contains both. We must have that $P(\mathcal{E}) = P(A) + P(B)$, because the number of times repeated experiments produce an outcome in $\mathcal{E}$ is given by the number of times we see A plus the number of times we see B. Now assume that C_i are N distinct outcomes, and $\mathcal{F}$ is the event that contains all of them, and no other outcomes. Then we must have $P(\mathcal{F}) = \sum_i P(C_i)$ (because we observe an outcome in $\mathcal{F}$ whenever we see any of the outcomes C_i).

The set of all outcomes, which we wrote Ω, must be an event. We must have $P(\Omega) = 1$ (because we said that every run of an experiment produces one outcome, and that outcome must be in Ω). In principle, there could be no outcome, although this never happens. This means that the empty set, which we write $\varnothing$, is an event, and we have $P(\varnothing) = 0$. Many times, computing the probability of an event can be reduced in this way to an advanced counting exercise.

The expression $P(\mathcal{F}) = \sum_i P(C_i)$ is particularly useful when you know each outcome in Ω has the same probability. In this case, you can show

$$P(\mathcal{F}) = \frac{\text{Number of outcomes in } \mathcal{F}}{\text{Total number of outcomes in } \Omega}$$

(look at the exercises).

Worked example 4.4 *Odd numbers with fair dice*

We throw a fair (each number has the same probability) die twice, then add the two numbers. What is the probability of getting an odd number?

Solution: There are 36 outcomes. Each has the same probability (1/36). 18 of them give an odd number, and the other 18 give an even number, so the probability is $18/36 = 1/2$

Worked example 4.5 *Numbers divisible by five with fair dice*

We throw a fair (each number has the same probability) die twice, then add the two numbers. What is the probability of getting a number divisible by five?

Solution: There are 36 outcomes. Each has the same probability (1/36). For this event, the spots must add to either 5 or to 10. There are 4 ways to get 5. There are 3 ways to get 10, so the probability is 7/36.

Sometimes a bit of fiddling with the space of outcomes makes it easy to compute what we want. Examples 4.7 and 4.46 show cases where you can use fictitious outcomes as an accounting device to simplify a computation.

Worked example 4.6 *Children - 1*

This example is a version of of example 1.12, p44, in Stirzaker, "Elementary Probability". A couple decides to have children. They decide simply to have three children. Assume that three births occur, each birth results in one child, and boys and girls are equally likely at each birth. Let $\mathcal{B}_\rangle$ be the event that there are i boys, and $\mathcal{C}$ be the event there are more girls than boys. Compute $P(\mathcal{B}_1)$ and $P(\mathcal{C})$.

Solution: There are eight outcomes. Each has the same probability. Three of them have a single boy, so $P(\mathcal{B}_1) = 3/8$. Four of these outcomes have more girls than boys, so $P(\mathcal{C}) = 1/2$.

Worked example 4.7 *Children - 2*

This example is a version of of example 1.12, p44, in Stirzaker, "Elementary Probability". A couple decides to have children. They decide to have children until the first girl is born, or until there are three, and then stop. Assume that each birth results in one child, and boys and girls are equally likely at each birth. Let $\mathcal{B}_i$ be the event that there are i boys, and $\mathcal{C}$ be the event there are more girls than boys. Compute $P(\mathcal{B}_1)$ and $P(\mathcal{C})$.

Solution: In this case, we could write the outcomes as $\{G, BG, BBG\}$, but if we think about them like this, we have no simple way to compute their probability. Instead, we could use the sample space from the previous answer, but assume that some of the later births are fictitious. This gives us natural collection of events for which it is easy to compute probabilities. Having one girl corresponds to the event $\{Gbb, Gbg, Ggb, Ggg\}$, where I have used lowercase letters to write the fictitious later births; the probability is 1/2. Having a boy then a girl corresponds to the event $\{BGb, BGg\}$ (and so has probability 1/4). Having two boys then a girl corresponds to the event $\{BBG\}$ (and so has probability 1/8). Finally, having three boys corresponds to the event $\{BBB\}$ (and so has probability 1/8). This means that $P(\mathcal{B}_1) = 1/4$ and $P(\mathcal{C}) = 1/2$.

Counting outcomes in an event can require pretty elaborate combinatorial arguments. One form of argument that is particularly important is to reason about permutations and combinations. You should recall that the number of distinct permutations of N items is $N!$. The number of combinations of k items, chosen

from N, *where the order does not matter*, is given by

$$\frac{N!}{k!(N-k)!} = \left(\begin{array}{c} N \\ k \end{array} \right).$$

Worked example 4.8 *Card hands*

You draw a hand of seven cards from a properly shuffled standard deck of cards. With what probability do receive 2, 3, 4, 5, 6, 7, 8 of hearts, *in that order*?

Solution: There are numerous ways to do this, but I'll use combinations. There are $\left(\begin{array}{c} 52 \\ 7 \end{array} \right)$ different hands, and each has the same probability because the deck was properly shuffled. Only one of these is the hand we are discussing, so the probability is

$$\frac{1}{\left(\begin{array}{c} 52 \\ 7 \end{array} \right)}.$$

Worked example 4.9 *Card hands - 2*

You draw a hand of seven cards from a properly shuffled standard deck of cards. With what probability do receive 2, 3, 4, 5, 6, 7, 8 of hearts, *in any order*?

Solution: There are numerous ways to do this, but I'll use combinations. There are $\left(\begin{array}{c} 52 \\ 7 \end{array} \right)$ different hands, and each has the same probability because the deck was properly shuffled. Now 7! of these are the hand we are discussing, so the probability is

$$\frac{7!}{\left(\begin{array}{c} 52 \\ 7 \end{array} \right)}.$$

Worked example 4.10 *Card hands - 3*

You draw a hand of seven cards from a properly shuffled standard deck of cards. With what probability does your hand contain 2, 3, 4, 5, 6, 7, 8 of *any suit*? The cards don't have to have the same suit, and they can arrive *in any order*.

Solution: There are numerous ways to do this, but I'll use combinations. There are $\begin{pmatrix} 52 \\ 7 \end{pmatrix}$ different hands, and each has the same probability because the deck was properly shuffled. Now we must count the number of hands we are interested in. For each of the seven numbers, we can choose a suit from four options. Now we can permute that hand. So there are $4^7 7!$ hands, and the probability is

$$\frac{4^7 7!}{\begin{pmatrix} 52 \\ 7 \end{pmatrix}}.$$

Remember this: *In some problems, you can compute the probabilities of events by counting outcomes.*

4.2.2 The Probability of Events

Recall an event is a set of outcomes. Assume we are given a finite sample space Ω. A natural choice of an event space is the collection of all subsets of Ω. It turns out that this is not the only possible choice, but we will ignore this point. So far, we have described the probability of each outcome with a non-negative number. This number represents the relative frequency of the outcome. Because we can tell when an event has occurred, we can compute the relative frequency of events, too. Because it is a relative frequency, the probability of an event is a non-negative number, and is no greater than one. But the probability of events must be consistent with the probability of outcomes. This implies a set of quite straightforward properties:

- **The probability of every event is between zero and one**, which we write $0 \leq P(\mathcal{A}) \leq 1$ for all $\mathcal{A}$ in the collection of events.
- **Every experiment has an outcome**, which we write $P(\Omega) = 1$.
- **The probability of disjoint events is additive**, which requires more notation. Assume that we have a collection of events $\mathcal{A}_i$, indexed by i. We require that these have the property $\mathcal{A}_i \cap \mathcal{A}_j = \varnothing$ when $i \neq j$. This means that there

is no outcome that appears in more than one $\mathcal{A}_i$. In turn, if we interpret probability as relative frequency, we must have that $P(\cup_i \mathcal{A}_i) = \sum_i P(\mathcal{A}_i)$.

Any function P taking events to numbers that has these properties is a probability. These very simple properties imply a series of other very important properties.

Useful Facts: 4.2 *Properties of the probability of events*

- $P(\mathcal{A}^c) = 1 - P(\mathcal{A})$
- $P(\varnothing) = 0$
- $P(\mathcal{A} - \mathcal{B}) = P(\mathcal{A}) - P(\mathcal{A} \cap \mathcal{B})$
- $P(\mathcal{A} \cup \mathcal{B}) = P(\mathcal{A}) + P(\mathcal{B}) - P(\mathcal{A} \cap \mathcal{B})$
- $P(\cup_1^n \mathcal{A}_i) = \sum_i P(\mathcal{A}_i) - \sum_{i<j} P(\mathcal{A}_i \cap \mathcal{A}_j) + \sum_{i<j<k} P(\mathcal{A}_i \cap \mathcal{A}_j \cap \mathcal{A}_k) + \ldots (-1)^{(n+1)} P(\mathcal{A}_1 \cap \mathcal{A}_2 \cap \ldots \cap \mathcal{A}_n)$

I prove each of these below. Looking at the useful facts should suggest a helpful analogy. Think about the probability of an event as the "size" of that event. This "size" is relative to Ω, which has "size" 1. I find this a good way to remember equations. Some people find Venn diagrams a useful way to keep track of this argument, and Figure 4.1 is for them.

This analogy allows us to explain $P(\mathcal{A}) + P(\mathcal{A}^c) = 1$ by noticing that $\mathcal{A}$ and $\mathcal{A}^c$ don't overlap, and together make up all of Ω. So the "size" of $\mathcal{A}$ and the "size" of $\mathcal{A}^c$ should add to the "size" of Ω.

We explain $P(\mathcal{A} - \mathcal{B}) = P(\mathcal{A}) - P(\mathcal{A} \cap \mathcal{B})$ by noticing the "size" of the part of $\mathcal{A}$ that isn't $\mathcal{B}$ is obtained by taking the "size" of $\mathcal{A}$ and subtracting the "size" of the part that is also in $\mathcal{B}$. This analogy, which can be made precise by thinking about "size" in the right way, also explains $P(\mathcal{A} \cup \mathcal{B}) = P(\mathcal{A}) + P(\mathcal{B}) - P(\mathcal{A} \cap \mathcal{B})$. You can get the "size" of $\mathcal{A} \cup \mathcal{B}$ by adding the two "sizes", then subtracting the "size" of the intersection because otherwise you would double-count the part where the two sets overlap.

Proposition: $P(\mathcal{A}^c) = 1 - P(\mathcal{A})$

Proof: $\mathcal{A}^c$ and $\mathcal{A}$ are disjoint, so that $P(\mathcal{A}^c \cup \mathcal{A}) = P(\mathcal{A}^c) + P(\mathcal{A}) = P(\Omega) = 1$.

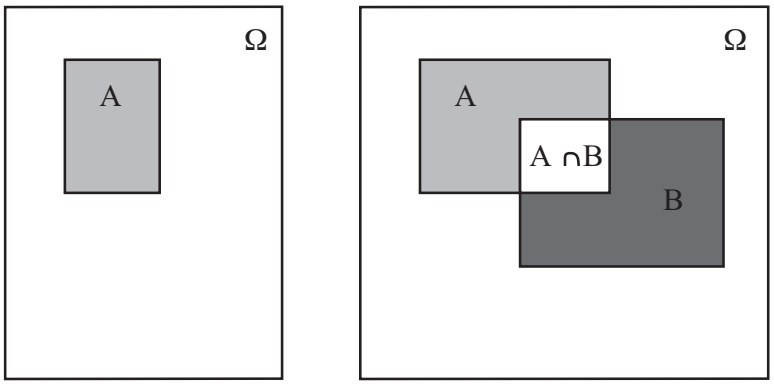

FIGURE 4.1: *If you think of the probability of an event as measuring its "size", many of the rules are quite straightforward to remember. Venn diagrams can sometimes help. On the* **left**, *a Venn diagram to help remember that* $P(\mathcal{A}) + P(\mathcal{A}^c) = 1$. *The "size" of* Ω *is 1, outcomes lie either in* $\mathcal{A}$ *or* $\mathcal{A}^c$, *and the two don't intersect. On the* **right**, *you can see that* $P(\mathcal{A} - \mathcal{B}) = P(\mathcal{A}) - P(\mathcal{A} \cap \mathcal{B})$ *by noticing that* $P(\mathcal{A} - \mathcal{B})$ *is the "size" of the part of* $\mathcal{A}$ *that isn't* $\mathcal{B}$. *This is obtained by taking the "size" of* $\mathcal{A}$ *and subtracting the "size" of the part that is also in* $\mathcal{B}$, *i.e. the "size" of* $\mathcal{A} \cap \mathcal{B}$. *Similarly, you can see that* $P(\mathcal{A} \cup \mathcal{B}) = P(\mathcal{A}) + P(\mathcal{B}) - P(\mathcal{A} \cap \mathcal{B})$ *by noticing that you can get the "size" of* $\mathcal{A} \cup \mathcal{B}$ *by adding the "sizes" of* $\mathcal{A}$ *and* $\mathcal{B}$, *then subtracting the "size" of the intersection to avoid double counting.*

Proposition: $P(\varnothing) = 0$

Proof: $P(\varnothing) = P(\Omega^c) = P(\Omega - \Omega) = 1 - P(\Omega) = 1 - 1 = 0.$

Proposition: $P(\mathcal{A} - \mathcal{B}) = P(\mathcal{A}) - P(\mathcal{A} \cap \mathcal{B})$

Proof: $\mathcal{A} - \mathcal{B}$ is disjoint from $\mathcal{A} \cap \mathcal{B}$, and $(\mathcal{A} - \mathcal{B}) \cup (\mathcal{A} \cap \mathcal{B}) = \mathcal{A}$. This means that $P(\mathcal{A} - \mathcal{B}) + P(\mathcal{A} \cap \mathcal{B}) = P(\mathcal{A})$.

Proposition: $P(\mathcal{A} \cup \mathcal{B}) = P(\mathcal{A}) + P(\mathcal{B}) - P(\mathcal{A} \cap \mathcal{B})$

Proof: $P(\mathcal{A} \cup \mathcal{B}) = P(\mathcal{A} \cup (\mathcal{B} \cap \mathcal{A}^c)) = P(\mathcal{A}) + P((\mathcal{B} \cap \mathcal{A}^c))$. Now $\mathcal{B} = (\mathcal{B} \cap \mathcal{A}) \cup (\mathcal{B} \cap \mathcal{A}^c)$. Furthermore, $(\mathcal{B} \cap \mathcal{A})$ is disjoint from $(\mathcal{B} \cap \mathcal{A}^c)$, so we have $P(\mathcal{B}) = P((\mathcal{B} \cap \mathcal{A})) + P((\mathcal{B} \cap \mathcal{A}^c))$. This means that $P(\mathcal{A} \cup \mathcal{B}) = P(\mathcal{A}) + P((\mathcal{B} \cap \mathcal{A}^c)) = P(\mathcal{A}) + P(\mathcal{B}) - P((\mathcal{B} \cap \mathcal{A}))$.

Proposition: $P(\cup_1^n \mathcal{A}_i) = \sum_i P(\mathcal{A}_i) - \sum_{i<j} P(\mathcal{A}_i \cap \mathcal{A}_j) + \sum_{i<j<k} P(\mathcal{A}_i \cap \mathcal{A}_j \cap \mathcal{A}_k) + \ldots (-1)^{(n+1)} P(\mathcal{A}_1 \cap \mathcal{A}_2 \cap \ldots \cap \mathcal{A}_n)$

Proof: This can be proven by repeated application of the previous result. As an example, we show how to work the case where there are three sets (you can get the rest by induction).

$$\begin{aligned}
P(\mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3) &= P(\mathcal{A}_1 \cup (\mathcal{A}_2 \cup \mathcal{A}_3)) \\
&= P(\mathcal{A}_1) + P(\mathcal{A}_2 \cup \mathcal{A}_3) - P(\mathcal{A}_1 \cap (\mathcal{A}_2 \cup \mathcal{A}_3)) \\
&= P(\mathcal{A}_1) + (P(\mathcal{A}_2) + P(\mathcal{A}_3) - P(\mathcal{A}_2 \cap \mathcal{A}_3)) - \\
&\quad P((\mathcal{A}_1 \cap \mathcal{A}_2) \cup (\mathcal{A}_1 \cap \mathcal{A}_3)) \\
&= P(\mathcal{A}_1) + (P(\mathcal{A}_2) + P(\mathcal{A}_3) - P(\mathcal{A}_2 \cap \mathcal{A}_3)) - \\
&\quad P(\mathcal{A}_1 \cap \mathcal{A}_2) - P(\mathcal{A}_1 \cap \mathcal{A}_3) \\
&\quad -(-P((\mathcal{A}_1 \cap \mathcal{A}_2) \cap (\mathcal{A}_1 \cap \mathcal{A}_3))) \\
&= P(\mathcal{A}_1) + P(\mathcal{A}_2) + P(\mathcal{A}_3) - \\
&\quad P(\mathcal{A}_2 \cap \mathcal{A}_3) - P(\mathcal{A}_1 \cap \mathcal{A}_2) - P(\mathcal{A}_1 \cap \mathcal{A}_3) + \\
&\quad P(\mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{A}_3)
\end{aligned}$$

4.2.3 Computing Probabilities by Reasoning about Sets

The rule $P(\mathcal{A}^c) = 1 - P(\mathcal{A})$ is occasionally useful for computing probabilities on its own. More commonly, you need other reasoning as well. The next problem illustrates an important feature of questions in probability: your intuition can be quite misleading. One problem is that the number of outcomes can be bigger or smaller than you expect.

Worked example 4.11 *Shared birthdays*

What is the probability that, in a room of 30 people, there is a pair of people who have the same birthday?

Solution: We simplify, and assume that each year has 365 days, and that none of them are special (i.e. each day has the same probability of being chosen as a birthday). This model isn't perfect (there tend to be slightly more births roughly 9 months after: the start of spring; blackouts; major disasters; and so on) but it's workable. The easy way to attack this question is to notice that our probability, $P(\{\text{shared birthday}\})$, is

$$1 - P(\{\text{all birthdays different}\}).$$

This second probability is rather easy to compute. Each outcome in the sample space is a list of 30 days (one birthday per person). Each outcome has the same probability. So

$$P(\{\text{all birthdays different}\}) = \frac{\text{Number of outcomes in the event}}{\text{Total number of outcomes}}.$$

The total number of outcomes is easily seen to be 365^{30}, which is the total number of possible lists of 30 days. The number of outcomes in the event is the number of lists of 30 days, all different. To count these, we notice that there are 365 choices for the first day; 364 for the second; and so on. So we have

$$P(\{\text{shared birthday}\}) = 1 - \frac{365 \times 364 \times \ldots 336}{365^{30}} = 1 - 0.2937 = 0.7063$$

which means there's really a pretty good chance that two people in a room of 30 share a birthday.

If we change the birthday example slightly, the problem changes drastically. If you stand up in a room of 30 people and bet that two people in the room have the same birthday, you have a probability of winning of about 0.71. If you bet that there is someone else in the room who has the same birthday that you do, your probability of winning is very different.

Worked example 4.12 *Shared birthdays*

You bet there is someone else in a room of 30 people who has the same birthday that you do. Assuming you know nothing about the other 29 people, what is the probability of winning?

Solution: The easy way to do this is

$$P(\{\text{winning}\}) = 1 - P(\{\text{losing}\}).$$

Now you will lose if everyone has a birthday different from you. You can think of the birthdays of the others in the room as a list of 29 days of the year. If your birthday is on the list, you win; if it's not, you lose. The number of losing lists is the number of lists of 29 days of the year such that your birthday is not in the list. This number is easy to get. We have 364 days of the year to choose from for each of 29 locations in the list. The total number of lists is the number of lists of 29 days of the year. Each list has the same probability. So

$$P(\{\text{losing}\}) = \frac{364^{29}}{365^{29}}$$

and

$$P(\{\text{winning}\}) \approx 0.0765.$$

There is a wide variety of problems like this; if you're so inclined, you can make a small but quite reliable profit off people's inability to estimate probabilities for this kind of problem correctly (examples 4.11 and 4.12 are reliably profitable; you could probably do quite well out of examples 4.44 and 4.45).

The rule $P(\mathcal{A}-\mathcal{B}) = P(\mathcal{A})-P(\mathcal{A}\cap\mathcal{B})$ is also occasionally useful for computing probabilities on its own; more commonly, you need other reasoning as well.

Worked example 4.13 *Dice*

You flip two fair six-sided dice, and add the number of spots. What is the probability of getting a number divisible by 2, but not by 5?

Solution: There is an interesting way to work the problem. Write $\mathcal{D}_n$ for the event the number is divisible by n. Now $P(\mathcal{D}_2) = 1/2$ (count the cases; or, more elegantly, notice that each die has the same number of odd and even faces, and work from there). Now $P(\mathcal{D}_2-\mathcal{D}_5) = P(\mathcal{D}_2)-P(\mathcal{D}_2\cap\mathcal{D}_5)$. But $\mathcal{D}_2\cap\mathcal{D}_5$ contains only three outcomes $(6,4,\ 5,5$ and $4,6)$, so $P(\mathcal{D}_2-\mathcal{D}_5) = 18/36-3/36 = 5/12$

Sometimes it is easier to reason about unions than to count outcomes directly.

Worked example 4.14 *Two fair dice*

I roll two fair dice. What is the probability that the result is divisible by either 2 or 5, or both?

Solution: Write $\mathcal{D}_n$ for the event the number is divisible by n. We want $P(\mathcal{D}_2 \cup \mathcal{D}_5) = P(\mathcal{D}_2) + P(\mathcal{D}_5) - P(\mathcal{D}_2 \cap \mathcal{D}_5)$. From example 4.13, we know $P(\mathcal{D}_2) = 1/2$ and $P(\mathcal{D}_2 \cap \mathcal{D}_5) = 3/36$. By counting outcomes, $P(\mathcal{D}_5) = 7/36$. So $P(\mathcal{D}_2 \cup \mathcal{D}_5) = (18 + 7 - 3)/36 = 22/36$.

4.3 INDEPENDENCE

Some experimental results do not affect others. For example, if I flip a coin twice, whether I get heads on the first flip has no effect on whether I get heads on the second flip. As another example, I flip a coin; the outcome does not affect whether I get hit on the head by a falling apple later in the day. We refer to events with this property as **independent**.

Here is a pair of events that is not independent. Imagine I throw a die. Write $\mathcal{A}$ for the event that the die comes up with an odd number of spots, and write $\mathcal{B}$ for the event that the number of spots is either 3 or 5. Now these events are interrelated in an important way. If I *know* that $\mathcal{B}$ has occurred, I also know that $\mathcal{A}$ has occurred — I don't need to check separately, because $\mathcal{B}$ implies $\mathcal{A}$.

Here is an example of a weaker interaction that results in events not being independent. Write $\mathcal{C}$ for the event that the die comes up with an odd number of spots, and write $\mathcal{D}$ for the event that the number of spots is larger than 3. These events are interrelated. The probability of each event separately is $1/2$. If I *know* that $\mathcal{C}$ has occurred, then I know that the die shows either 1, 3, or 5 spots. One of these outcomes belongs to $\mathcal{D}$, and two do not. This means that knowing that $\mathcal{C}$ has occured tells you something about whether $\mathcal{D}$ has occurred. Independent events do *not* have this property. This means that the probability that they occur together has an important property, given in the box below.

Definition: 4.2 *Independent events*

Two events $\mathcal{A}$ and $\mathcal{B}$ are **independent** if and only if

$$P(\mathcal{A} \cap \mathcal{B}) = P(\mathcal{A})P(\mathcal{B})$$

The "size" analogy helps motivate this expression. We think of $P(A)$ as the "size" of $\mathcal{A}$ relative to Ω, and so on. Now $P(\mathcal{A} \cap \mathcal{B})$ measures the "size" of $\mathcal{A} \cap \mathcal{B}$ — that is, the part of $\mathcal{A}$ that lies inside $\mathcal{B}$. But if $\mathcal{A}$ and $\mathcal{B}$ are independent, then

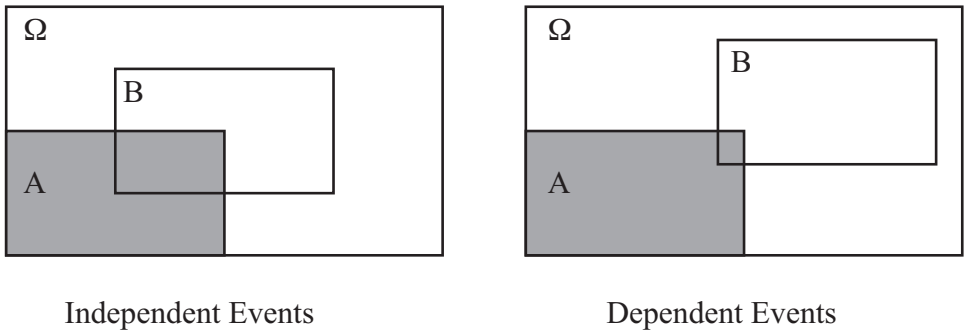

FIGURE 4.2: *On the* **left**, $\mathcal{A}$ *and* $\mathcal{B}$ *are independent.* $\mathcal{A}$ *spans* $1/4$ *of* Ω, *and* $\mathcal{A} \cap \mathcal{B}$ *spans* $1/4$ *of* $\mathcal{B}$. *This means that knowing whether an outcome is in* $\mathcal{A}$ *or not doesn't affect the probability that it is in* $\mathcal{B}$. $1/4$ *of the outcomes of* Ω *lie in* $\mathcal{A}$, *and* $1/4$ *of the outcomes in* $\mathcal{B}$ *lie in* $\mathcal{A} \cap \mathcal{B}$. *On the* **right**, *they are not. Very few of the outcomes in* $\mathcal{B}$ *lie in* $\mathcal{B} \cap \mathcal{A}$, *so that observing* $\mathcal{B}$ *means that* $\mathcal{A}$ *becomes less likely, because very few of the outcomes in* $\mathcal{B}$ *also lie in* $\mathcal{A} \cap \mathcal{B}$.

the size of $\mathcal{A} \cap \mathcal{B}$ relative to $\mathcal{B}$ should be the same as the size of $\mathcal{A}$ relative to Ω (Figure 4.2). Otherwise, $\mathcal{B}$ affects $\mathcal{A}$, because $\mathcal{A}$ is more (or less) likely when $\mathcal{B}$ has occurred.

So for $\mathcal{A}$ and $\mathcal{B}$ to be independent, we must have

$$\text{“Size” of } \mathcal{A} = \frac{\text{“Size” of piece of } \mathcal{A} \text{ in } \mathcal{B}}{\text{“Size” of } \mathcal{B}},$$

or, equivalently,

$$P(\mathcal{A}) = \frac{P(\mathcal{A} \cap \mathcal{B})}{P(\mathcal{B})}$$

which yields our expression.

Worked example 4.15 *A fair die*

The space of outcomes for a fair six-sided die is

$$\{1, 2, 3, 4, 5, 6\}.$$

The die is fair means that each outcome has the same probability. Now we toss two fair dice. The outcome for each die is independent of that for the other. With what probability do we get two threes?

Solution:

$$\begin{aligned} P(\text{first die yields 3} \cap \text{second die yields 3}) &= P(\text{first die yields 3}) \times \\ &\quad P(\text{second die yields 3}) \\ &= (1/6)(1/6) \\ &= 1/36 \end{aligned}$$

Worked example 4.16 *Find the lady, twice*

Recall the setup of worked example **??**. Assume that the card that is chosen is chosen fairly — that is, each card is chosen with the same probability. The game is played twice, and the cards are reshuffled between games. What is the probability of turning up a Queen and then a Queen again?

Solution: The events are independent, so 1/9.

You can use definition 4.2 (i.e. $\mathcal{A}$ and $\mathcal{B}$ are independent if and only if $P(\mathcal{A} \cap \mathcal{B}) = P(\mathcal{A})P(\mathcal{B})$) to tell whether events are independent or not. Quite small changes to a problem affect whether events are independent, as in the worked example below.

Worked example 4.17 *Cards and Independence*

We shuffle a standard deck of 52 cards and draw one card. The event $\mathcal{A}$ is "the card is a red suit" and the event $\mathcal{B}$ is "the card is a 10". (1): Are $\mathcal{A}$ and $\mathcal{B}$ independent?
Now we take a standard deck of cards, and remove the ten of hearts. We shuffle this deck, and draw one card. The event $\mathcal{C}$ is "the card drawn from the modified deck is a red suit" and the event $\mathcal{D}$ is "the card drawn from the modified deck is a 10". (2): Are $\mathcal{C}$ and $\mathcal{D}$ independent?

Solution: (1): $P(\mathcal{A}) = 1/2$, $P(\mathcal{B}) = 1/13$ and in example 4.43 we determined $P(\mathcal{A} \cap \mathcal{B}) = 2/52$. But $2/52 = 1/26 = P(\mathcal{A})P(\mathcal{B})$, so they are independent.
(2): These are not independent because $P(\mathcal{C}) = 25/51$, $P(\mathcal{D}) = 3/51$ and $P(\mathcal{C} \cap \mathcal{D}) = 1/51 \neq P(\mathcal{C})P(\mathcal{D}) = 75/(51^2)$

The probability of a sequence of independent events can become very small very quickly, and this often misleads people.

Worked example 4.18 *Accidental DNA Matches*

I search a DNA database with a sample. Each time I attempt to match this sample to an entry in the database, there is a probability of an accidental chance match of $1e-4$. Chance matches are independent. There are 20, 000 people in the database. What is the probability I get at least one match, purely by chance?

Solution: This is $1 - P(\text{no chance matches})$. But $P(\text{no chance matches})$ is much smaller than you think. We have

$$\begin{aligned} P(\text{no chance matches}) &= P\left(\begin{array}{c} \text{no chance match to record } 1 \cap \\ \text{no chance match to record } 2 \cap \\ \ldots \cap \\ \text{no chance match to record } 20,000 \end{array}\right) \\ &= P(\text{no chance match to a record})^{20,000} \\ &= (1 - 1e-4)^{20,000} \\ &\approx 0.14 \end{aligned}$$

so the probability is about 0.86 that you get at least one match by chance. If you're surprised, look at the exponent. Notice that if the database gets bigger, the probability grows; so at 40, 000 the probability of one match by chance is 0.98.

Remember this: *The probability of a set of independent events can become very small, and this often misleads the intuition.*

4.3.1 Example: Airline Overbooking

We can now quite easily study airline overbooking. Airlines generally sell more tickets for a flight than there are seats on the aircraft, because some passengers don't turn up on time, usually for random reasons. If the airline only sold one ticket per seat, their planes would likely have empty seats — which are lost profit — on each flight. If too many passengers turn up for a flight, someone will accept money to take the next flight. The amount of money that an airline has to give a passenger who doesn't get a seat is usually prescribed by regulators. Now an airline must compute with what probability they will have to pay out as a function of the number of tickets sold.

Worked example 4.19 *Overbooking - 1*

An airline has a regular flight with six seats. It always sells seven tickets. Passengers turn up for the flight with probability p, and do so independent of other passengers. What is the probability that the flight is overbooked?

Solution: This is like a coin-flip problem; think of each passenger as a biased coin. With probability p, the biased coin comes up T (for *t*urn up) and with probability $(1-p)$, it turns up H (for *n*o-show). This coin is flipped seven times, and we are interested in the probability that there are seven T's. This is p^7, because the flips are independent.

Worked example 4.20 *Overbooking - 2*

An airline has a regular flight with six seats. It always sells eight tickets. Passengers turn up for the flight with probability p, and do so independent of other passengers. What is the probability that the flight is overbooked?

Solution: Now we flip the coin eight times, and are interested in the probability of getting more than six T's. This is the union of two disjoint events (seven T's and eight T's). For the case of seven T's, one flip must be H; there are eight choices. For the case of eight T's, all eight flips must be T, and there is only one way to achieve this. So the probability the flight is overbooked is

$$\begin{aligned} P(\text{overbooked}) &= P(7\ T\text{'s} \cup 8\ T\text{'s}) \\ &= P(7\ T\text{'s}) + P(8\ T\text{'s}) \\ &= 8p^7(1-p) + p^8 \end{aligned}$$

Worked example 4.21 *Overbooking - 3*

An airline has a regular flight with six seats. It always sells eight tickets. Passengers turn up for the flight with probability p, and do so independent of other passengers. What is the probability that six passengers arrive? (i.e. the flight is not overbooked or underbooked).

Solution: Now we flip the coin eight times, and are interested in the probability of getting exactly six T's. The probability that a particular set of six passengers arrives is given by the probability of getting any given string of six T's and two H's. This must have probability $p^6(1-p)^2$. But there are a total of $\frac{8!}{2!6!}$ such strings. So the probability that six passengers arrive is

$$\frac{8!}{2!6!}p^6(1-p)^2 = 28p^6(1-p)^2.$$

Worked example 4.22 *Overbooking - 4*

An airline has a regular flight with s seats. It always sells t tickets. Passengers turn up for the flight with probability p, and do so independent of other passengers. What is the probability that u passengers turn up?

Solution: Now we flip the coin t times, and are interested in the probability of getting u T's. There are

$$\frac{t!}{u!(t-u)!}$$

disjoint outcomes with u T's and $t-u$ N's. Each such outcome is independent, and has probability $p^u(1-p)^{t-u}$. So

$$P(u \text{ passengers turn up}) = \frac{t!}{u!(t-u)!}p^u(1-p)^{t-u}$$

Worked example 4.23 *Overbooking - 5*

An airline has a regular flight with s seats. It always sells t tickets. Passengers turn up for the flight with probability p, and do so independent of other passengers. What is the probability that the flight is oversold?

Solution: We need $P(\{s+1 \text{ turn up}\} \cup \{s+2 \text{ turn up}\} \cup \ldots \cup \{t \text{ turn up}\})$. But the events $\{i \text{ turn up}\}$ and $\{j \text{ turn up}\}$ are disjoint if $i \neq j$. So we can exploit example 4.22, and write

$$\begin{aligned} P(\text{overbooked}) &= P(\{s+1 \text{ turn up}\}) + P(\{s+2 \text{ turn up}\}) + \\ &\quad \ldots P(\{t \text{ turn up}\}) \\ &= \sum_{i=s+1}^{t} P(\{i \text{ turn up}\}) \\ &= \sum_{i=s+1}^{t} \frac{t!}{i!(t-i)!}p^i(1-p)^{t-i} \end{aligned}$$

4.4 CONDITIONAL PROBABILITY

As we saw in section 4.3, observing one event may change the probability another event will occur. Conditional probability is a mechanism to keep track of this effect.

There were several examples of events that weren't independent in that section (and you should look at these again). Here is another example. If you throw a fair

die twice and add the numbers, then the probability of getting a number less than six is $\frac{10}{36}$. Now imagine you *know* that the first die came up three. In this case, the probability that the sum will be less than six is $\frac{1}{3}$, which is slightly larger than $\frac{10}{36}$. If the first die came up four, then the probability the sum will be less than six is $\frac{1}{6}$, which is rather less than $\frac{10}{36}$. If the first die came up one, then the probability that the sum is less than six becomes $\frac{2}{3}$, which is much larger than $\frac{10}{36}$.

Each of these probabilities is an example of a **conditional probability**. We assume we have a space of outcomes and a collection of events. The conditional probability of $\mathcal{B}$, conditioned on $\mathcal{A}$, is the probability that $\mathcal{B}$ occurs given that $\mathcal{A}$ has definitely occurred. We write this as

$$P(\mathcal{B}|\mathcal{A})$$

From the examples, it should be clear to you that for some cases $P(\mathcal{B}|\mathcal{A})$ is the same as $P(\mathcal{B})$, and for other cases it is not.

4.4.1 Evaluating Conditional Probabilities

To get an expression for $P(\mathcal{B}|\mathcal{A})$, notice that, because $\mathcal{A}$ is known to have occurred, our space of outcomes or sample space is now reduced to $\mathcal{A}$. We know that our outcome lies in $\mathcal{A}$; $P(\mathcal{B}|\mathcal{A})$ is the probability that it also lies in $\mathcal{B} \cap \mathcal{A}$.

The outcome lies in $\mathcal{A}$, and so it must lie in either $\mathcal{B} \cap \mathcal{A}$ or in $\mathcal{B}^c \cap \mathcal{A}$, and it cannot lie in both. This means that

$$P(\mathcal{B}|\mathcal{A}) + P(\mathcal{B}^c|\mathcal{A}) = 1.$$

Now recall the idea of probabilities as relative frequencies. If $P(\mathcal{C} \cap \mathcal{A}) = kP(\mathcal{B} \cap \mathcal{A})$, this means that outcomes in $\mathcal{C} \cap \mathcal{A}$ will appear k times as often as outcomes in $\mathcal{B} \cap \mathcal{A}$. But this must apply even if we know in advance that the outcome is in $\mathcal{A}$. This means that, if $P(\mathcal{C} \cap \mathcal{A}) = kP(\mathcal{B} \cap \mathcal{A})$, then $P(\mathcal{C}|\mathcal{A}) = kP(\mathcal{B}|\mathcal{A})$. In turn, we must have

$$P(\mathcal{B}|\mathcal{A}) \propto P(\mathcal{B} \cap \mathcal{A}).$$

Now we need to determine the constant of proportionality; write c for this constant, meaning

$$P(\mathcal{B}|\mathcal{A}) = cP(\mathcal{B} \cap \mathcal{A}).$$

We have that

$$P(\mathcal{B}|\mathcal{A}) + P(\mathcal{B}^c|\mathcal{A}) = cP(\mathcal{B} \cap \mathcal{A}) + cP(\mathcal{B}^c \cap \mathcal{A}) = cP(\mathcal{A}) = 1,$$

so that

$$P(\mathcal{B}|\mathcal{A}) = \frac{P(\mathcal{B} \cap \mathcal{A})}{P(\mathcal{A})}.$$

I find the "size" metaphor helpful here. We have that $P(\mathcal{B}|\mathcal{A})$ measures the probability that an outcome is in $\mathcal{B}$, given we *know* it is in $\mathcal{A}$. From the "size" perspective, $P(\mathcal{B}|\mathcal{A})$ measures the size of $(\mathcal{A} \cap \mathcal{B})$ *relative to* $\mathcal{A}$. So our expression makes sense, because the fraction of the event $\mathcal{A}$ that is also part of the event $\mathcal{B}$ is given by the size of the intersection divided by the size of $\mathcal{A}$.

Another, very useful, way to write the expression $P(\mathcal{B}|\mathcal{A}) = P(\mathcal{B} \cap \mathcal{A})/P(\mathcal{A})$ is:

$$P(\mathcal{B}|\mathcal{A})P(\mathcal{A}) = P(\mathcal{B} \cap \mathcal{A}).$$

Now, since $\mathcal{B} \cap \mathcal{A} = \mathcal{A} \cap \mathcal{B}$, we must have that

$$P(\mathcal{B}|\mathcal{A}) = \frac{P(\mathcal{A}|\mathcal{B})P(\mathcal{B})}{P(\mathcal{A})}$$

Worked example 4.24 *Car factories*

There are two car factories, A and B. Each year, factory A produces 1000 cars, of which 10 are lemons. Factory B produces 2 cars, each of which is a lemon. All cars go to a single lot, where they are thoroughly mixed up. I buy a car.

- What is the probability it is a lemon?
- What is the probability it came from factory B?
- The car is now revealed to be a lemon. What is the probability it came from factory B, conditioned on the fact it is a lemon?

Solution:

- Write the event the car is a lemon as $\mathcal{L}$. There are 1002 cars, of which 12 are lemons. The probability that I select any given car is the same, so we have $P(\mathcal{L}) = 12/1002$.
- Same argument yields $P(\mathcal{B}) = 2/1002$.
- Write $\mathcal{B}$ for the event the car comes from factory B. I need $P(\mathcal{B}|\mathcal{L}) = P(\mathcal{L} \cap \mathcal{B})/P(\mathcal{L}) = P(\mathcal{L}|\mathcal{B})P(\mathcal{B})/P(\mathcal{L})$. I have $P(\mathcal{L}|\mathcal{B})P(\mathcal{B})/P(\mathcal{L}) = (1 \times 2/1002)/(12/1002) = 1/6$.

Worked example 4.25 *Royal flushes in poker - 1*

This exercise is after Stirzaker, p. 51.
You are playing a straightforward version of poker, where you are dealt five cards face down. A royal flush is a hand of AKQJ10 all in one suit. What is the probability that you are dealt a royal flush?

Solution: This is

$$\frac{\text{number of hands that are royal flushes, ignoring card order}}{\text{total number of different five card hands, ignoring card order}}.$$

There are four hands that are royal flushes (one for each suit). Now the total number of five card hands is

$$\left(\begin{array}{c} 52 \\ 5 \end{array} \right) = 2598960$$

so we have

$$\frac{4}{2598960} = \frac{1}{649740}.$$

Worked example 4.26 *Royal flushes in poker - 2*

This exercise is after Stirzaker, p. 51.
You are playing a straightforward version of poker, where you are dealt five cards face down. A royal flush is a hand of AKQJ10 all in one suit. The fifth card that you are dealt lands face up. It is the nine of spades. What now is the probability that your have been dealt a royal flush? (i.e. what is the conditional probability of getting a royal flush, conditioned on the event that one card is the nine of spades)

Solution: No hand containing a nine of spades is a royal flush, so this is easily zero.

Worked example 4.27 *Royal flushes in poker - 3*

This exercise is after Stirzaker, p. 51.
You are playing a straightforward version of poker, where you are dealt five cards face down. A royal flush is a hand of AKQJ10 all in one suit. The fifth card that you are dealt lands face up. It is the Ace of spades. What now is the probability that your have been dealt a royal flush? (i.e. what is the conditional probability of getting a royal flush, conditioned on the event that one card is the Ace of spades)

Solution: Now consider the events

$$\mathcal{A} = \text{you get a royal flush } \textit{and} \text{ the last card is the ace of spades}$$

and

$$\mathcal{B} = \text{the last card you get is the ace of spades,}$$

and the expression

$$P(\mathcal{A}|\mathcal{B}) = \frac{P(\mathcal{A} \cap \mathcal{B})}{P(\mathcal{B})}.$$

Now $P(\mathcal{B}) = \frac{1}{52}$. $P(\mathcal{A} \cap \mathcal{B})$ is given by

$$\frac{\text{number of five card royal flushes where card five is Ace of spades}}{\text{total number of different five card hands}}.$$

where we DO NOT ignore card order. This is

$$\frac{4 \times 3 \times 2 \times 1}{52 \times 51 \times 50 \times 49 \times 48}$$

yielding

$$P(\mathcal{A}|\mathcal{B}) = \frac{1}{249900}.$$

Notice the interesting part: the conditional probability is rather larger than the probability. If you see this ace, the conditional probability is $\frac{13}{5}$ times the probability that you will get a flush if you don't. Seeing this card has really made a difference.

Worked example 4.28 *Two dice*

We throw two fair dice. What is the conditional probability that the sum of spots on both dice is greater than six, conditioned on the event that the first die comes up five?

Solution: Write the event that the first die comes up 5 as $\mathcal{F}$, and the event the sum is greather than six as $\mathcal{S}$. There are five outcomes where the first die comes up 5 and the number is greater than 6, so $P(\mathcal{F} \cap \mathcal{S}) = 5/36$. Now

$$P(\mathcal{S}|\mathcal{F}) = P(\mathcal{F} \cap \mathcal{S})/P(\mathcal{F}) = (5/36)/(1/6) = 5/6.$$

Notice that $\mathcal{A} \cap \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}^c$ are disjoint sets, and that $\mathcal{A} = (\mathcal{A} \cap \mathcal{B}) \cup (\mathcal{A} \cap \mathcal{B}^c)$. So, because $P(\mathcal{A}) = P(\mathcal{A} \cap \mathcal{B}) + P(\mathcal{A} \cap \mathcal{B}^c)$, we have

$$P(\mathcal{A}) = P(\mathcal{A}|\mathcal{B})P(\mathcal{B}) + P(\mathcal{A}|\mathcal{B}^c)P(\mathcal{B}^c)$$

a tremendously important and useful fact. Another version of this fact is also very useful. Assume we have a collection of disjoint sets $\mathcal{B}_i$. These sets must have the property that (a) $\mathcal{B}_i \cap \mathcal{B}_j = \varnothing$ for $i \neq j$ and (b) they cover $\mathcal{A}$, meaning that $\mathcal{A} \cap (\cup_i \mathcal{B}_i) = \mathcal{A}$. Then, because $P(\mathcal{A}) = \sum_i P(\mathcal{A} \cap \mathcal{B}_i)$, so we have

$$P(\mathcal{A}) = \sum_i P(\mathcal{A}|\mathcal{B}_i)P(\mathcal{B}_i)$$

It is wise to be suspicious of your intuitions when thinking about problems in conditional probability. There is a really big difference between $P(\mathcal{A}|\mathcal{B})$ and $P(\mathcal{B}|\mathcal{A})P(\mathcal{A})$. Not respecting this difference can lead to serious problems (section **??**), and seems to be easy to do. The division sign in the expression

$$P(\mathcal{A}|\mathcal{B}) = P(\mathcal{B}|\mathcal{A})P(\mathcal{A})/P(\mathcal{B})$$

can have alarming effects; as a result, most people have quite poor intuitions about conditional probability. Here is one helpful example. If you buy a lottery ticket ($\mathcal{L}$), the probability of winning ($\mathcal{W}$) is small. So $P(\mathcal{W}|\mathcal{L})$ may be very small. But $P(\mathcal{L}|\mathcal{W})$ is 1 — the winner is always someone who bought a ticket.

Useful Facts: 4.3 *Conditional probability formulas*

You should remember the following formulas:

- $P(\mathcal{B}|\mathcal{A}) = \dfrac{P(\mathcal{A}|\mathcal{B})P(\mathcal{B})}{P(\mathcal{A})}$
- $P(\mathcal{A}) = P(\mathcal{A}|\mathcal{B})P(\mathcal{B}) + P(\mathcal{A}|\mathcal{B}^c)P(\mathcal{B}^c)$
- Assume (a) $\mathcal{B}_i \cap \mathcal{B}_j = \varnothing$ for $i \neq j$ and (b) $\mathcal{A} \cap (\cup_i \mathcal{B}_i) = \mathcal{A}$; then $P(\mathcal{A}) = \sum_i P(\mathcal{A}|\mathcal{B}_i)P(\mathcal{B}_i)$

Worked example 4.29 *False positives*

After Stirzaker, p55. You have a blood test for a rare disease that occurs by chance in 1 person in 100, 000. If you have the disease, the test will report that you do with probability 0.95 (and that you do not with probability 0.05). If you do not have the disease, the test will report a false positive with probability 1e-3. If the test says you do have the disease, what is the probability it that you actually have the disease?

Solution: Write S for the event you are sick and R for the event the test reports you are sick. We need $P(S|R)$. We have

$$\begin{aligned} P(S|R) &= \frac{P(R|S)P(S)}{P(R)} \\ &= \frac{P(R|S)P(S)}{P(R|S)P(S) + P(R|S^c)P(S^c)} \\ &= \frac{0.95 \times 1e-5}{0.95 \times 1e-5 + 1e-3 \times (1 - 1e-5)} \\ &= 0.0094 \end{aligned}$$

which should strike you as being a bit alarming. Notice what is happening here. There are two ways that the test could come back positive: either you have the disease, or the test is producing a false positive. But the disease is so rare that it's much more likely you have a false positive result than you have the disease.

Worked example 4.30 *False positives -2*

After Stirzaker, p55. You want to *design* a blood test for a rare disease that occurs by chance in 1 person in 100, 000. If you have the disease, the test will report that you do with probability p (and that you do not with probability $(1-p)$). If you do not have the disease, the test will report a false positive with probability q. You want to choose the value of p so that if the test says you have the disease, there is at least a 50% probability that you do.

Solution: Write S for the event you are sick and R for the event the test reports you are sick. We need $P(S|R)$. We have

$$\begin{aligned} P(S|R) &= \frac{P(R|S)P(S)}{P(R)} \\ &= \frac{P(R|S)P(S)}{P(R|S)P(S) + P(R|S^c)P(S^c)} \\ &= \frac{p \times 1e-5}{p \times 1e-5 + q \times (1 - 1e-5)} \\ &\geq 0.5 \end{aligned}$$

which means that $p \geq 99999q$ which should strike you as being very alarming indeed, because $p \leq 1$ and $q \geq 0$. One plausible pair of values is $q = 1e-5$, $p = 1 - 1e-5$. The test has to be spectacularly accurate to be of any use.

4.4.2 Independence and Conditional Probability

As we have seen, two events are independent if

$$P(\mathcal{A} \cap \mathcal{B}) = P(\mathcal{A})P(\mathcal{B}).$$

If two events $\mathcal{A}$ and $\mathcal{B}$ are independent, then

$$P(\mathcal{A}|\mathcal{B}) = P(\mathcal{A})$$

and

$$P(\mathcal{B}|\mathcal{A}) = P(\mathcal{B}).$$

Again, this means that knowing that $\mathcal{A}$ occurred tells you nothing about $\mathcal{B}$ — the probability that $\mathcal{B}$ will occur is the same whether you know that $\mathcal{A}$ occurred or not. There are weaker kinds of independence that are sometimes useful.

Definition: 4.3 *Pairwise independence and conditional independence*

Events $\mathcal{A}_1 \ldots \mathcal{A}_n$ are **pairwise independent** if each pair is independent (i.e. $\mathcal{A}_1$ and $\mathcal{A}_2$ are independent, etc.). They are **conditionally independent** conditioned on event $\mathcal{B}$ if

$$P(\mathcal{A}_{i_1} \cap \ldots \cap \mathcal{A}_{i_k}|\mathcal{B}) = P(\mathcal{A}_{i_1}|\mathcal{B}) \ldots P(\mathcal{A}_{i_k}|\mathcal{B})$$

We usually do not have the information required to prove that events are independent. Instead, we use intuition (for example, two flips of the same coin are likely to be independent unless there is something very funny going on) or simply choose to apply models in which some variables are independent.

Worked example 4.31 *Cards and pairwise independence*

We draw three cards from a properly shuffled standard deck, with replacement and reshuffling (i.e., draw a card, make a note, return to deck, shuffle, draw the next, make a note, shuffle, draw the third). Let $\mathcal{A}$ be the event that "card 1 and card 2 have the same suit"; let $\mathcal{B}$ be the event that "card 2 and card 3 have the same suit"; let $\mathcal{C}$ be the event that "card 1 and card 3 have the same suit". Show these events are pairwise independent, but not independent.

Solution: By counting, you can check that $P(\mathcal{A}) = 1/4$; $P(\mathcal{B}) = 1/4$; and $P(\mathcal{A} \cap \mathcal{B}) = 1/16$, so that these two are independent. This argument works for other pairs, too. But $P(\mathcal{C} \cap \mathcal{A} \cap \mathcal{B}) = 1/16$ which is not $1/4^3$, so the events are not independent; this is because the third event is logically implied by the first two.

Worked example 4.32 *Cards and conditional independence*

We remove a red 10 and a red 6 from a standard deck of playing cards. We shuffle the remaining cards, and draw one card. Write $\mathcal{A}$ for the event that the card drawn is a 10, $\mathcal{B}$ for the event the card drawn is red, and $\mathcal{C}$ for the event that the card drawn is either a 10 or a 6. Show that $\mathcal{A}$ and $\mathcal{B}$ are not independent, but are conditionally independent conditioned on $\mathcal{C}$.

Solution: We have $P(\mathcal{A}) = 3/50$, $P(\mathcal{B}) = 24/50$, $P(\mathcal{A} \cap \mathcal{B}) = 1/50$, so

$$P(\mathcal{A}|\mathcal{B}) = \frac{1/50}{24/50} = \frac{1}{24} \neq P(\mathcal{A})$$

so $\mathcal{A}$ and $\mathcal{B}$ are not independent. We have also that $P(\mathcal{A}|\mathcal{C}) = 1/2$ and $P(\mathcal{B}|\mathcal{C}) = 2/6 = 1/3$. Now

$$P(\mathcal{A} \cap \mathcal{B}|\mathcal{C}) = 1/6 = P(\mathcal{A}|\mathcal{C})P(\mathcal{B}|\mathcal{C})$$

so $\mathcal{A}$ and $\mathcal{B}$ are conditionally independent conditioned on $\mathcal{C}$.

4.4.3 Some Common Fallacies

People quite often reason poorly about independent events. The most common problem is known as the **gambler's fallacy**. This occurs when you reason that the probability of an independent event has been changed by previous outcomes. For example, imagine I toss a coin that is known to be fair 20 times and get 20 heads. The probability that the next toss will result in a head has not changed at all — it is still 0.5 — but many people will believe that it has changed. At time of writing, Wikipedia has some fascinating stories about the gambler's fallacy which suggest that it's quite a common mistake. People may interpret, say, a run of 20 heads as evidence that either the coin isn't fair, or the tosses aren't independent.

It seems particularly easy to make mistakes in conditional probability. One important mistake is the **prosecutor's fallacy** (which has a name because it's so common). A prosecutor has evidence $\mathcal{E}$ against a suspect. Write $\mathcal{I}$ for the event that the suspect is innocent. Things get interesting when $P(\mathcal{E}|\mathcal{I})$ is small. The prosecutor argues, incorrectly, that the suspect must be guilty, because $P(\mathcal{E}|\mathcal{I})$ is so small. The argument is incorrect because $P(\mathcal{E}|\mathcal{I})$ is irrelevant to the issue. What matters is $P(\mathcal{I}|\mathcal{E})$, which is the probability you are innocent, given the evidence.

The distinction is very important, because $P(\mathcal{I}|\mathcal{E})$ could be big even if $P(\mathcal{E}|\mathcal{I})$ is small. In the expression

$$P(\mathcal{I}|\mathcal{E}) = \frac{P(\mathcal{E}|\mathcal{I})P(\mathcal{I})}{P(\mathcal{E})} = \frac{P(\mathcal{E}|\mathcal{I})P(\mathcal{I})}{(P(\mathcal{E}|\mathcal{I})P(\mathcal{I}) + P(\mathcal{E}|\mathcal{I}^c)(1 - P(\mathcal{I})))}$$

notice that if $P(\mathcal{I})$ is large or if $P(\mathcal{E}|\mathcal{I}^c)$ is much smaller than $P(\mathcal{E}|\mathcal{I})$, then $P(\mathcal{I}|\mathcal{E})$ could be close to one even if $P(\mathcal{E}|\mathcal{I})$ is small. You need to be careful about the prosecutor's fallacy, as the confusion is not confined to the criminal law.

This fallacy can be made even more mischievious. Assume the prosecuter incorrectly adopts a model that items of evidence are independent when they're not. Then this model could result in an estimate of $P(\mathcal{E}|\mathcal{I})$ that is much smaller than it should be. The prosecutor's fallacy has contributed to a variety of miscarriages of justice. One famous incident occurred in the UK, involving a mother, Sally Clark, who was convicted of murdering two of her children. Expert evidence by paediatrician Roy Meadow argued that the probability of both deaths resulting from Sudden Infant Death Syndrome was extremely small. Her first appeal cited, among other grounds, statistical error in the evidence (you should spot the prosecutors fallacy; others were involved, too). The appeals court rejected this appeal, calling the statistical point "a sideshow". This prompted a great deal of controversy, both in the public press and various professional journals, including a letter from the then president of the Royal Statistical Society to the Lord Chancellor, pointing out that "*statistical evidence ... (should be) ... presented only by appropriately qualified statistical experts*". A second appeal (on other grounds) followed, and was successful. The appelate judges specifically criticized the statistical evidence, although it was not a point of appeal. Clark never recovered from this horrific set of events and died in tragic circumstances shortly after the second appeal. Roy Meadow was then struck off the rolls for serious professional misconduct as an expert witness, a ruling he appealed successfully. You can find a more detailed account of this case, with pointers to important documents including the letter to the Lord Chancellor (which is well worth reading), at http://en.wikipedia.org/wiki/Roy_Meadow; there is further material on the prosecutors fallacy at http://en.wikipedia.org/wiki/Prosecutor%27s_fallacy.

Remember this: *You need to be careful reasoning about conditional probability and about independent events. These topics mislead intuition so regularly that some errors have names. Be very careful.*

4.4.4 Example: The Monty Hall Problem

Careless thinking about probability, particularly conditional probability, can cause wonderful confusion. The Monty Hall problem is a relatively simple exercise in conditional probability. Nonetheless, it has been the subject of extensive, lively, and often quite inaccurate correspondence in various national periodicals — it seems to catch the attention, which is why we describe it in some detail. The problem works like this: There are three doors. Behind one is a car. Behind each of the others is a goat. The car and goats are placed randomly and fairly, so that the probability that there is a car behind each door is the same. You will get the object that lies behind the door you choose at the end of the game. The goats are interchangeable, and, for reasons of your own, you would prefer the car to a goat.

The game goes as follows. You select a door. The host then opens a door and shows you a goat. You must now choose to either keep your door, or switch to the

other door. What should you do?

You cannot tell what to do *using the information provided*, by the following argument. Label the door you chose at the start of the game 1; the other doors 2 and 3. Write C_i for the event that the car lies behind door i. Write G_m for the event that a goat is revealed behind door m, where m is the number of the door where the goat was revealed (which could be 1, 2, or 3). You need to know $P(C_1|G_m)$. But

$$P(C_1|G_m) = \frac{P(G_m|C_1)P(C_1)}{P(G_m|C_1)P(C_1) + P(G_m|C_2)P(C_2) + P(G_m|C_3)P(C_3)}$$

and you do not know $P(G_m|C_1)$, $P(G_m|C_2)$, $P(G_m|C_3)$, because you don't know the rule by which the host chooses which door to open to reveal a goat. Different rules lead to quite different analyses.

There are several possible rules for the host to show a goat:

- **Rule 1:** choose a door uniformly at random.
- **Rule 2:** choose from the doors with goats behind them *that are not door 1* uniformly and at random.
- **Rule 3:** if the car is at 1, then choose 2; if at 2, choose 3; if at 3, choose 1.
- **Rule 4:** choose from the doors with goats behind them uniformly and at random.

We should keep track of the rules in the conditioning, so we write $P(G_m|C_1, r_1)$ for the conditional probability that a goat was revealed behind door m when the car is behind door 1, using rule 1 (and so on). This means we are interested in

$$P(C_1|G_m, r_n) = \frac{P(G_m|C_1, r_n)P(C_1)}{P(G_m|C_1, r_n)P(C_1) + P(G_m|C_2, r_n)P(C_2) + P(G_m|C_3, r_n)P(C_3)}.$$

Notice that each of these rules is consistent with your observations — what you saw could have occurred under any of these rules. You have to know which rule the host uses to proceed. You should be aware that in many of the discussions of this problem, people assume without comment that the host uses rule 2, then proceed with this assumption.

Worked example 4.33 *Monty Hall, rule one*

Assume the host uses rule one, and shows you a goat behind door two. What is $P(C_1|G_2, r_1)$?

Solution: To work this out, we need to know $P(G_2|C_1, r_1)$, $P(G_2|C_2, r_1)$ and $P(G_2|C_3, r_1)$. Now $P(G_2|C_2, r_1)$ must be zero, because the host could not reveal a goat behind door two if there was a car behind that door. Write O_2 for the event the host *chooses* to open door two, and B_2 for the event there happens to be a goat behind door two. These two events are independent — the host chose the door uniformly at random. We can compute

$$\begin{aligned} P(G_2|C_1, r_1) &= P(O_2 \cap B_2|C_1, r_1) \\ &= P(O_2|C_1, r_1)P(B_2|C_1, r_1) \\ &= (1/3)(1) \\ &= 1/3 \end{aligned}$$

where $P(B_2|C_1, r_1) = 1$ because we conditioned on the fact there was a car behind door one, so there is a goat behind each other door. This argument establishes $P(G_2|C_3, r_1) = 1/3$, too. So $P(C_1|G_2, r_1) = 1/2$ — the host showing you the goat does not motivate you to do anything, because if $P(C_1|G_2, r_1) = 1/2$, then $P(C_3|G_2, r_1) = 1/2$, too — there's nothing to choose between the two closed doors.

Worked example 4.34 *Monty Hall, rule two*

Assume the host uses rule two, and shows you a goat behind door two. What is $P(C_1|G_2, r_2)$?

Solution: To work this out, we need to know $P(G_2|C_1, r_2)$, $P(G_2|C_2, r_2)$ and $P(G_2|C_3, r_2)$. Now $P(G_2|C_2, r_2) = 0$, because the host chooses from doors with goats behind them. $P(G_2|C_1, r_2) = 1/2$, because the host chooses uniformly and at random from doors with goats behind them that are not door one; if the car is behind door one, there are two such doors. $P(G_2|C_3, r_2) = 1$, because there is only one door that (a) has a goat behind it and (b) isn't door one. Plug these numbers into the formula, to get $P(C_1|G_2, r_2) = 1/3$. This is the source of all the fuss. It says that, if you know the host is using rule two, you should switch doors if the host shows you a goat behind door two (because $P(C_3|G_2, r_2) = 2/3$).

Notice what is happening: if the car is behind door three, then the *only* choice of goat for the host is the goat behind two. So by choosing a door under rule two,

the host is signalling some information to you, which you can use. By using rule three, the host can tell you precisely where the car is (exercises).

Many people find the result of example 4.34 counterintuitive, and object (sometimes loudly, in newspaper columns, letters to the editor, etc.). One example that some people find helpful is an extreme case. Imagine that, instead of three doors, there are 1002. The host is using rule two, modified in the following way: open all but one of the doors that are not door one, choosing only doors that have goats behind them to open. You choose door one; the host opens 1000 doors — say, all but doors one and 1002. What would you do?

4.5 EXTRA WORKED EXAMPLES

4.5.1 Outcomes and Probability

Worked example 4.35 *Children*

A couple decides to have children until either (a) they have both a boy and a girl or (b) they have three children. What is the set of outcomes?

Solution: Write B for boy, G for girl, and write them in birth order; we have $\{BG, GB, BBG, BBB, GGB, GGG\}$.

Worked example 4.36 *Monty Hall (sigh!) with indistinguishable goats*

There are three boxes. There is a goat, a second goat, and a car. These are placed into the boxes at random. The goats are indistinguishable for our purposes; equivalently, we do not care about the difference between goats. What is the sample space?

Solution: Write G for goat, C for car. Then we have $\{CGG, GCG, GGC\}$.

Worked example 4.37 *Monty Hall with distinguishable goats*

There are three boxes. There is a goat, a second goat, and a car. These are placed into the boxes at random. One goat is male, the other female, and the distinction is important. What is the sample space?

Solution: Write M for male goat, F for female goat, C for car. Then we have $\{CFM, CMF, FCM, MCF, FMC, MFC\}$. Notice how the number of outcomes has increased, because we now care about the distinction between goats.

Worked example 4.38 *A poor choice of strategy for planning a family*

A couple decides to have children. As they know no mathematics, they decide to have children until a girl then a boy are born. What is the sample space? Does this strategy bound the number of children they could be planning to have?

Solution: Write B for boy, G for girl. The sample space looks like any string of B's and G's that (a) ends in GB and (b) does not contain any other GB. In regular expression notation, you can write such strings as B^*G^+B. There is a lower bound on the length of the string (two), but no upper bound. As a family planning strategy, this is unrealistic, but it serves to illustrate the point that sample spaces don't have to be finite to be tractable.

Worked example 4.39 *Find the Lady, with even probabilities*

Recall the problem of worked example **??**. Assume that the card that is chosen is chosen fairly — that is, each card is chosen with the same probability. What is the probability of turning up a Queen?

Solution: There are three outcomes, and each is chosen with the same probability, so the probability is 1/3.

Worked example 4.40 *Monty Hall, indistinguishable goats, even probabilities*

Recall the problem of worked example 4.37. Each outcome has the same probability. We choose to open the first box. With what probability will we find a goat (any goat)?

Solution: There are three outcomes, each has the same probability, and two give a goat, so 2/3

Worked example 4.41 *Monty Hall, yet again*

Each outcome has the same probability. We choose to open the first box. With what probability will we find the car?

Solution: There are three places the car could be, each has the same probability, so 1/3

Worked example 4.42 *Monty Hall, with distinct goats, again*

Each outcome has the same probability. We choose to open the first box. With what probability will we find a female goat?

Solution: Using the reasoning of the previous example, but substituting "female goat" for "car", 1/3. The point of this example is that the sample space matters. If you care about the gender of the goat, then it's important to keep track of it; if you don't, it's a good idea to omit it from the sample space.

4.5.2 Events

Worked example 4.43 *Drawing a red ten*

I shuffle a standard pack of cards, and draw one card. What is the probability that it is a red ten?

Solution: There are 52 cards, and each is an outcome. Two of these outcomes are red tens; so we have $2/52 = 1/26$.

Worked example 4.44 *Birthdays in succession*

We stop three people at random, and ask the day of the week on which they are born. What is the probability that they are born on three days of the week in succession (for example, the first on Monday; the second on Tuesday; the third on Wednesday; or Saturday-Sunday-Monday; and so on).

Solution: We assume that births are equally common on each day of the week. The space of outcomes consists of triples of days, and each outcome has the same probability. The event is the set of triples of three days in succession (which has seven elements, one for each starting day). The space of outcomes has 7^3 elements in it, so the probability is

$$\frac{\text{Number of outcomes in the event}}{\text{Total number of outcomes}} = \frac{7}{7^3} = \frac{1}{49}.$$

Worked example 4.45 *Shared birthdays*

We stop two people at random. What is the probability that they were born on the same day of the week?

Solution: The day the first person was born doesn't matter; the probability the second person was born on that day is 1/7. Or you could count outcomes explicitly to get

$$\frac{\text{Number of outcomes in the event}}{\text{Total number of outcomes}} = \frac{7}{7 \times 7} = \frac{1}{7}.$$

Worked example 4.46 *Children - 3*

This example is a version of of example 1.12, p44, in Stirzaker, "Elementary Probability". A couple decides to have children. They decide to have children until there is one of each gender, or until there are three, and then stop. Assume that each birth results in one child, and each gender is equally likely at each birth. Let $\mathcal{B}_i$ be the event that there are i boys, and $\mathcal{C}$ be the event there are more girls than boys. Compute $P(\mathcal{B}_1)$ and $P(\mathcal{C})$.

Solution: We could write the outcomes as $\{GB, BG, GGB, GGG, BBG, BBB\}$. Again, if we think about them like this, we have no simple way to compute their probability; so we use the sample space from the previous example with the device of the fictitious births again. The important events are $\{GBb, GBg\}$; $\{BGb, BGg\}$; $\{GGB\}$; $\{GGG\}$; $\{BBG\}$; and $\{BBB\}$. Like this, we get $P(\mathcal{B}_1) = 5/8$ and $P(\mathcal{C}) = 1/4$.

4.5.3 Independence

Worked example 4.47 *Children*

A couple decides to have two children. Genders are assigned to children at random, fairly, at birth and independently at each birth (our models have to abstract a little!). What is the probability of having a boy and then a girl?

Solution:

$$\begin{aligned} P(\text{first is boy} \cap \text{second is girl}) &= P(\text{first is boy})P(\text{second is girl}) \\ &= (1/2)(1/2) \\ &= 1/4 \end{aligned}$$

Worked example 4.48 *Programs*

We sample the processes on a computer at random intervals. Write $\mathcal{A}$ for the event that program A is observed to be running in a sample, $\mathcal{B}$ for the event that program B is observed to be running in a sample, and $\mathcal{N}$ for the (*N*asty) event that program C is observed to be behaving badly in a sample. We find $P(\mathcal{A} \cap \mathcal{N}) = 0.07$; $P(\mathcal{B} \cap \mathcal{N}) = 0.05$; $P(\mathcal{A} \cap \mathcal{B} \cap \mathcal{N}) = 0.04$; and $P(\mathcal{N}) = 0.1$. Are $\mathcal{A}$ and $\mathcal{B}$ conditionally independent conditioned on $\mathcal{N}$?

Solution: This is a straightforward calculation. You should get $P(\mathcal{A}|\mathcal{N}) = 0.7$; $P(\mathcal{B}|\mathcal{N}) = 0.5$; $P(\mathcal{A} \cap \mathcal{B}|\mathcal{N}) = 0.4$; and so $P(\mathcal{A} \cap \mathcal{B}|\mathcal{N}) \neq P(\mathcal{A}|\mathcal{N}) \times P(\mathcal{B}|\mathcal{N})$, and they are not conditionally independent – there is some form of interaction here.

Worked example 4.49 *Independent test results*

You have a blood test for a rare disease. We study the effect of repeated tests. Write $\mathcal{S}$ for the event that the patient is sick; $\mathcal{D}_i^+$ for the event that the i'th repetition of the test reports positive; and $\mathcal{D}_i^-$ for the event that the i'th repetition of the test reports negative. The test has $P(\mathcal{D}^+|\mathcal{S}) = 0.8$ and $P(\mathcal{D}^-|\overline{\mathcal{S}}) = 0.8$, and $P(\mathcal{S}) = 1e-5$. This blood test has the property that, if you repeat the test, results are conditionally independent conditioned on the true result, meaning that $P(\mathcal{D}_1^+ \cap \mathcal{D}_2^+|\overline{\mathcal{S}}) = P(\mathcal{D}_1^+|\overline{\mathcal{S}})P(\mathcal{D}_2^+|\overline{\mathcal{S}})$. Assume you test positive once; twice; and ten times. In each case, what is the posterior probability that you are sick?

Solution: I will work the case for two positive tests. We need $P(\mathcal{S}|\mathcal{D}_1^+ \cap \mathcal{D}_2^+)$. We have

$$\begin{aligned} P(\mathcal{S}|\mathcal{D}_1^+ \cap \mathcal{D}_2^+) &= \frac{P(\mathcal{D}_1^+ \cap \mathcal{D}_2^+|\mathcal{S})P(\mathcal{S})}{P(\mathcal{D}_1^+ \cap \mathcal{D}_2^+)} \\ &= \frac{P(\mathcal{D}_1^+ \cap \mathcal{D}_2^+|\mathcal{S})P(\mathcal{S})}{P(\mathcal{D}_1^+ \cap \mathcal{D}_2^+|\mathcal{S})P(\mathcal{S}) + P(\mathcal{D}_1^+ \cap \mathcal{D}_2^+|\overline{\mathcal{S}})P(\overline{\mathcal{S}})} \\ &= \frac{0.8 \times 0.8 \times 1e-5}{0.8 \times 0.8 \times 1e-5 + 0.2 \times 0.2 \times (1 - 1e-5)} \\ &\approx 1.6e-4. \end{aligned}$$

You should check that once yields a posterior of approximately 4e-5, and ten times yields a posterior of approximately .91. This isn't an argument for repeating tests; rather, you should regard it as an indication of how implausible the assumption of conditional independence of test results is.

4.5.4 Conditional Probability

Worked example 4.50 *Card games*

You have two decks of 52 standard playing cards. One has been shuffled properly. The other is organized as 26 black cards, then 26 red cards. You are shown one card from one deck, which turns out to be black; what is the posterior probability that you have a card from the shuffled deck?

Solution: Write $\mathcal{S}$ for the event the card comes from the shuffled deck, and $\mathcal{B}$ the event you are given a black card. We want

$$\begin{aligned} P(\mathcal{S}|\mathcal{B}) &= \frac{P(\mathcal{B}|\mathcal{S})P(\mathcal{S})}{P(\mathcal{B})} \\ &= \frac{P(\mathcal{B}|\mathcal{S})P(\mathcal{S})}{P(\mathcal{B}|\mathcal{S})P(\mathcal{S}) + P(\mathcal{B}|\overline{\mathcal{S}})P(\overline{\mathcal{S}})} \\ &= \frac{(1/2)\times(1/2)}{(1/2)\times(1/2) + 1\times(1/2)} \\ &= 1/3 \end{aligned}$$

Worked example 4.51 *Finding a common disease*

A disease occurs with probability 0.4 (i.e. it is present in 40% of the population). You have a test that detects the disease with probability 0.6, and produces a false positive with probability 0.1. What is the posterior probability you have the disease if the test comes back positive?

Solution: Write $\mathcal{S}$ for the event you are sick, and $\mathcal{P}$ for the event the test comes back positive. We want

$$\begin{aligned} P(\mathcal{S}|\mathcal{P}) &= \frac{P(\mathcal{P}|\mathcal{S})P(\mathcal{S})}{P(\mathcal{P})} \\ &= \frac{P(\mathcal{P}|\mathcal{S})P(\mathcal{S})}{P(\mathcal{P}|\mathcal{S})P(\mathcal{S}) + P(\mathcal{P}|\overline{\mathcal{S}})P(\overline{\mathcal{S}})} \\ &= \frac{0.6\times 0.4}{0.6\times 0.4 + 0.1\times 0.6} \\ &= 0.8 \end{aligned}$$

Notice that if the disease is quite common, even a rather weak test is helpful.

Worked example 4.52 *Which disease do you have?*

Disease A occurs with probability 0.1 (i.e. it is present in 20% of the population), and disease B occurs with probability 0.2. It is not possible to have both diseases. You have a single test. This test reports positive with probability 0.8 for a patient with disease A, with probability 0.5 for a patient with disease B, and with probability 0.01 for a patient with no disease. What is the posterior probability you have either disease, or neither, if the test comes back positive?

Solution: We are interested in $\mathcal{A}$ (the event you have disease A), $\mathcal{B}$ (the event you have disease B), and $\mathcal{W}$ (the event you are well). Write $\mathcal{P}$ for the event the test comes back positive. We want $P(\mathcal{A}|\mathcal{P})$, $P(\mathcal{B}|\mathcal{P})$ and $P(\mathcal{W}|\mathcal{P}) = 1 - P(\mathcal{A}|\mathcal{P}) - P(\mathcal{B}|\mathcal{P})$. We have

$$\begin{aligned} P(\mathcal{A}|\mathcal{P}) &= \frac{P(\mathcal{P}|\mathcal{A})P(\mathcal{A})}{P(\mathcal{P})} \\ &= \frac{P(\mathcal{P}|\mathcal{A})P(\mathcal{A})}{P(\mathcal{P}|\mathcal{A})P(\mathcal{A}) + P(\mathcal{P}|\mathcal{B})P(\mathcal{B}) + P(\mathcal{P}|\mathcal{W})P(\mathcal{W})} \\ &= \frac{0.8 \times 0.1}{0.8 \times 0.1 + 0.5 \times 0.2 + 0.01 \times 0.7} \\ &\approx 0.43 \end{aligned}$$

A similar calculation yields $P(\mathcal{B}|\mathcal{P})) \approx 0.53$ and $P(\mathcal{W}|\mathcal{P}) \approx 0.04$. The low probability of a false positive means that a positive result very likely comes from some disease. Even though the test isn't particularly sensitive to disease B, the fact B is twice as common as A means a positive result is somewhat more likely to have come from B than from A.

Worked example 4.53 *Fraud or psychic powers?*

You want to investigate the powers of a putative psychic. You blindfold this person, then flip a fair coin 10 times. Each time, the subject correctly tells you whether it came up heads or tails. There are three possible explanations: chance, fraud, or psychic powers. What is the posterior probability of each, conditioned on the evidence.

Solution: We have to do some modelling here. We must choose reasonable numbers for the prior of chance ($\mathcal{C}$), fraud ($\mathcal{F}$), and psychic powers ($\mathcal{P}$). There's little reliable evidence for psychic powers to date, so we can choose $P(\mathcal{P}) = 2\epsilon$ (where ϵ is a very small number), and allocate the remaining probability evenly between $\mathcal{C}$ and $\mathcal{F}$. Write $\mathcal{E}$ for the event the subject correctly calls 10 flips of a fair coin. We have $P(\mathcal{E}|\mathcal{C}) = (1/2)^{10}$. Assume that fraud and psychic powers are efficient, so that $P(\mathcal{E}|\mathcal{F}) = P(\mathcal{E}|\mathcal{P}) = 1$. Then we have

$$\begin{aligned} P(\mathcal{P}|\mathcal{E}) &= \frac{P(\mathcal{E}|\mathcal{P})P(\mathcal{P})}{P(\mathcal{E}|\mathcal{P})P(\mathcal{P}) + P(\mathcal{E}|\mathcal{C})P(\mathcal{C}) + P(\mathcal{E}|\mathcal{F})P(\mathcal{F})} \\ &= \frac{2\epsilon}{2\epsilon + (1/2)^{10} \times (0.5 - \epsilon) + (0.5 - \epsilon)} \\ &\approx 4\epsilon \end{aligned}$$

and $P(\mathcal{F}|\mathcal{E})$ is rather close to 1. I'd check how well the blindfold works; it's a traditional failure point in experiments like this.

4.6 YOU SHOULD

4.6.1 remember these definitions:

4.6.2 remember these terms:

4.6.3 remember these facts:

4.6.4 be able to:

- Write out a set of outcomes for an experiment.
- Construct an event space.
- Compute the probabilities of outcomes and events.
- Determine when events are independent.
- Compute the probabilities of outcomes by counting events, when the count is straightforward.
- Compute a conditional probability.

PROBLEMS

Outcomes

4.1. You roll a four sided die. What is the space of outcomes?

4.2. King Lear decides to allocate three provinces (1, 2, and 3) to his daughters (Goneril, Regan and Cordelia - read the book) at random. Each gets one province. What is the space of outcomes?

4.3. You randomly wave a flyswatter at a fly. What is the space of outcomes?

4.4. You read the book, so you know that King Lear had family problems. As a result, he decides to allocate two provinces to one daughter, one province to another daughter, and no provinces to the third. Because he's a bad problem solver, he does so at random. What is the space of outcomes?

The Probability of an Outcome

4.5. You roll a fair four sided die. What is the probability of getting a 3?

4.6. You shuffle a standard deck of playing cards and draw a card. What is the probability that this is the king of hearts?

4.7. A roulette wheel has 36 slots numbered 1-36, and two slots numbered zero. The croupier spins the wheel, and throws a ball onto the surface; the ball bounces around and ends up in a slot (which is chosen fairly and at random). What is the probability the ball ends up in slot 2?

Events

4.8. At a particular University, 1/2 of the students drink alcohol and 1/3 of the students smoke cigarettes.

(a) What is the largest possible fraction of students who do neither?

(b) It turns out that, in fact, 1/3 of the students do neither. What fraction of the students does both?

The Probability of Events

4.9. You flip a fair coin three times. What is the probability of seeing HTH? (i.e. Heads, then Tails, then Heads)

4.10. You flip a fair coin three times. What is the probability of seeing two heads and one tail?

4.11. You remove the king of hearts from a standard deck of cards, then shuffle it and draw a card.

(a) What is the probability this card is a king?

(b) What is the probability this card is a heart?

4.12. You shuffle a standard deck of cards, then draw four cards.

(a) What is the probability all four are the same suit?

(b) What is the probability all four are red?

(c) What is the probability each has a different suit?

4.13. You roll three fair six-sided dice and add the numbers. What is the probability the result is even?

4.14. You roll three fair six-sided dice and add the numbers. What is the probability the result is even *and* not divisible by 20?

4.15. You shuffle a standard deck of cards, then draw seven cards. What is the probability that you see no aces?

4.16. Show that $P(\mathcal{A} - (\mathcal{B} \cup \mathcal{C})) = P(\mathcal{A}) - P(\mathcal{A} \cap \mathcal{B}) - P(\mathcal{A} \cap \mathcal{C}) + P(\mathcal{A} \cap \mathcal{B} \cap \mathcal{C})$.

4.17. You draw a single card from a standard 52 card deck. What is the probability that it is red?

4.18. You remove all heart cards from a standard 52 card deck, then draw a single card from the result. What is the probability that the card you draw is red?

Computing Probabilities by Counting Outcomes

4.19. Assume each outcome in Ω has the same probability. In this case, show

$$P(\mathcal{E}) = \frac{\text{Number of outcomes in } \mathcal{E}}{\text{Total number of outcomes in } \Omega}$$

4.20. You roll a fair four sided die, and then a fair six sided die. You add the numbers on the two dice. What is the probability the result is even?

4.21. You roll a fair 20 sided die. What is the probability of getting an even number?

4.22. You roll a fair five sided die. What is the probability of getting an even number?

4.23. *I am indebted to Amin Sadeghi for this exercise.* You must sort four balls into two buckets. There are two white, one red and one green ball.

(a) For each ball, you choose a bucket independently and at random, with probability $\frac{1}{2}$. Show that the probability each bucket has a colored ball in it is $\frac{1}{2}$.

(b) You now choose to sort these balls in such a way that each bucket has two balls in it. You can do so by generating a permutation of the balls uniformly and at random, then placing the first two balls in the first bucket and the second two balls in the second bucket. Show that there are 16 permutations where there is one colored ball in each bucket.

(c) Use the results of the previous step to show that, using the sorting procedure of that step, the probability of having a colored ball in each bucket is $\frac{2}{3}$.

(d) Why do the two sorting procedures give such different outcomes?

Permutations and Combinations

4.24. You shuffle a standard deck of playing cards, and deal a hand of 10 cards. With what probability does this hand have five red cards?

4.25. Magic the Gathering is a popular card game. Cards can be land cards, or other cards. We consider a game with two players. Each player has a deck of 40 cards. Each player shuffles their deck, then deals seven cards, called their *hand*.

(a) Assume that player one has 10 land cards in their deck and player two has 20. With what probability will *each* player have four lands in their hand?

(b) Assume that player one has 10 land cards in their deck and player two has 20. With what probability will player one have two lands and player two have three lands in hand?

(c) Assume that player one has 10 land cards in their deck and player two has 20. With what probability will player two have more lands in hand than player one?

4.26. The previous exercise divided Magic the Gathering cards into lands vs. other. We now recognize four kinds of cards: land, spell, creature and artifact. We consider a game with two players. Each player has a deck of 40 cards. Each player shuffles their deck, then deals seven cards, called their *hand*.

(a) Assume that player one has 10 land cards, 10 spell cards, 10 creature cards

and 10 artifact cards in their deck. With what probability will player one have at least one of each kind of card in hand?

(b) Assume that player two has 20 land cards, 5 spell cards, 7 creature cards and 8 artifact cards in their deck. With what probability will player two have at least one of each kind of card in hand?

(c) Assume that player one has 10 land cards, 10 spell cards, 10 creature cards and 10 artifact cards in their deck;. and player two has 20 land cards, 5 spell cards, 7 creature cards and 8 artifact cards in their deck. With what probability will at least one of the players have at least one of each kind card in hand?

Independence

4.27. Event $\mathcal{A}$ has $P(\mathcal{A}) = 0.5$. Event $\mathcal{B}$ has $P(\mathcal{B}) = 0.2$. We also know that $P(\mathcal{A} \cup \mathcal{B}) = 0.65$. Are A and B independent? Why?

4.28. You take a standard deck of cards, shuffle it, and remove both red kings. You then draw a card.

(a) Is the event {card is red} independent of the event {card is a queen}?

(b) Is the event {card is black} independent of the event {card is a king}?

4.29. You flip a fair coin seven times. What is the probability that you see three H's and two T's?

4.30. An airline sells T tickets for a flight with S seats, where $T > S$. Passengers turn up for the flight independently, and the probability that a passenger with a ticket will turn up for a flight is p_t. The pilot is eccentric, and will fly only if precisely E passengers turn up, where $E < S$. Write an expression for the probability the pilot will fly.

Conditional Probability

4.31. You roll two fair six-sided dice. What is the conditional probability the sum of numbers is greater than three, conditioned on the first die coming up even.

4.32. You take a standard deck of cards, shuffle it, and remove one card. You then draw a card.

(a) What is the conditional probability that the card you draw is a red king, conditioned on the removed card being a king?

(b) What is the conditional probability that the card you draw is a red king, conditioned on the removed card being a red king?

(c) What is the conditional probability that the card you draw is a red king, conditioned on the removed card being a black ace?

4.33. A royal flush is a hand of five cards, consisting of Ace, King, Queen, Jack and 10 of a single suit. Poker players like this hand, but don't see it all that often.

(a) You draw three cards from a standard deck of playing cards. These are Ace, King, Queen of hearts. What is the probability that the next two cards you draw will result in a getting a royal flush? (this is the conditional probability of getting a royal flush, conditioned on the first three cards being AKQ of hearts).

4.34. You roll a fair five-sided die, and a fair six-sided die.

(a) What is the probability that the sum of numbers is even?

(b) What is the conditional probability that the sum of numbers is even, conditioned on the six-sided die producing an odd number?

4.35. You take a standard deck of playing cards, shuffle it, and remove 13 cards

without looking at them. You then shuffle the resulting deck of 39 cards, and draw three cards. Each of these three cards is red. What is the conditional probability that every card you removed is black?

4.36. Magic the Gathering is a popular card game. Cards can be land cards, or other cards. We will consider a deck of 40 cards, containing 10 land cards and 30 other cards. A player shuffles that deck, and draws seven cards but does not look at them. The player then chooses one of these cards at random; it is a land.
- **(a)** What is the conditional probability that the original hand of seven cards is all lands?
- **(b)** What is the conditional probability that the original hand of seven cards contains only one land?

4.37. Magic the Gathering is a popular card game. Cards can be land cards, or other cards. We will consider a deck of 40 cards, containing 10 land cards and 30 other cards. A player shuffles that deck, and draws seven cards but does not look at them. The player then chooses three of these cards at random; each of these three is a land.
- **(a)** What is the conditional probability that the original hand of seven cards is all lands?
- **(b)** What is the conditional probability that the original hand of seven cards contains only three lands?

4.38. You take a standard deck of playing cards, and remove one card at random. You then draw a single card. Write $\mathcal{S}$ for the event that the card you remove is a six. Write $\mathcal{N}$ for the event that the card you remove is not a six. Write $\mathcal{R}$ for the event that the card you remove is red. Write $\mathcal{B}$ for the event the card you remove is black.
- **(a)** Write $\mathcal{A}$ for the event you draw a 6. What is $P(\mathcal{A}|\mathcal{S})$?
- **(b)** Write $\mathcal{A}$ for the event you draw a 6. What is $P(\mathcal{A}|\mathcal{N})$?
- **(c)** Write $\mathcal{A}$ for the event you draw a 6. What is $P(\mathcal{A})$?
- **(d)** Write $\mathcal{D}$ for the event you draw a red six. Are $\mathcal{D}$ and $\mathcal{A}$ independent? why?
- **(e)** Write $\mathcal{D}$ for the event you draw a red six. What is $P(\mathcal{D})$?

4.39. A student takes a multiple choice test. Each question has N answers. If the student knows the answer to a question, the student gives the right answer, and otherwise guesses uniformly and at random. The student knows the answer to 70% of the questions. Write $\mathcal{K}$ for the event a student knows the answer to a question and $\mathcal{R}$ for the event the student answers the question correctly.
- **(a)** What is $P(\mathcal{K})$?
- **(b)** What is $P(\mathcal{R}|\mathcal{K})$?
- **(c)** What is $P(\mathcal{K}|\mathcal{R})$, as a function of N?
- **(d)** What values of N will ensure that $P(\mathcal{K}|\mathcal{R}) > 99\%$?

The Monty Hall Problem

4.40. **Monty Hall, Rule 3:** If the host uses rule 3, then what is $P(C_1|G_2, r_3)$? Do this by computing conditional probabilities.

4.41. **Monty Hall, Rule 4:** If the host uses rule 4, and shows you a goat behind door 2, what is $P(C_1|G_2, r_4)$? Do this by computing conditional probabilities.

CHAPTER 5

Random Variables and Expectations

We have machinery to describe experiments with random outcomes, but we mostly care about numbers that are random. It is straightforward to link a number to the outcome of an experiment. The result is a random variable, a useful new idea. Random variables turn up in all sorts of places. For example, the amount of money you win or lose on a bet is a random variable. Now if you take the same bet repeatedly, you could wonder how much money will change hands in total, per bet. The answer to questions like this is the expected value of a random variable.

Expected values have strong properties. When one knows some expected values, you can bound various probabilities. This phenomenon parallels the property of data that we saw earlier – you don't find a large fraction of the dataset many standard deviations away from the mean. Particularly important to computer scientists (and gamblers!) is the weak law of large numbers. This law says, loosely, that the value, per bet, of repeating a bet many times will almost certainly be the expected value. Among other things, this law legitimizes estimating expectations and probabilities that are hard to calculate by using a simulation. This turns out to be really useful, because simulations are often quite easy programs to write.

5.1 RANDOM VARIABLES

Quite commonly, we would like to deal with numbers that are random. We can do so by linking numbers to the outcome of an experiment. We define a **random variable**:

Definition: 5.1 *Discrete random variable*

Given a sample space Ω, a set of events $\mathcal{F}$, a probability function P, and a countable set of of real numbers D, a discrete random variable is a function with domain Ω and range D.

This means that for any outcome ω there is a number $X(\omega)$. P will play an important role, but first we give some examples.

Example: 5.1 *Numbers from coins*

We flip a coin. Whenever the coin comes up heads, we report 1; when it comes up tails, we report 0. This is a random variable.

Example: 5.2 *Numbers from coins II*

We flip a coin 32 times. We record a 1 when it comes up heads, and when it comes up tails, we record a 0. This produces a 32 bit random number, which is a random variable.

Example: 5.3 *The number of pairs in a poker hand*

(from Stirzaker). We draw a hand of five cards. The number of pairs in this hand is a random variable, which takes the values $0, 1, 2$ (depending on which hand we draw)

A function that takes a discrete random variable to a set of numbers is also a discrete random variable.

Example: 5.4 *Parity of coin flips*

We flip a coin 32 times. We record a 1 when it comes up heads, and when it comes up tails, we record a 0. This produces a 32 bit random number, which is a random variable. The parity of this number is also a random variable.

Associated with any value x of the random variable X are a series of events. The most important is the set of outcomes ω such that $X(\omega) = x$, which we can write $\{\omega : X(\omega) = x\}$; it is usual to simplify to $\{X = x\}$, and we will do so. The probability that a random variable X takes the value x is given by $P(\{\omega : X(\omega) = x\})$, which is more usually written $P(\{X = x\})$. This is sometimes written as $P(X = x)$, and rather often written as $P(x)$.

We could also be interested in the set of outcomes ω such that $X(\omega) \leq x$ (i.e. in $\{\omega : X(\omega) \leq x\}$), which we will write $\{X \leq x\}$; The probability that X takes a value less than or equal to x is given by $P(\{\omega : X(\omega) \leq x\})$, which is more usually written $P(\{X \leq x\})$. Similarly, we could be interested in ω such that $\{X(\omega) > x\}$, and so on.

Definition: 5.2 *Probability distribution of a discrete random variable*

The probability distribution of a discrete random variable is the set of numbers $P(\{X = x\})$ for each value x that X can take. The distribution takes the value 0 at all other numbers. Notice that the distribution is non-negative. **Notation warning:** probability notation can be quirky. You may encounter $p(x)$ with the meaning "some probability distribution" or $p(x)$ meaning "the value of the probability distribution $P(\{X = x\})$ at the point x" or $p(x)$ with the meaning "the probability distribution $P(\{X = x\})$". Context may help disambiguate these uses.

Definition: 5.3 *Cumulative distribution of a discrete random variable*

The cumulative distribution of a discrete random variable is the set of numbers $P(\{X <= x\})$ for each value x that X can take. Notice that this is a non-decreasing function of x. **Notation warning:** Cumulative distributions are often written with an f, so that $f(x)$ might mean $P(\{X <= x\})$.

Worked example 5.1 *Numbers from coins III*

We flip a biased coin 2 times. The flips are independent. The coin has $P(H) = p$, $P(T) = 1 - p$. We record a 1 when it comes up heads, and when it comes up tails, we record a 0. This produces a 2 bit random number, which is a random variable taking the values 0, 1, 2, 3. What is the probability distribution and cumulative distribution of this random variable?

Solution: Probability distribution: $P(0) = (1-p)^2$; $P(1) = (1-p)p$; $P(2) = p(1-p)$; $P(3) = p^2$. Cumulative distribution: $f(0) = (1-p)^2$; $f(1) = (1-p)$; $f(2) = p(1-p) + (1-p) = (1-p^2)$; $f(3) = 1$.

Worked example 5.2 *Betting on coins*

One way to get a random variable is to think about the reward for a bet. We agree to play the following game. I flip a coin. The coin has $P(H) = p$, $P(T) = 1 - p$. If the coin comes up heads, you pay me q; if the coin comes up tails, I pay you r. The number of dollars that change hands is a random variable. What is its probability distribution?

Solution: We see this problem from my perspective. If the coin comes up heads, I get q; if it comes up tails, I get $-r$. So we have $P(X = q) = p$ and $P(X = -r) = (1 - p)$, and all other probabilities are zero.

5.1.1 Joint and Conditional Probability for Random Variables

All the concepts of probability that we described for events carry over to random variables. This is as it should be, because random variables are really just a way of getting numbers out of events. However, terminology and notation change a bit.

Definition: 5.4 *Joint probability distribution of two discrete random variables*

Assume we have two random variables X and Y. The probability that X takes the value x and Y takes the value y could be written as $P(\{X = x\} \cap \{Y = y\})$. It is more usual to write it as

$$P(x, y).$$

This is referred to as the **joint probability distribution** of the two random variables (or, quite commonly, the **joint**). You can think of this as a table of probabilities, one for each possible pair of x and y values.

We will simplify notation further. Usually, we are interested in random variables, rather than potentially arbitrary outcomes or sets of outcomes. We will write $P(X)$ to denote the probability distribution of a random variable, and $P(x)$ or $P(X = x)$ to denote the probability that that random variable takes a particular value. This means that, for example, the rule we could write as

$$P(\{X = x\} \,|\, \{Y = y\})P(\{Y = y\}) = P(\{X = x\} \cap \{Y = y\})$$

will be written as

$$P(x|y)P(y) = P(x, y).$$

Recall the rule from section 4.4.1:

$$P(\mathcal{A}|\mathcal{B}) = \frac{P(\mathcal{B}|\mathcal{A})P(\mathcal{A})}{P(\mathcal{B})}.$$

This rule can be rewritten in our notation for random variables. This is the most familiar form of **Bayes' rule**, which is important enough to appear in its own box.

Definition: 5.5 *Bayes' rule*

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

Random variables have another useful property. If $x_0 \neq x_1$, then the event $\{X = x_0\}$ must be disjoint from the event $\{X = x_1\}$. This means that

$$\sum_x P(x) = 1$$

and that, for any y,

$$\sum_x P(x|y) = 1$$

(if you're uncertain on either of these points, check them by writing them out in the language of events).

Now assume we have the joint probability distribution of two random variables, X and Y. Recall that we write $P(\{X = x\} \cap \{Y = y\})$ as $P(x, y)$. Now consider the sets of outcomes $\{Y = y\}$ for each different value of y. These sets must be disjoint, because y cannot take two values at the same time. Furthermore, each element of the set of outcomes $\{X = x\}$ must lie in one of the sets $\{Y = y\}$. So we have

$$\sum_y P(\{X = x\} \cap \{Y = y\}) = P(\{X = x\})$$

Definition: 5.6 *The marginal probability of a random variable*

Write $P(x, y)$ for the joint probability distribution of two random variables X and Y. Then

$$P(x) = \sum_y P(x, y) = \sum_y P(\{X = x\} \cap \{Y = y\}) = P(\{X = x\})$$

is referred to as the **marginal probability distribution** of X.

Definition: 5.7 *Independent random variables*

The random variables X and Y are **independent** if the events $\{X = x\}$ and $\{Y = y\}$ are independent. This means that

$$P(\{X = x\} \cap \{Y = y\}) = P(\{X = x\})P(\{Y = y\}),$$

which we can rewrite as

$$P(x, y) = P(x)P(y)$$

Worked example 5.3 *Sums and differences of dice*

You throw two dice. The number of spots on the first die is a random variable (call it X); so is the number of spots on the second die (Y). X and Y are independent. Now define $S = X + Y$ and $D = X - Y$. What is the probability distribution of S and of D?

Solution: S can have values in the range $2, \ldots, 12$. There is only one way to get a $S = 2$; two ways to get $S = 3$; and so on. Using the methods of chapter 4 for each case, the probabilities for $[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]$ are $[1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1]/36$. Similarly, D can have values in the range $-5, \ldots, 5$. Again, using the methods of chapter 13.1, the probabilities for $[-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5]$ are $[1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1]/36$.

Worked example 5.4 *Sums and differences of dice, II*

Using the terminology of example 5.3, what is the joint probability distribution of S and D?

Solution: This is more interesting to display, because it's an 11x11 table. Each entry of the table represents a pair of S, D values. Many pairs can't occur (for example, for $S = 2$, D can only be zero; if S is even, then D must be even; and so on). You can work out the table by checking each case; it's in Table 5.1.

$$\frac{1}{36} \times \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

TABLE 5.1: *A table of the joint probability distribution of S (vertical axis; scale $2, \ldots, 12$) and D (horizontal axis; scale $-5, \ldots, 5$) from example 5.4*

Worked example 5.5 *Sums and differences of dice, III*

Using the terminology of example 5.3, are X and Y independent? are S and D independent?

Solution: X and Y are clearly independent. But S and D are not. There are several ways to see this. One way is to notice that, if you know $S = 2$, then you know the value of D precisely; but if you know $S = 3$, D could be either 1 or -1. This means that $P(S|D)$ depends on D, so they're not independent. Another way is to notice that the rank of the table, as a matrix, is 6, which means that it can't be the outer product of two vectors.

Worked example 5.6 *Sums and differences of dice, IV*

Using the terminology of example 5.3, what is $P(S|D = 0)$? what is $P(D|S = 11)$?

Solution: You could work it out either of these from the table, or by first principles. If $D = 0$, S can have values $2, 4, 6, 8, 10, 12$, and each value has conditional probability $1/6$. If $S = 11$, D can have values 1, or -1, and each value has conditional probability $1/2$.

5.1.2 Just a Little Continuous Probability

Our random variables take values from a discrete set of numbers D. This makes the underlying machinery somewhat simpler to describe, and is often, but not

always, enough for model building. Some phenomena are more naturally modelled as being continuous — for example, human height; human weight; the mass of a distant star; and so on. Giving a complete formal description of probability on a continuous space is surprisingly tricky, and would involve us in issues that do not arise much in practice.

These issues are caused by two interrelated facts: real numbers have infinite precision; and you can't count real numbers. A continuous random variable is still a random variable, and comes with all the stuff that a random variable comes with. We will not speculate on what the underlying sample space is, nor on the underlying events. This can all be sorted out, but requires moderately heavy lifting that isn't particularly illuminating for us. The most interesting thing for us is specifying the probability distribution. Rather than talk about the probability that a real number takes a particular value (which we can't really do satisfactorily most of the time), we will instead talk about the probability that it lies in some interval. So we can specify a probability distribution for a continuous random variable by giving a set of (very small) intervals, and for each interval providing the probability that the random variable lies in this interval.

The easiest way to do this is to supply a **probability density function**. Let $p(x)$ be a probability density function for a continuous random variable X. We interpret this function by thinking in terms of small intervals. Assume that dx is an infinitesimally small interval. Then

$$p(x)dx = P(\{\text{event that } X \text{ takes a value in the range } [x, x + dx]\}).$$

Important properties of probability density functions follow from this definition.

Useful Facts: 5.1 *Properties of probability density functions*

- Probability density functions are non-negative. This follows from the definition; a negative value at some u would imply that $P(\{x \in [u, u + du]\})$ was negative, and this cannot occur.
- For $a < b$

$$P(\{X \text{ takes a value in the range } [a, b]\}) = \int_a^b p(x)dx.$$

  which we obtain by summing $p(x)dx$ over all the infinitesimal intervals between a and b.
- We must have that

$$\int_{-\infty}^{\infty} p(x)dx = 1.$$

  This is because

$$P(\{X \text{ takes a value in the range } [-\infty, \infty]\}) = 1 = \int_{-\infty}^{\infty} p(x)dx$$

- Probability density functions are usually called pdf's.
- It is quite usual to write all pdf's as lower-case p's. If one specifically wishes to refer to probability (as opposed to probability density), one writes an upper case P, as in the previous points.

The property that $\int_{-\infty}^{\infty} p(x)dx = 1$ is useful, because when we are trying to determine a probability density function, we can ignore a constant factor. So if $g(x)$ is a non-negative function that is proportional to the probability density function (often pdf) we are interested in, we can recover the pdf by computing

$$p(x) = \frac{1}{\int_{-\infty}^{\infty} g(x)dx} g(x).$$

This procedure is sometimes known as **normalizing**, and $\int_{-\infty}^{\infty} g(x)dx$ is the **normalizing constant**.

One good way to think about pdf's is as the limit of a histogram. Imagine you collect an arbitrarily large dataset of data items, each of which is independent. You build a histogram of that dataset, using arbitrarily narrow boxes. You scale the histogram so that the sum of the box areas is one. The result is a probability density function.

The pdf doesn't represent the probability that a random variable takes a value. Instead, you should think of $p(x)$ as being the limit of a ratio (which is why

it's called a density):

$$\frac{\text{the probability that the random variable will lie in a small interval centered on } x}{\text{the length of the small interval centered on } x}$$

Notice that, while a pdf has to be non-negative, and it has to integrate to 1, it does *not* have to be smaller than one. A ratio like this could be a lot larger than one, as long as it isn't larger than one for too many x (because the integral must be one). In fact, probability density functions can be moderately strange functions (exercises).

Worked example 5.7 *A probability density function that is larger than one*

Assume we have a physical system that can produce random numbers. It produces numbers in the range 0 to ϵ, where $\epsilon > 0$. Each number has the same probability of appearing. No number larger than ϵ or smaller than 0 can ever appear. What is the probability density function?

Solution: Write $p(x)$ for the probability density function. We must have that $p(x) = 0$ for $x < 0$ and $p(x) = 0$ for $x > \epsilon$. We must have that $p(x)$ is constant between 0 and ϵ and that $\int p(x)dx = 1$. So

$$p(x) = \begin{cases} 0 & \text{if } x < 0 \\ 0 & \text{if } x > \epsilon \\ \frac{1}{\epsilon} & \text{otherwise} \end{cases}$$

Notice that if $\epsilon < 1$, we have that $p(x) > 1$ for all x.

5.2 EXPECTATIONS AND EXPECTED VALUES

Example 5.2 described a simple game. I flip a coin. The coin has $P(H) = p$, $P(T) = 1 - p$. If the coin comes up heads, you pay me q; if the coin comes up tails, I pay you r. Now imagine we play this game many times. Our frequency definition of probability means that in N games, we expect to see about pN heads and $(1 - p)N$ tails. In turn, this means that my total income from these N games should be about $(pN)q - ((1 - p)N)r$. The N in this expression is inconvenient; instead, we could say that for any single game, my expected income is

$$pq - (1 - p)r.$$

This isn't the actual income from a single game (which would be either q or $-r$, depending on what the coin did). Instead, it's an estimate of what would happen over a large number of games, on a per-game basis. This is an example of an expected value.

5.2.1 Expected Values

Definition: 5.8 *Expected value*

Given a discrete random variable X which takes values in the set $\mathcal{D}$ and which has probability distribution P, we define the expected value

$$\mathbb{E}[X] = \sum_{x \in \mathcal{D}} xP(X = x).$$

This is sometimes written $\mathbb{E}_P[X]$, to clarify which distribution one has in mind

Notice that an expected value could take a value that the random variable doesn't take.

Example: 5.5 *Betting on coins*

We agree to play the following game. I flip a fair coin (i.e. $P(H) = P(T) = 1/2$). If the coin comes up heads, you pay me 1; if the coin comes up tails, I pay you 1. The expected value of my income is 0, even though the random variable never takes that value.

Worked example 5.8 *Betting on coins, again*

We agree to play the following game. I flip a fair coin (i.e. $P(H) = P(T) = 1/2$). If the coin comes up heads, you pay me 2; if the coin comes up tails, I pay you 1. What is the expected value of this game?

Solution: The expected value of my income is

$$(\frac{1}{2}) \times 2 - (\frac{1}{2}) \times 1 = \frac{1}{2}.$$

Notice this isn't even an integer, and there's no way that any one instance of the game would yield a payoff of 1/2. But this is what I would get, per game, if I played many times.

Your intuition is likely to tell you that the game of example 5.8 is good for me and bad for you. This intuition is correct. It turns out that an even stronger statement is possible: playing this game repeatedly is pretty much guaranteed to be excellent for me and disastrous for you. It'll take some pages before I can be crisp about precisely what I mean here and why it is true.

Definition: 5.9 *Expectation*

Assume we have a function f that maps a discrete random variable X into a set of numbers $\mathcal{D}_f$. Then $f(X)$ is a discrete random variable, too, which we write F. The expected value of this random variable is written

$$\mathbb{E}[f] = \sum_{u \in \mathcal{D}_f} uP(F = u) = \sum_{x \in \mathcal{D}} f(x)P(X = x)$$

which is sometimes referred to as "the expectation of f". The process of computing an expected value is sometimes referred to as "taking expectations".

We can compute expectations for continuous random variables, too, though summing over all values now turns into an integral. Assume I have a continuous random variable X with probability density function $p(x)$. Remember I interpret the probability density function as meaning that, for an infinitesimal interval size dx, $p(x)dx = P(\{X \in [x, x + dx]\}))$. Divide the set of possible values that X can take into small intervals of width Δx, centered on x_i. We can construct a discrete random variable $\hat{X}$ which takes values x_i. We have that $P(\{\hat{X} = x_i\}) \approx p(x_i)\Delta x$, where I used the approximation sign because Δx may not be infinitesimally small.

Now write $\mathbb{E}\left[\hat{X}\right]$ for the expected value of $\hat{X}$. We have

$$\mathbb{E}\left[\hat{X}\right] = \sum_{x_i} x_i P(x_i) \approx \sum_{x_i} x_i p(x_i)\Delta x.$$

As the intervals limit to infinitesimal intervals, $\hat{X}$ limits to X (think of a picture of a histogram with infinitely narrow boxes). Then $\mathbb{E}\left[\hat{X}\right]$ has a limit which is an integral, and this defines the expected value. So we have the expressions in the boxes below.

Definition: 5.10 *Expected value of a continuous random variable*

Given a continuous random variable X which takes values in the set $\mathcal{D}$ and which has probability distribution P, we define the expected value

$$\mathbb{E}[X] = \int_{x \in \mathcal{D}} xp(x)dx.$$

This is sometimes written $\mathbb{E}_p[X]$, to clarify which distribution one has in mind.

The expected value of a continuous random variable could be a value that the random variable doesn't take, too. Notice one attractive feature of the $\mathbb{E}[X]$ notation; we don't need to make any commitment to whether X is a discrete random variable (where we would write a sum) or a continuous random variable (where we would write an integral). The reasoning by which we turned a sum into an integral works for functions of continuous random variables, too.

Definition: 5.11 *Expectation of a continuous random variable*

Assume we have a function f that maps a continuous random variable X into a set of numbers $\mathcal{D}_f$. Then $f(X)$ is a continuous random variable, too, which we write F. The expected value of this random variable is

$$\mathbb{E}[f] = \int_{x \in \mathcal{D}} f(x)p(x)dx$$

which is sometimes referred to as "the expectation of f". The process of computing an expected value is sometimes referred to as "taking expectations".

Under some circumstances the expected value may not exist. The integral needs to exist, and be finite, for us to interpret the expected value meaningfully, and that isn't guaranteed for every continuous random variable. Nothing we do will encounter this issue, and so we will ignore it.

You can see an expectation as an operation you apply to a random variable. It doesn't matter whether the random variable is discrete or continuous; that just changes the recipe for computing the value of the expectation. The crucial property of this operation is that it is linear; this is so important I have put it in its own box.

Useful Facts: 5.2 *Expectations are linear*

Write f, g for functions of random variables.

- $\mathbb{E}[0] = 0$
- for any constant k, $\mathbb{E}[kf] = k\mathbb{E}[f]$
- $\mathbb{E}[f + g] = \mathbb{E}[f] + \mathbb{E}[g]$.

I have written this box in a rather compact form. This is because the expression $\mathbb{E}[X]$ for the expected value of a random variable is actually a special case of $\mathbb{E}[f]$ — one just uses the identity function for f. So the box also tells us that $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$, and so on.

5.2.2 Mean, Variance and Covariance

There are three very important expectations with special names.

Definition: 5.12 *Mean or expected value*

The mean or expected value of a random variable X is

$$\mathbb{E}[X]$$

Definition: 5.13 *Variance*

The variance of a random variable X is

$$\mathsf{var}[X] = \mathbb{E}\big[(X - \mathbb{E}[X])^2\big]$$

Definition: 5.14 *Covariance*

The covariance of two random variables X and Y is

$$\mathsf{cov}\,(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$

Worked example 5.9 *Mean of a coin flip*

We flip a biased coin, with $P(H) = p$. The random variable X has value 1 if the coin comes up heads, 0 otherwise. What is the mean of X? (i.e. $\mathbb{E}[X]$).

Solution: $\mathbb{E}[X] = \sum_{x \in D} xP(X = x) = 1p + 0(1 - p) = p$

Worked example 5.10 *Variance of a coin flip*

We flip a biased coin, with $P(H) = p$. The random variable X has value 1 if the coin comes up heads, 0 otherwise. What is the variance of X? (i.e. $\mathsf{var}[X]$).

Solution: $\mathsf{var}[X] = \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 = (1p - 0(1-p)) - p^2 = p(1-p)$

Worked example 5.11 *Variance*

Can a random variable have $\mathbb{E}[X] > \sqrt{\mathbb{E}[X^2]}$?

Solution: No, because that would mean that $\mathbb{E}\left[(X - \mathbb{E}[X])^2\right] < 0$. But this is the expected value of a non-negative quantity; it must be non-negative.

Worked example 5.12 *More variance*

We just saw that a random variable can't have $\mathbb{E}[X] > \sqrt{\mathbb{E}[X^2]}$. But I can easily have a random variable with large mean and small variance - isn't this a contradiction?

Solution: No, you're confused. Your question means you think that the variance of X is given by $\mathbb{E}\left[X^2\right]$; but actually $\mathsf{var}[X] = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2$

Useful Facts: 5.3 *Properties of variance*

1. For any constant k, $\mathsf{var}[k] = 0$
2. $\mathsf{var}[X] \geq 0$
3. $\mathsf{var}[kX] = k^2\mathsf{var}[X]$
4. if X and Y are independent, then $\mathsf{var}[X + Y] = \mathsf{var}[X] + \mathsf{var}[Y]$
5. $\mathsf{var}[X] = \mathsf{cov}(X, X)$.

1, 2, and 5 are obvious. You will prove 3 and 4 in the exercises.

Useful Facts: 5.4 *A useful expression for variance*

$$\begin{aligned}\mathsf{var}[X] &= \mathbb{E}\left[(X-\mathbb{E}[X])^2\right] \\ &= \mathbb{E}\left[(X^2 - 2X\mathbb{E}[X] + \mathbb{E}[X]^2)\right] \\ &= \mathbb{E}\left[X^2\right] - 2\mathbb{E}[X]\mathbb{E}[X] + \mathbb{E}[X]^2 \\ &= \mathbb{E}\left[X^2\right] - (\mathbb{E}[X])^2\end{aligned}$$

Useful Facts: 5.5 *A useful expression for covariance*

$$\begin{aligned}\mathsf{cov}(X,Y) &= \mathbb{E}[(X-\mathbb{E}[X])(Y-\mathbb{E}[Y])] \\ &= \mathbb{E}[(XY - Y\mathbb{E}[X] - X\mathbb{E}[Y] + \mathbb{E}[X]\mathbb{E}[Y])] \\ &= \mathbb{E}[XY] - 2\mathbb{E}[Y]\mathbb{E}[X] + \mathbb{E}[X]\mathbb{E}[Y] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y].\end{aligned}$$

Now assume that we have a probability distribution $P(X)$ defined on some discrete set of numbers. There is some random variable that produced this probability distribution. This means that we could talk about the mean of a probability distribution P (rather than the mean of a random variable whose probability distribution is $P(X)$). It is quite usual to talk about the mean of a probability distribution. Furthermore, we could talk about the variance of a probability distribution P (rather than the variance of a random variable whose probability distribution is $P(X)$).

Useful Facts: 5.6 *Independent random variables have zero covariance*

1. if X and Y are independent, then $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$.
2. if X and Y are independent, then $\mathsf{cov}(X,Y) = 0$.

If 1 is true, then 2 is obviously true (apply the expression of useful facts 5.5). I prove 5 below.

Proposition: *If X and Y are independent random variables, then $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$.*

Proof: Recall that $\mathbb{E}[X] = \sum_{x \in D} xP(X = x)$, so that

$$\begin{aligned}
\mathbb{E}[XY] &= \sum_{(x,y) \in D_x \times D_y} xyP(X = x, Y = y) \\
&= \sum_{x \in D_x} \sum_{y \in D_y} (xyP(X = x, Y = y)) \\
&= \sum_{x \in D_x} \sum_{y \in D_y} (xyP(X = x)P(Y = y)) \\
&\quad \text{because } X \text{ and } Y \text{ are independent} \\
&= \sum_{x \in D_x} \sum_{y \in D_y} (xP(X = x))\,(yP(Y = y)) \\
&= \left(\sum_{x \in D_x} xP(X = x) \right) \left(\sum_{y \in D_y} yP(Y = y) \right) \\
&= (\mathbb{E}[X])(\mathbb{E}[Y]).
\end{aligned}$$

This is certainly not true when X and Y are not independent (try $Y = -X$).

The variance of a random variable is often inconvenient, because its units are the square of the units of the random variable. Instead, we could use the **standard deviation**.

Definition: 5.15 *Standard deviation*

The **standard deviation** of a random variable X is defined as

$$\mathsf{std}\left(\{X\}\right) = \sqrt{\mathsf{var}[X]}$$

You do need to be careful with standard deviations. If X and Y are independent random variables, then $\mathsf{var}[X + Y] = \mathsf{var}[X] + \mathsf{var}[Y]$, but $\mathsf{std}\left(\{X + Y\}\right) = \sqrt{\mathsf{std}\left(\{X\}\right)^2 + \mathsf{std}\left(\{Y\}\right)^2}$. One way to avoid getting mixed up is to remember that

variances add, and derive expressions for standard deviations from that.

5.2.3 Expectations and Statistics

You should have noticed we now have two notions each for mean, variance, covariance, and standard deviation. One, which we expounded in section 2.3, describes datasets. We will call these **descriptive statistics**. The other, described above, is a property of probability distributions. We will call these **expectations**. In each case, the reason we have one name for two notions is that the notions are not really all that different.

Imagine we have a dataset $\{x\}$ of N items, where the i'th item is x_i. We can build a probability distribution out of this dataset, by placing a probability on each data item. We will give each data item the same probability (which must be $1/N$, so all probabilities add to 1). Write $\mathbb{E}[x]$ for the mean of this distribution. We have

$$\mathbb{E}[x] = \sum_i x_i p(x_i) = \frac{1}{N} \sum_i x_i = \text{mean}\,(\{x\}).$$

The variances, standard deviations and covariance have the same property: For this particular distribution (sometimes called the **empirical distribution**), the expectations have the same value as the descriptive statistics (exercises).

In section 5.3.4, we will see a form of converse to this fact. Imagine we have a dataset that consists of independent, identically distributed samples from a probability distribution (i.e. we know that each data item was obtained independently from the distribution). For example, we might have a count of heads in each of a number of coin flip experiments. Then the descriptive statistics will turn out to be accurate estimates of the expectations.

5.3 THE WEAK LAW OF LARGE NUMBERS

Assume you see repeated values of a random variable. For example, let X be the random variable which has value 1 if a coin comes up heads (which happens with probability p) and -1 if it comes up tails. You now actually flip a coin N times, recording 1 for heads and -1 for tails. Intuition should say that the average of these numbers should be a good estimate of the value of $\mathbb{E}[X]$, by the following argument. You should see 1 about pN times, and -1 about $(1-p)N$ times. So the average should be close to $p-(1-p)$, which is $\mathbb{E}[X]$. Furthermore, intuition should suggest that this estimate gets better as the number of flips goes up.

It turns out that these intuitions are correct, in a very general way. is a general property of expectations, that you can estimate them very accurately by experiment. This is extremely useful, because it means that you can use quite simple programs to estimate values that might take a great deal of work to obtain any other way. Most people find it natural that something of this sort should be true. What is neat is that it is quite easy to prove.

5.3.1 IID Samples

We need first to be crisp about what we are averaging. Imagine a random variable X, obtained by flipping a fair coin and reporting 1 for an H and -1 for a T. We can

talk about the probability distribution $P(X)$ of this random variable; we can talk about the expected value that the random variable takes; but the random variable itself doesn't have a value. However, if we actually flip a coin, we get either a 1 or a -1. Observing a value is sometimes called a **trial**. The resulting value is often called a **sample** of the random variable (or of its probability distribution). So flipping the coin is a trial, and the number you get is a sample. If we flipped a coin many times, we'd have a set of numbers (or samples). These numbers would be independent. Their histogram would look like $P(X)$. Collections of data items like this are important enough to have their own name.

Assume we have a set of data items x_i such that (a) they are independent; and (b) the histogram of a very large set of data items looks increasingly like the probability distribution $P(X)$ as the number of data items increases. Then we refer to these data items as **independent identically distributed samples** of $P(X)$; for short, **iid samples** or even just **samples**. It's worth knowing that it can be a difficult computational problem to get IID samples from some given probability distribution. For all of the cases we will deal with, it will be obvious how to get IID samples. Usually, they were generated for us — i.e. somebody flipped the coin, etc.

Now assume you take N IID samples, and average them. The weak law of large numbers states that, as N gets larger, this average is an increasingly good estimate of $\mathbb{E}[X]$. This fact allows us estimate expectations (and so probabilities) by simulation. Furthermore, it will allow us to make strong statements about how repeated games with random outcomes will behave. Finally, it will allow us to build a theory of decision making.

5.3.2 Two Inequalities

To go further, we need two useful inequalities. Consider

$$\mathbb{E}[|X|] = \sum_{x \in D} |x| P(\{X = x\}).$$

Now notice that all the terms in the sum are non-negative. Then the only way to have a small value of $\mathbb{E}[|X|]$ is to be sure that, when $|x|$ is large, $P(\{X = x\})$ is small. It turns out to be possible (and useful!) to be more crisp about how quickly $P(\{X = x\})$ falls as $|x|$ grows, resulting in Markov's inequality (which I'll prove below)

Definition: 5.16 *Markov's inequality*

Markov's inequality is

$$P(\{|X| \geq a\}) \leq \frac{\mathbb{E}[|X|]}{a}.$$

Notice that we've seen something like this before (the result about standard deviation in section 10 has this form). The reason this is worth proving is that it leads to a second result, and that gives us the weak law of large numbers. It should seem clear that the probability of a random variable taking a particular value must fall off rather fast as that value moves away from the mean, in units scaled to the standard deviation. This is because values of a random variable that are many standard deviations above the mean must have low probability, otherwise the values would occur more often and so the standard deviation would be bigger. This result is Chebyshev's inequality, which I shall also prove below.

Definition: 5.17 *Chebyshev's inequality*

Chebyshev's inequality is

$$P(\{|X - \mathbb{E}[X]| \geq a\}) \leq \frac{\mathsf{var}[X]}{a^2}.$$

It is common to see this in another form, obtained by writing σ for the standard deviation of X, substituting $k\sigma$ for a, and rearranging

$$P(\{|X - \mathbb{E}[X]| \geq k\sigma\}) \leq \frac{1}{k^2}$$

We care about Chebyshev's inequality because it gives us the weak law of large numbers.

5.3.3 Proving the Inequalities

An **indicator function** is a function that is one when some condition is true, and zero otherwise. The reason indicator functions are useful is that their expectated values have interesting properties.

Definition: 5.18 *Indicator functions*

An indicator function for an event is a function that takes the value zero for values of X where the event does not occur, and one where the event occurs. For the event $\mathcal{E}$, we write

$$\mathbb{I}_{[\mathcal{E}]}(X)$$

for the relevant indicator function.

For example,

$$\mathbb{I}_{[\{|X|\}\leq a]}(X) = \begin{cases} 1 & \text{if } -a < X < a \\ 0 & \text{otherwise} \end{cases}$$

Indicator functions have one useful property.

$$\mathbb{E}\big[\mathbb{I}_{[\mathcal{E}]}\big] = P(\mathcal{E})$$

which you can establish by checking the definition of expectations.

Proposition: *Markov's inequality: for X a random variable, $a > 0$,*

$$P(\{|X| \geq a\}) \leq \frac{\mathbb{E}[|X|]}{a}.$$

Proof: (from Wikipedia). Notice that, for $a > 0$,

$$a\mathbb{I}_{[\{|X|\geq a\}]}(X) \leq |X|$$

(because if $|X| \geq a$, the LHS is a; otherwise it is zero). Now we have

$$\mathbb{E}\big[a\mathbb{I}_{[\{|X|\geq a\}]}\big] \leq \mathbb{E}[|X|]$$

but, because expectations are linear, we have

$$\mathbb{E}\big[a\mathbb{I}_{[\{|X|\geq a\}]}\big] = a\mathbb{E}\big[\mathbb{I}_{[\{|X|\geq a\}]}\big] = aP(\{|X| \geq a\})$$

and so we have

$$aP(\{|X| \geq a\}) \leq \mathbb{E}[|X|]$$

and we get the inequality by division, which we can do because $a > 0$.

Proposition: *Chebyshev's inequality: for X a random variable, $a > 0$,*

$$P(\{|X - \mathbb{E}[X]| \geq a\}) \leq \frac{\mathsf{var}[X]}{a^2}.$$

Proof: Write U for the random variable $(X - \mathbb{E}[X])^2$. Markov's inequality gives us

$$P(\{|U| \geq w\}) \leq \frac{\mathbb{E}[|U|]}{w}$$

Now notice that, if $a^2 = w$,

$$P(\{|U| \geq w\}) = P(\{|X - \mathbb{E}[X]|\} \geq a)$$

so we have

$$P(\{|U| \geq w\}) = P(\{|X - \mathbb{E}[X]|\} \geq a) \leq \frac{\mathbb{E}[|U|]}{w} = \frac{\mathsf{var}[X]}{a^2}$$

5.3.4 The Weak Law of Large Numbers

Assume we have a set of N IID samples x_i of a probability distribution $P(X)$. Write

$$X_N = \frac{\sum_{i=1}^{N} x_i}{N}.$$

Now X_N is a random variable (the x_i are IID samples, and for a different set of samples you will get a different, random, X_N). Notice that $P(X = x_1, X = x_2, \ldots, X = x_n) = P(X = x_1)P(X = x_2) \ldots P(X = x_n)$, because the samples are independent and each is a sample of $P(X)$. This means that

$$\mathbb{E}[X_N] = \mathbb{E}[X]$$

because

$$\mathbb{E}[X_N] = \left(\frac{1}{N}\right) \sum_{i=1}^{N} \mathbb{E}[X].$$

This means that

$$\frac{\sum_{i=1}^{N} x_i}{N}$$

should be an accurate estimate of $\mathbb{E}[X]$. The weak law of large numbers states that, as N gets large, the estimate becomes more accurate.

Definition: 5.19 *Weak Law of Large Numbers*

The **weak law of large numbers** states that, if $P(X)$ has finite variance, then for any positive number ϵ

$$\lim_{N\to\infty} P(\{|\, X_N - \mathbb{E}[X]\,| \geq \epsilon\}) = 0.$$

Equivalently, we have

$$\lim_{N\to\infty} P(\{|\, X_N - \mathbb{E}[X]\,| < \epsilon\}) = 1.$$

Each form means that, for a large enough set of IID samples, the average of the samples (i.e. X_N) will, with high probability, be very close to the expectation $\mathbb{E}[X]$.

Proposition: *Weak law of large numbers*

$$\lim_{N\to\infty} P(\{|X_N - \mathbb{E}[X]| \geq \epsilon\}) = 0.$$

Proof: Assume that $P(X)$ has finite variance; that is, $\mathsf{var}(\{X\}) = \sigma^2$. Choose $\epsilon > 0$. Now we have that

$$\begin{aligned}
\mathsf{var}(\{X_N\}) &= \mathsf{var}\left(\left\{\frac{\sum_{i=1}^N x_i}{N}\right\}\right) \\
&= (\frac{1}{N^2})\mathsf{var}\left(\left\{\sum_{i=1}^N x_i\right\}\right) \\
&= (\frac{1}{N^2})(N\sigma^2) \\
&\quad \text{because the } x_i \text{ are independent} \\
&= \frac{\sigma^2}{N}
\end{aligned}$$

and that

$$\mathbb{E}[X_N] = \mathbb{E}[X].$$

Now Chebyshev's inequality gives

$$P(\{|X_N - \mathbb{E}[X]| \geq \epsilon\}) \leq \frac{\sigma^2}{N\epsilon^2}$$

so

$$\lim_{N\to\infty} P(\{|X_N - \mathbb{E}[X]| \geq \epsilon\}) = \lim_{N\to\infty} \frac{\sigma^2}{N\epsilon^2} = 0.$$

Notice that

$$1 - P(\{|X_N - \mathbb{E}[X]| \geq \epsilon\}) = P(\{|X_N - \mathbb{E}[X]| < \epsilon\}) \geq 1 - \frac{\sigma^2}{N\epsilon^2}.$$

so that

$$\lim_{N\to\infty} P(\{|X_N - \mathbb{E}[X]| < \epsilon\}) = \lim_{N\to\infty} \left(1 - \frac{\sigma^2}{N\epsilon^2}\right) = 1.$$

The weak law of large numbers gives us a very valuable way of thinking about expectations. Assume we have a random variable X. Then the weak law says that, if you observe a large number of IID samples of this random variable, the average of the values you observe should be very close to $\mathbb{E}[X]$. This result is extremely powerful. The next section explores some applications. The weak law allows us

to estimate expectations (and so probabilities, which are expectations of indicator functions) by observing random behavior. The weak law can be used to build a theory of decision making.

5.4 USING THE WEAK LAW OF LARGE NUMBERS

5.4.1 Should you accept a bet?

We can't answer this as a moral question, but we can as a practical question, using expectations. Generally, a bet involves an agreement that amounts of money will change hands, depending on the outcome of an experiment. Mostly, you are interested in how much you get from the bet, so it is natural to give sums of money you receive a positive sign, and sums of money you pay out a negative sign. The weak law says that if you repeat a bet many times, you are increasingly likely to receive the expected value of the bet, per bet. Under this convention, the practical answer is easy: accept a bet enthusiastically if its expected value is positive, otherwise decline it. It is interesting to notice how poorly this advice describes actual human behavior.

In the hope that not every reader is familiar with roulette, I will sketch how the game works, because I can then use it for examples and exercises. A roulette wheel has a collection of slots. There are 36 slots numbered with the digits $1 \ldots 36$, and then one, two or even three slots numbered with zero. There are no other slots. A ball is thrown at the wheel when it is spinning, and it bounces around and eventually falls into a slot. If the wheel is properly balanced, the ball has the same probability of falling into each slot. The number of the slot the ball falls into is said to "come up". There are a variety of bets available.

Worked example 5.13 *Red or Black?*

On a roulette wheel, 18 of the numbers $1 \ldots 36$ are red, and the others are black. The zero (or zeros) are colorless. You can bet on (among other things) whether a red number or a black number comes up. If you bet 1 on red, and a red number comes up, you keep your stake and get 1; if a black number or a zero comes up, you get -1 (i.e. the house keeps your bet).

- On a wheel with one zero, what is the expected value of a 1 bet on red?
- On a wheel with two zeros, what is the expected value of a 1 bet on red?
- On a wheel with three zeros, what is the expected value of a 1 bet on red?

Solution: Write p_r for the probability a red number comes up. The expected value is $1 \times p_r + (-1)(1 - p_r)$ which is $2p_r - 1$.

- In this case, $p_r =$ (number of red numbers)/(total number of numbers) $= 18/37$. So the expected value is $-1/37$ (you lose about three cents each time you bet).
- In this case, $p_r = 18/38$. So the expected value is $-2/38 = -1/19$ (you lose slightly more than five cents each time you bet).
- In this case, $p_r = 18/39$. So the expected value is $-3/39 = -1/13$ (you lose slightly less than eight cents each time you bet).

Notice that in the roulette game, the money you lose will go to the house. So the expected value to the house is just the negative of the expected value to you. You might not play the wheel often, but the house plays the wheel very often when there are many players. The weak law means a house with many players can rely on receiving about three, five, or eight cents per bet depending on the number of zeros on the wheel. This is a partial explanation of why there are lots of roulette wheels, and usually free food nearby. Not all bets are like this, though.

Worked example 5.14 *Coin game*

In this game, P1 flips a fair coin and P2 calls "H" or "T". If P2 calls right, then P1 throws the coin into the river; otherwise, P1 keeps the coin. What is the expected value of this game to P2? and to P1?

Solution: To P2, which we do first, because it's easiest: P2 gets 0 if P2 calls right, and 0 if P2 calls wrong; these are the only cases, so the expected value is 0. To P1: P1 gets -1 if P2 calls right, and 0 if P1 calls wrong. The coin is fair, so the probability P2 calls right is $1/2$. The expected value is $-1/2$. While I can't explain why people would play such a game, I've actually seen this done.

We call a bet **fair** when its expected value is zero. Taking a bet with a negative expected value is unwise, because, on average, you will lose money. Worse, the more times you play, the more you lose. Similarly, repeatedly taking a bet with a positive expected value is reliably profitable. However, you do need to be careful you computed the expected value right.

Worked example 5.15 *Birthdays in succession*

P1 and P2 agree to the following bet. P1 gives P2 a stake of 1. If three people, stopped at random on the street, have birthdays in succession (i.e. Mon-Tue-Wed, and so on), then P2 gives P1 100. Otherwise, P1 loses the stake. What is the expected value of this bet to P1?

Solution: Write p for the probability of winning. Then the expected value is $p \times 100 - (1-p) \times 1$. We computed p in example 4.44 (it was $1/49$). So the bet is worth $(52/49)$, or slightly more than a dollar, to P1. P1 should be happy to agree to this as often as possible.

The reason P2 agrees to bets like that of example 5.15 is most likely that P2 can't compute the probability exactly. P2 thinks the event is quite unlikely, so the expected value is negative; but it isn't as unlikely as P2 thought it was, and this is how P1 makes a profit. This is one of the many reasons you should be careful accepting a bet from a stranger: they might be able to compute better than you.

5.4.2 Odds, Expectations and Bookmaking — a Cultural Diversion

Gamblers sometimes use a terminology that is a bit different from ours. In particular, the term **odds** is important. The term comes from the following idea: P1 pays a bookmaker b (the stake) to make a bet; if the bet is successful, P1 receives a and the stake back, and if not, loses the original stake. This bet is referred to as odds of $a : b$ (read "odds of a to b").

Assume the bet is fair, so that the expected value is zero. Write p for the probability of winning. The net income to P1 is $ap - b(1-p)$. If this is zero, then

$p = b/(a + b)$. So you can interpret odds in terms of probability, *if* you assume the bet is fair.

A bookmaker sets odds at which to accept bets from gamblers. The bookmaker does not wish to lose money at this business, and so must set odds which are potentially profitable. Doing so is not simple (bookmakers can, and occasionally do, lose catastrophically, and go out of business). In the simplest case, assume that the bookmaker knows the probability p that a particular bet will win. Then the bookmaker could set odds of $(1 - p)/p : 1$. In this case, the expected value of the bet is zero; this is fair, but not attractive business, so the bookmaker will set odds assuming that the probability is a bit higher than it really is. There are other bookmakers out there, so there is some reason for the bookmaker to try to set odds that are close to fair.

In some cases, you can tell when you are dealing with a bookmaker who is likely to go out of business soon. For example, imagine there are two horses running in a race, both at $10 : 1$ odds — whatever happens, you could win by betting 1 on each. There is a more general version of this phenomenon. Assume the bet is placed on a horse race, and that bets pay off only for the winning horse. Assume also that exactly one horse will win (i.e. the race is never scratched, there aren't any ties, etc.), and write the probability that the i'th horse will win as p_i. Then

$$\sum_{i \in \text{horses}} p_i$$

must be 1. Now if the bookmaker's odds yield a set of probabilities that is less than 1, their business should fail, because there is at least one horse on which they are paying out too much. Bookmakers deal with this possibility by writing odds so that $\sum_{i \in \text{horses}} p_i$ is larger than one.

But this is not the only problem a bookmaker must deal with. The bookmaker doesn't actually know the probability that a particular horse will win, and must account for errors in this estimate. One way to do so is to collect as much information as possible (talk to grooms, jockeys, etc.). Another is to look at the pattern of bets that have been placed already. If the bookmaker and the gamblers agree on the probability that each horse will win, then there should be no expected advantage to choosing one horse over another — each should pay out slightly less than zero to the gambler (otherwise the bookmaker doesn't eat). But if the bookmaker has underestimated the probability that a particular horse will win, a gambler may get a positive expected payout by betting on that horse. This means that if one particular horse attracts a lot of money from bettors, it is wise for the bookmaker to offer less generous odds on that horse. There are two reasons: first, the bettors might know something the bookmaker doesn't, and they're signalling it; second, if the bets on this horse are very large and it wins, the bookmaker may not have enough capital left to pay out or to stay in business. All this means that real bookmaking is a complex, skilled business.

5.4.3 Ending a Game Early

Imagine two people are playing a game for a stake, but must stop early — who should get what percentage of the stake? One way to do this is to give each player

what they put in at the start, but this is (mildly) unfair if one has an advantage over the other. The alternative is to give each player the expected value of the game at that state for that player. Sometimes one can compute that expectation quite easily.

Worked example 5.16 *Ending a game early*

(from Durrett), two players each pay 25 to play the following game. They toss a fair coin. If it comes up heads, player H wins that toss; if tails, player T wins. The first player to reach 10 wins takes the stake of 50. But one player is called away when the state is 8-7 (H-T) — how should the stake be divided?

Solution: In this state, each player can either win — and so get 50 — or lose — and so get 0. The expectation for H is $50P(\{\text{H wins from 8-7}\}) + 0P(\{\text{T wins from 8-7}\})$, so we need to compute $P(\{\text{H wins from 8-7}\})$. Similarly, the expectation for T is $50P(\{\text{T wins from 8-7}\}) + 0P(\{\text{H wins from 8-7}\})$, so we need to compute $P(\{\text{T wins from 8-7}\})$; but $P(\{\text{T wins from 8-7}\}) = 1 - P(\{\text{H wins from 8-7}\})$. Now it is slightly easier to compute $P(\{\text{T wins from 8-7}\})$, because T can only win in two ways: 8-10 or 9-10. These are independent. For T to win 8-10, the next three flips must come up T, so that event has probability 1/8. For T to win 9-10, the next four flips must have one H in them, but the last flip may not be H (or else H wins); so the next four flips could be HTTT, THTT, or TTHT. The probability of this is 3/16. This means the total probability that T wins is 5/16. So T should get 15.625 and H should get the rest (although they might have to flip for the odd half cent).

5.4.4 Making a Decision with Decision Trees and Expectations

Imagine we have to choose an action. Once we have chosen, a sequence of random events occurs, and we get a reward with some probability. Which action should we choose? A good answer is to choose the action with the best expected outcome. If we encounter this situation repeatedly, the weak law tells us that choosing any other action than the one with best expected outcome is unwise. If we make a choice that is even only slightly worse than the best, we will reliably do worse than we could. This is a very common recipe, and it can be applied to many situations. Usually, but not always, the reward is in money, and we will compute with money rewards for the first few examples.

For such problems, it can be useful to draw a **decision tree**. A decision tree is a drawing of possible outcomes of decisions, which makes costs, benefits and random elements explicit. Each node of the tree represents a test of an attribute (which could be either a decision, or a random variable), and each edge represents a possible outcome of a test. The final outcomes are leaves. Usually, decision nodes are drawn as squares, chance elements as circles, and leaves as triangles.

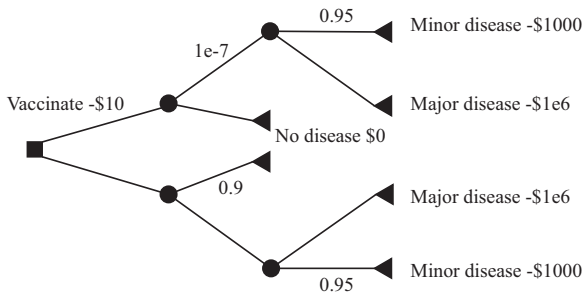

FIGURE 5.1: *A decision tree for the vaccination problem. The only decision is whether to vaccinate or not (the box at the root of the tree). I have only labelled edges where this is essential, so I did not annotate the "no vaccination" edge with zero cost. Once you decide whether to vaccinate or not, there is a random node (whether you get the disease or not), and, if you get it, another (minor or major).*

Worked example 5.17 *Vaccination*

It costs 10 to be vaccinated against a common disease. If you have the vaccination, the probability you will get the disease is $1e-7$. If you do not, the probability is 0.1. The disease is unpleasant; with probability 0.95, you will experience effects that cost you 1000 (eg several days in bed), but with probability 0.05, you will experience effects that cost you $1e6$. Should you be vaccinated?

Solution: Figure 5.1 shows a decision tree for this problem. I have annotated some edges with the choices represented, and some edges with probabilities; the sum of probabilities over all rightward (downgoing) edges leaving a random node is 1. It is straightforward to compute expectations. The expected cost of the disease is $0.95 \times 1000 + 0.05 \times 1e6 = 50,950$. If you are vaccinated, your expected income will be $-(10 + 1e-7 \times 50,950) \approx -10.01$. If you are not, your expected income is $-5,095$. You should be vaccinated.

Example 5.17 has interesting weaknesses. It's a rather shaky, though very common, use of the weak law. The weak law has nothing to say about the outcome of a decision that you make only once. The proper interpretation of the example is that, if you had to make the choice many times over, you should choose to be vaccinated. It's tough to use this model to argue that everyone should be vaccinated, even though there would then be a large number of decisions to average over. If everyone is vaccinated, then the probability of getting the disease will change.

Sometimes there is more than one decision. We can still do simple examples, though drawing a decision tree is now quite important, because it allows us to keep track of cases and avoid missing anything. For example, assume I wish to buy a cupboard. Two nearby towns have used furniture shops (usually called antique

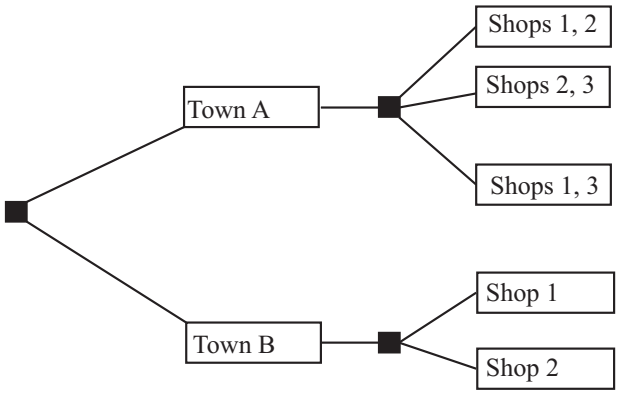

FIGURE 5.2: *The decision tree for the example of visiting furniture shops. Town A is nearer than town B, so if I go there I can choose to visit two of the three shops there; if I go to town B, I can visit only one of the two shops there. To decide what to do, I could fill in the probabilities and values of outcomes, compute the expected value of each pair of decisions, and choose the best. This could be tricky to do (where do I get the probabilities from?) but offers a rational and principled way to make the decision.*

shops these days). One is further away than the other. If I go to town A, I will have time to look in two (of three) shops; if I go to town B, I will have time to look in one (of two) shops. I could lay out this sequence of decisions (which town to go to; which shop to visit when I get there) as Figure 5.2.

You should notice that this figure is missing a lot of information. What is the probability that I will find what I'm looking for in the shops? What is the value of finding it? What is the cost of going to each town? and so on. This information is not always easy to obtain. In fact, I might simply need to give my best subjective guess of these numbers. Furthermore, particularly if there are several decisions, computing the expected value of each possible sequence could get difficult. There are some kinds of model where one can compute expected values easily, but a good viable hypothesis about why people don't make optimal decisions is that optimal decisions are actually too hard to compute.

5.4.5 Utility

Sometimes it is hard to work with money. For example, in the case of a serious disease, choosing treatments often boils down to expected survival times, rather than money.

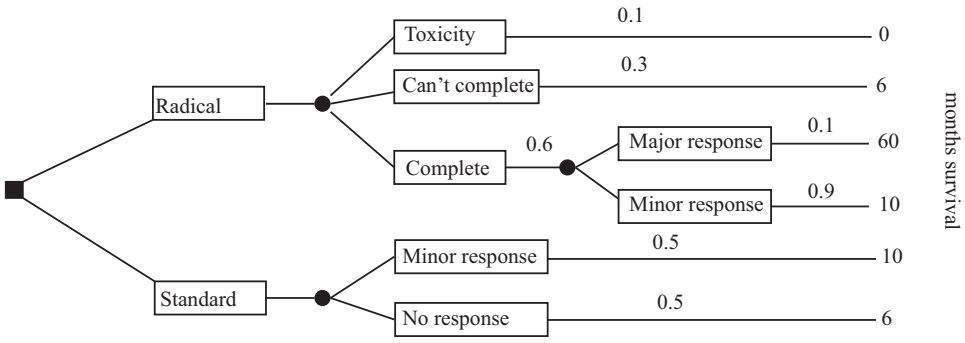

FIGURE 5.3: *A decision tree for example 5.18. Notations vary a bit, and here I have put boxes around the labels for the edges.*

Worked example 5.18 *Radical treatment*

(This example largely after Vickers, p97). Imagine you have a nasty disease. There are two kinds of treatment: standard, and radical. Radical treatment might kill you (with probability 0.1); might be so damaging that doctors stop (with probability 0.3); but otherwise you will complete the treatment. If you do complete radical treatment, there could be a major response (probability 0.1) or a minor response. If you follow standard treatment, there could be a major response (probability 0.5) or a minor response, but the outcomes are less good. All this is best summarized in a decision tree (Figure 5.3). What gives the longest expected survival time?

Solution: In this case, expected survival time with radical treatment is $(0.1 \times 0 + 0.3 \times 6 + 0.6 \times (0.1 \times 60 + 0.9 \times 10)) = 10.8$ months; expected survival time without radical treatment is $0.5 \times 10 + 0.5 \times 6 = 8$ months.

Working with money values is not always a good idea. For example, many people play state lotteries. The expected value of a 1 bet on a state lottery is well below 1 — why do people play? It's easy to assume that all players just can't do sums, but many players are well aware that the expected value of a bet is below the cost. It seems to be the case that people value money in a way that doesn't depend linearly on the amount of money. So, for example, people may value a million dollars rather more than a million times the value they place on one dollar. If this is true, we need some other way to keep track of value; this is sometimes called **utility**. It turns out to be quite hard to know how people value things, and there is quite good evidence that (a) human utility is complicated and (b) it is difficult to explain human decision making in terms of expected utility.

Worked example 5.19 *Human utility is not expected payoff*

Here are four games:

- **Game 1:** The player is given 1. A biased coin is flipped, and the money is taken back with probability p; otherwise, the player keeps it.
- **Game 2:** The player stakes 1, and a fair coin is flipped; if the coin comes up heads, the player gets r and the stake back, but otherwise loses the original stake.
- **Game 3:** The player bets nothing; a biased coin is flipped, and if it comes up heads (probability q), the player gets $1e6$.
- **Game 4:** The player stakes 1000; a fair coin is flipped, and if it comes up heads, the player gets s and the stake back, but otherwise loses the original stake.

In particular, what happens if $r = 3 - 2p$ and $q = (1 - p)/1e6$ and $s = 2 - 2p + 1000$?

Solution: Game 1 has expected value $(1 - p)1$. Game 2 has expected value $(1/2)(r - 1)$. Game 3 has expected value $q1e6$. Game 4 has expected value $(1/2)s - 500$.
In the case given, each game has the same expected value. Nonetheless, people usually have decided preferences for which game they would play. Generally, 4 is unattractive (seems expensive to play); 3 seems like free money, and so a good thing; 2 might be OK but is often seen as uninteresting; and 1 is unattractive. This should suggest to you that people's reasoning about money and utility is not what simple expectations predict.

5.5 YOU SHOULD

5.5.1 remember these definitions:

5.5.2 remember these terms:

5.5.3 remember these facts:

5.5.4 be able to:

- Interpret notation for joint and conditional probability for random variables; in particular, understand notation such as: $P(\{X\})$, $P(\{X = x\})$, $p(x)$, $p(x, y)$, $p(x|y)$
- Interpret a probability density function $p(x)$ as $P(\{X \in [x, x + dx]\})$.
- Interpret the expected value of a discrete random variable.
- Interpret the expected value of a continuous random variable.
- Compute expected values of random variables for straightforward cases.
- Write down expressions for mean, variance and covariance for random variables.
- Write out a decision tree.

PROBLEMS

Joint and Conditional Probability for Random Variables

5.1. A roulette wheel has one zero. Write X for the random variable representing the number that will come up on the wheel. What is the probability distribution of X?

5.2. Define a random variable X by the following procedure. Draw a card from a standard deck of playing cards. If the card is knave, queen, or king, then $X = 11$. If the card is an ace, then $X = 1$; otherwise, X is the number of the card (i.e. two through ten). Now define a second random variable Y by the following procedure. When you evaluate X, you look at the color of the card. If the card is red, then $Y = X - 1$; otherwise, $Y = X + 1$.
- **(a)** What is $P(\{X \leq 2\})$?
- **(b)** What is $P(\{X \geq 10\})$?
- **(c)** What is $P(\{X \geq Y\})$?
- **(d)** What is the probability distribution of $Y - X$?
- **(e)** What is $P(\{Y \geq 12\})$?

5.3. Define a random variable by the following procedure. Flip a fair coin. If it comes up heads, the value is 1. If it comes up tails, roll a die: if the outcome is 2 or 3, the value of the random variable is 2. Otherwise, the value is 3.
- **(a)** What is the probability distribution of this random variable?
- **(b)** What is the cumulative distribution of this random variable?

5.4. Define three random variables, X, Y and Z by the following procedure. Roll a six-sided die and a four-sided die. Now flip a coin. If the coin comes up heads, then X takes the value of the six-sided die and Y takes the value of the four-sided die. Otherwise, X takes the value of the four-sided die and Y takes the value of the six-sided die. Z always takes the value of the sum of the dice.

- **(a)** What is $P(X)$, the probability distribution of this random variable?
- **(b)** What is $P(X, Y)$, the joint probability distribution of these two random variables?
- **(c)** Are X and Y independent?
- **(d)** Are X and Z independent?

5.5. Define two random variables X and Y by the following procedure. Flip a fair coin; if it comes up heads, then $X = 1$, otherwise $X = -1$. Now roll a six-sided die, and call the value U. We define $Y = U + X$.
- **(a)** What is $P(Y|X = 1)$?
- **(b)** What is $P(X|Y = 0)$?
- **(c)** What is $P(X|Y = 7)$?
- **(d)** What is $P(X|Y = 3)$?
- **(e)** Are X and Y independent?

5.6. Magic the Gathering is a popular card game. Cards can be land cards, or other cards. We consider a game with two players. Each player has a deck of 40 cards. Each player shuffles their deck, then deals seven cards, called their *hand*. The rest of each player's deck is called their *library*. Assume that player one has 10 land cards in their deck and player two has 20. Write L_1 for the number of lands in player one's hand and L_2 for the number of lands in player two's hand. Write L_t for the number of lands in the top 10 cards of player one's library.
- **(a)** Write $S = L_1 + L_2$. What is $P(\{S = 0\})$?

(b) Write $D = L_1 - L_2$. What is $P(\{D = 0\})$?
(c) What is the probability distribution for L_1?
(d) Write out the probability distribution for $P(L_1|L_t = 10)$.
(e) Write out the probability distribution $P(L_1|L_t = 5)$.

Continuous Random Variables

5.7. A continuous random variable has probability density function $p(x)$ which is proportional to $g(x)$, where

$$g(x) = \begin{cases} 0 & \text{if } x < -\frac{\pi}{2} \\ 0 & \text{if } x > \frac{\pi}{2} \\ \cos(x) & \text{otherwise} \end{cases} .$$

Write c for the constant of proportionality, so that $p(x) = cg(x)$.
(a) What is c? (you can look up the integral if you want)
(b) What is $P(\{X \geq 0\})$ (i.e. the probability you will observe a value greater than 0)? (you can look up the integral if you want)
(c) What is $P(\{|X| \leq 1\})$? (you can look up the integral if you want)

5.8. There is some (small!) voltage over the terminals of a warm resistor caused by noise (electrons moving around in the heat and banging into one another). This is a good example of a continuous random variable, and we can assume there is some probability density function for it, say $p(x)$. We assume that $p(x)$ has the property that

$$\lim_{\epsilon \to 0} \int_{v-\epsilon}^{v+\epsilon} p(x)dx = 0$$

which is what you'd expect for any function you're likely to have dealt with. Now imagine I define a new random variable by the following procedure: I flip a coin; if it comes up heads, I report 0; if tails, I report the voltage over the resistor. This random variable, u, has a probability $1/2$ of taking the value 0, and $1/2$ of taking a value from $p(x)$. Write this random variable's probability density function $q(u)$. Show that

$$\lim_{\epsilon \to 0} \int_{-\epsilon}^{\epsilon} q(u)du = \frac{1}{2}$$

Expected Values

5.9. Magic the Gathering is a popular card game. Cards can be land cards, or other cards. We consider a game with two players. Each player has a deck of 40 cards. Each player shuffles their deck, then deals seven cards, called their *hand*. The rest of each player's deck is called their *library*. Assume that player one has 10 land cards in their deck and player two has 20. Write L_1 for the number of lands in player one's hand and L_2 for the number of lands in player two's hand. Write L_t for the number of lands in the top 10 cards of player one's library.
(a) What is $\mathbb{E}[L_1]$?
(b) What is $\mathbb{E}[L_2]$?
(c) What is $\mathsf{var}[L_1]$?

5.10. A simple coin game is as follows: we have a box, which starts empty. P1 flips a fair coin. If it comes up heads, P2 gets the contents of the box, and the game

ends. If it comes up tails, P1 puts a dollar in the box and they flip again; this repeats until it comes up heads

(a) With what probability will P2 win exactly 10 units?

(b) Write $S_\infty = \sum_{i=0}^{\infty} r^i$. Show that $(1-r)S_\infty = 1$, so that

$$S_\infty = \frac{1}{1-r}$$

(c) Show that

$$\sum_{i=0}^{\infty} ir^i = (\sum_{i=1}^{\infty} r^i) + r(\sum_{i=1}^{\infty} r^i) + r^2(\sum_{i=1}^{\infty} r^i) + \ldots$$

(look carefully at the limits of the sums!) and so show that

$$\sum_{i=0}^{\infty} ir^i = \frac{r}{(1-r)^2}.$$

(d) What is the expected value of the game? (you may find the results of the two previous subexercises helpful; they're not there just for show).

(e) How much should P2 pay to play, to make the game fair?

5.11. A simple card game is as follows. P1 pays a stake of 1 to play. P1 and P2 then each draw a card. If both cards are the same color, P2 keeps the stake and the game ends. If they are different colors, P2 pays P1 the stake and 1 extra (a total of 2).

(a) What is the expected value of the game to P1?

(b) P2 modifies the game, as follows. If both cards are court cards (that is, knave, queen, king), then P2 keeps the stake and the game ends; otherwise, the game works as before. Now what is the expected value of the game to P1?

5.12. A coin game that is occasionally played is odd person out. In this game, there are rounds. In a round, each person flips a coin. There is an odd person out in that round if all but one have H and the other has T, OR all but one have T and the last has H.

(a) Three people play one round. What is the probability that there is an odd person out?

(b) Now four people play one round. What is the probability that there is an odd person out?

(c) Five people play until there is an odd person out. What is the expected number of rounds that they will play? (you can save yourself quite a lot of calculation by reading section **??**, if you don't mind skipping ahead a bit).

Mean, Variance and Covariance

5.13. Show that $\mathsf{var}[kX] = k^2\mathsf{var}[X]$.

5.14. Show that if X and Y are independent random variables, then $\mathsf{var}[X+Y] = \mathsf{var}[X] + \mathsf{var}[Y]$. You will find it helpful to remember that, for X and Y independent, $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$.

Markov and Chebyshev Inequalities

5.15. The random variable X takes the values -2, -1, 0, 1, 2, but has an unknown probability distribution. You know that $\mathbb{E}[\|X\|] = 0.2$. Use Markov's inequality to give a *lower* bound on $P(\{X = 0\})$. *Hint:* Notice that $P(\{X = 0\}) = 1 - P(\{\|X\| = 1\}) - P(\{\|X\|\} = 2)$.

5.16. The random variable X takes the values 1, 2, 3, 4, 5, but has unknown probability distribution. You know that $\mathbb{E}[X] = 2$ and $\mathsf{var}(\{X\}) = 0.01$. Use Chebychev's inequality to give a *lower* bound on $P(\{X = 2\})$.

5.17. You have a biased random number generator. This generator produces a random number with mean value -1, and standard deviation 0.5. Write $\mathcal{A}$ for the event that the number generator produces a non-negative number. Use Chebyshevs inequality to bound $P(\mathcal{A})$.

5.18. You observe a random number generator. You know that it can produce the values -2, -1, 0, 1, or 2. You are told that it has been adjusted so that: (1) the mean value it produces is zero and; (2) the standard deviation of the numbers it produces is 1.

(a) Write $\mathcal{A}$ for the event that the number generator produces a number that is not 0. Use Chebyshevs inequality to bound $P(\mathcal{A})$.

(b) Write $\mathcal{B}$ for the event that the number generator produces -2 or 2. Use Chebyshevs inequality to bound $P(\mathcal{B})$.

Using Expectations

5.19. Imagine we have a game with two players, who are playing for a stake. There are no draws, the winner gets the whole stake, and the loser gets nothing. The game must end early. We decide to give each player the expected value of the game for that player, from that state. Show that the expected values add up to the value of the stake (i.e. there won't be too little or too much money in the stake.

PROGRAMMING EXERCISES

5.20. An airline company runs a flight that has six seats. Each passenger who buys a ticket has a probability p of turning up for the flight. These events are independent.

(a) The airline sells six tickets. What is the expected number of passengers, if $p = 0.9$?

(b) How many tickets should the airline sell to ensure that the expected number of passengers is greater than six, if $p = 0.7$? **Hint:** The easiest way to do this is to write a quick program that computes the expected value of passengers that turn up for each the number of tickets sold, then search the number of tickets sold.

5.21. An airline company runs a flight that has 10 seats. Each passenger who buys a ticket has a probability p of turning up for the flight. The gender of the passengers is not known until they turn up for a flight, and women buy tickets with the same frequency that men do. The pilot is eccentric, and will not fly unless at least two women turn up.

(a) How many tickets should the airline sell to ensure that the expected number of passengers that turn up is greater than 10?

(b) The airline sells 10 tickets. What is the expected number of passengers on the aircraft, given that it flies? (i.e. that at least two women turn up).

Estimate this value with a simulation.

CHAPTER 6

Useful Probability Distributions

We will use probability as a tool to resolve practical questions about data. I describe some forms these questions take below, for concreteness. Generally, resolving these questions requires some form of model. This model gives an abstract representation of the problem that is useful for problem solving, and (typically) comes with recipes for attacking the major types of problem.

We could ask **what process produced the data?** For example, I observe a set of independent coin flips. I would now like to know the probability of observing a head when the coin is flipped. This might seem a bit empty as a problem, but an analogous problem is: are male children or female children more frequent? Notice that the answers to these questions are typically not exact. Instead, they are estimates. We will see a variety of methods to estimate the probability of observing a head from a sequence of independent coin flips, but none is guaranteed to return the "right" answer (because you can't). Instead, we have to live with information about how accurate the estimate is.

We could ask **what sort of data can we expect in the future?** For example, we could ask: is gender assigned independently? equivalently, can you predict the gender of the next child born to a couple more accurately if you look at the genders of the previous children? In reliability engineering, one asks: how long will it be until this product fails? One version of this question that occupies many people is: how long until I die? By the way, the best answer seems to be subtract your age from a number that seems to be close to 85. Again, these are questions which don't lend themselves to the "right" answer, as opposed to the best possible estimate. You might get hit by a truck tomorrow.

We could ask **what labels should we attach to unlabelled data?** For example, we might see a large number of credit card transactions, some known to be legitimate and others known to be fraudulent. We now see a new transaction: is it legitimate? You can see that versions of this question appear in many applications. As another example, you see many programs downloaded from the web, some known to be legitimate and others known to be malware. You now see a new program: is it safe to run? It may be possible in some circumstances to know the "right" answer to this question, but we are usually stuck with the best answer.

We could ask **is an effect easily explained by chance variations, or is it real?** For example, you believe that the average weight of a mouse is 15 grams. You could test this by catching 100 mice (a sample) and weighing them; but, when you do this, the answer won't be 15 grams *even if* the average weight of a mouse is 15 grams. This is because you have a random selection of mice. But we will be able to build a model of the random variations in the sample average. We can use this model to tell whether the difference between sample average and 15 grams is easily explained by random variations in the sample, or is significant. This allows

us to tell how forcefully the evidence contradicts your original belief.

Building a model requires understanding the specific problem you want to solve, then choosing from a vocabulary of many different models that might apply to the problem. Long experience shows that even if the model you choose does not match the problem exactly, it can still be useful. In this chapter, I describe the properties of some probability distributions that are used again and again in model building.

6.1 DISCRETE DISTRIBUTIONS

6.1.1 The Discrete Uniform Distribution

If every value of a discrete random variable has the same probability, then the probability distribution is the discrete uniform distribution. We have seen this distribution before, numerous times. For example, I define a random variable by the number that shows face-up on the throw of a fair die. This has a uniform distribution. As another example, write the numbers 1-52 on the face of each card of a standard deck of playing cards. The number on the face of the first card drawn from a well-shuffled deck is a random variable with a uniform distribution.

One can construct expressions for the mean and variance of a discrete uniform distribution, but they're not usually much use (too many terms, not often used). Keep in mind that if two random variables have a uniform distribution, their sum and difference will not (recall example 5.3).

6.1.2 Bernoulli Random Variables

A Bernoulli random variable models a biased coin with probability p of coming up heads in any one flip.

Definition: 6.1 *Bernoulli random variable*

A Bernoulli random variable takes the value 1 with probability p and 0 with probability $1 - p$. This is a model for a coin toss, among other things

Useful Facts: 6.1 *Mean and variance of a Bernoulli random variable*

1. A Bernoulli random variable has mean p.
2. A Bernoulli random variable has variance $p(1 - p)$.

Proofs are easy, and in the exercises.

6.1.3 The Geometric Distribution

We have a biased coin. The probability it will land heads up, $P(\{H\})$ is given by p. We flip this coin until the first head appears. The number of flips required is a discrete random variable which takes integer values greater than or equal to one, which we shall call X. To get n flips, we must have $n-1$ tails followed by 1 head. This event has probability $(1-p)^{(n-1)}p$. We can now write out the probability distribution that n flips are required.

Definition: 6.2 *Geometric distribution*

We have an experiment with a binary outcome (i.e. heads or tails; 0 or 1; and so on), with $P(H) = p$ and $P(T) = 1-p$. We repeat this experiment until the first head occurs. The outcomes of each repetition are independent. The probability distribution for n, the number of repetitions, is the geometric distribution. It has the form

$$P(\{X = n\}) = (1-p)^{(n-1)}p.$$

for $0 \leq p \leq 1$ and $n \geq 1$; for other n it is zero. p is called the **parameter** of the distribution.

Notice that the geometric distribution is non-negative everywhere. It is straightforward to show that it sums to one, and so is a probability distribution (exercises).

Useful Facts: 6.2 *Mean and variance of a geometric distribution*

1. The mean of the geometric distribution is $\frac{1}{p}$.
2. The variance of the geometric distribution is $\frac{1-p}{p^2}$.

The proof of these facts requires some work with series, and is relegated to the exercises.

It should be clear that this model isn't really about coins, but about repeated trials. The trial could be anything that has some probability of failing. Each trial is independent, and the rule for repeating is that you keep trying until the first success. Textbooks often set exercises involving missiles and aircraft; I'll omit these on the grounds of taste.

6.1.4 The Binomial Probability Distribution

Assume we have a biased coin with probability p of coming up heads in any one flip. The binomial probability distribution gives the probability that it comes up heads h times in N flips.

Worked example **??** yields one way of deriving this distribution. In that example, I showed that there are

$$N!/(h!(N-h)!)$$

outcomes of N coin flips that have h heads. These outcomes are disjoint, and each has probability $p^h(1-p)^{(N-h)}$. As a result, we must have the probability distribution below.

Definition: 6.3 *Binomial distribution*

In N independent repetitions of an experiment with a binary outcome (ie heads or tails; 0 or 1; and so on) with $P(H) = p$ and $P(T) = 1 - p$, the probability of observing a total of h H's and $N - h$ T's is

$$P_b(h; N, p) = \left(\begin{array}{c} N \\ h \end{array} \right) p^h (1-p)^{(N-h)}$$

as long as $0 \leq h \leq N$; in any other case, the probability is zero.

The binomial distribution really is a probability distribution. For $0 \leq p \leq 1$, it is clearly non-negative for any i. It also sums to one. Write $P_b(i; N, p)$ for the binomial distribution that one observes i H's in N trials.

$$\sum_{i=0}^{N} P_b(i; N, p) = (p + (1-p))^N = (1)^N = 1$$

by pattern matching to the binomial theorem. As a result,

$$\sum_{i=0}^{N} P_b(i; N, p) = 1$$

Furthemore, the binomial distribution satisfies a recurrence relation. We must have that

$$P_b(h; N, p) = pP_b(h-1; N-1, p) + (1-p)P_b(h; N-1, p).$$

This is because can get h heads in N flips either by having $h-1$ heads in $N-1$ flips, then flipping another, or by having h heads in N flips then flipping a tail. You can verify by induction that the binomial distribution satisfies this recurrence relation.

Useful Facts: 6.3 *Mean and variance of the binomial distribution*

1. The mean of $P_b(i; N, p)$ is Np.
2. The variance of $P_b(i; N, p)$ is $Np(1-p)$

The proofs are informative, and so are not banished to the exercises.

Proof: 6.1 *The binomial distribution*

Notice that the number of heads in N coin tosses is can be obtained by adding the number of heads in each toss. Write Y_i for the Bernoulli random variable representing the i'th toss. If the coin comes up heads, $Y_i = 1$, otherwise $Y_i = 0$. The Y_i are independent. Now

$$\begin{aligned}
\mathbb{E}[X] &= \mathbb{E}\left[\sum_{j=1}^{N} Y_i\right] \\
&= \sum_{j=1}^{N} \mathbb{E}[Y_i] \\
&= N\mathbb{E}[Y_1] \qquad \text{because the } Y_i \text{ are independent} \\
&= Np.
\end{aligned}$$

The variance is easy, too. Each coin toss is independent, so the variance of the sum of coin tosses is the sum of the variances. This gives

$$\begin{aligned}
\mathsf{var}[X] &= \mathsf{var}\left[\sum_{j=1}^{N} Y_i\right] \\
&= N\mathsf{var}[Y_1] \\
&= Np(1-p)
\end{aligned}$$

6.1.5 Multinomial probabilities

The binomial distribution describes what happens when a coin is flipped multiple times. But we could toss a die multiple times too. Assume this die has k sides, and we toss it N times. The distribution of outcomes is known as the **multinomial distribution**.

We can guess the form of the multinomial distribution in rather a straightforward way. The die has k sides. We toss the die N times. This gives us a sequence of N numbers. Each toss of the die is independent. Assume that side 1 appears n_1 times, side 2 appears n_2 times, ... side k appears n_k times. Any single sequence with this property will appear with probability $p_1^{n_1} p_2^{n_2} ... p_k^{n_k}$, because the tosses are independent. However, there are

$$\frac{N!}{n_1! n_2! ... n_k!}$$

such sequences. Using this reasoning, we arrive at the distribution below

Definition: 6.4 *Multinomial distribution*

I perform N independent repetitions of an experiment with k possible outcomes. The i'th such outcome has probability p_i. I see outcome 1 n_1 times, outcome 2 n_2 times, etc. Notice that $n_1+n_2+n_3+\ldots+n_k = N$. The probability of observing this set of outcomes is

$$P_m(n_1,\ldots,n_k;N,p_1,\ldots,p_k) = \frac{N!}{n_1!n_2!\ldots n_k!}p_1^{n_1}p_2^{n_2}\ldots p_k^{n_k}.$$

Worked example 6.1 *Dice*

I throw five fair dice. What is the probability of getting two 2's and three 3's?

Solution: $\frac{5!}{2!3!}(\frac{1}{6})^2(\frac{1}{6})^3$

6.1.6 The Poisson Distribution

Assume we are interested in counts that occur in an interval of time (e.g. within a particular hour). Because they are counts, they are non-negative and integer valued. We know these counts have two important properties. First, they occur with some fixed average rate. Second, an observation occurs independent of the interval since the last observation. Then the Poisson distribution is an appropriate model.

There are numerous such cases. For example, the marketing phone calls you receive during the day time are likely to be well modelled by a Poisson distribution. They come at some average rate — perhaps 5 a day as I write, during the last phases of an election year — and the probability of getting one clearly doesn't depend on the time since the last one arrived. Classic examples include the number of Prussian soldiers killed by horse-kicks each year; the number of calls arriving at a call center each minute; the number of insurance claims occurring in a given time interval (outside of a special event like a hurricane, etc.).

Definition: 6.5 *Poisson distribution*

A non-negative, integer valued random variable X has a Poisson distribution when its probability distribution takes the form

$$P(\{X = k\}) = \frac{\lambda^k e^{-\lambda}}{k!},$$

where $\lambda > 0$ is a parameter often known as the **intensity** of the distribution.

Notice that the Poisson distribution is a probability distribution, because it is non-negative and because

$$\sum_{i=0}^{\infty} \frac{\lambda^i}{i!} = e^{\lambda}$$

so that

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} = 1$$

Useful Facts: 6.4 *Mean and variance of the Poisson distribution*

1. The mean of a Poisson distribution with intensity λ is λ.
2. The variance of a Poisson distribution with intensity λ is λ (no, that's not an accidentally repeated line or typo).

The proof of these facts requires some work with series, and is relegated to the exercises.

I described the Poisson distribution as a natural model for counts of randomly distributed points along a time axis. But it doesn't really matter that this is a time axis — it could be a space axis instead. For example, you could take a length of road, divide it into even intervals, then count the number of road-killed animals is in each interval. If the location of each animal is independent of the location of any other animal, then you could expect a Poisson model to apply to the count data. Assume that the Poisson model that best describes the data has parameter λ. One property of such models is that if you doubled the length of the intervals, then the resulting dataset would be described by a Poisson model with parameter 2λ; similarly, if you halved the length of the intervals, the best model would have parameter $\lambda/2$. This corresponds to our intuition about such data; roughly, the number of road-killed animals in two miles of road should be twice the number in

one mile of road. This property means that no pieces of the road are "special" — each behaves the same as the other.

We can build a really useful model of spatial randomness by observing this fact and generalizing very slightly. A **Poisson point process** with intensity λ is a set of random points with the property that the number of points in an interval of length s is a Poisson random variable with parameter λs. Notice how this captures our intuition that if points are "very randomly" distributed, there should be twice as many of them in an interval that is twice as long.

This model is easily, and very usefully, extended to points on the plane, on surfaces, and in 3D. In each case, the process is defined on a domain D (which has to meet some very minor conditions that are of no interest to us). The number of points in any subset s of D is a Poisson random variable, with intensity $\lambda m(s)$, where $m(s)$ is the area (resp. volume) of s. These models are useful, because they capture the property that (a) the points are random and (b) the probability you find a point doesn't depend on where you are. You could reasonably believe models like this apply to, say, dead flies on windscreens; the places where you find acorns at the foot of an oak tree; the distribution of cowpats in a field; the distribution of cherries in a fruitcake; and so on.

6.2 CONTINUOUS DISTRIBUTIONS

6.2.1 The Continuous Uniform Distribution

Some continuous random variables have a natural upper bound and a natural lower bound but otherwise we know nothing about them. For example, imagine we are given a coin of unknown properties by someone who is known to be a skillful maker of unfair coins. The manufacturer makes no representations as to the behavior of the coin. The probability that this coin will come up heads is a random variable, about which we know nothing except that it has a lower bound of zero and an upper bound of one.

If we know nothing about a random variable apart from the fact that it has a lower and an upper bound, then a **uniform distribution** is a natural model. Write l for the lower bound and u for the upper bound. The probability density function for the uniform distribution is

$$p(x) = \begin{cases} 0 & x < l \\ 1/(u-l) & l \leq x \leq u \\ 0 & x > u \end{cases}$$

A continuous random variable whose probability distribution is the uniform distribution is often called a **uniform random variable**.

6.2.2 The Beta Distribution

It's hard to explain now why the Beta (or β) distribution is useful, but it will come in useful later (section 13.1). The Beta distribution is a probability distribution for a continuous random variable x in the range $0 \leq x \leq 1$. There are two parameters, $\alpha > 0$ and $\beta > 0$. Recall the definition of the Γ function from section **??**.

Definition: 6.6 *Beta distribution*

A continuous random variable x in the range $0 \leq x \leq 1$ has a Beta distribution if its probability density function has the form

$$P_\beta(x|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{(\alpha-1)}(1-x)^{(\beta-1)}.$$

where $\alpha > 0$ and $\beta > 0$.

From the expression for the Beta distribution, you can see that:

- $P_\beta(x|1,1)$ is a uniform distribution on the unit interval.
- $P_\beta(x|\alpha,\beta)$ has a single maximum at $x = (\alpha-1)/(\alpha+\beta-2)$ for $\alpha > 1$, $\beta > 1$) (differentiate and set to zero).
- Generally, as α and β get larger, this peak gets narrower.
- For $\alpha = 1$, $\beta > 1$ the largest value of $P_\beta(x|\alpha,\beta)$ is at $x = 0$.
- For $\alpha > 1$, $\beta = 1$ the largest value of $P_\beta(x|\alpha,\beta)$ is at $x = 1$.

Figure 6.1 shows plots of the probability density function of the Beta distribution for a variety of different values of α and β.

Useful Facts: 6.5 *Mean and variance of a Beta distribution*

For a Beta distribution with parameters α, β

1. The mean is $\frac{\alpha}{\alpha+\beta}$.
2. The variance is $\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$.

6.2.3 The Gamma Distribution

The Gamma (or γ) distribution will also come in useful later on (section 13.1). The Gamma distribution is a probability distribution for a non-negative continuous random variable $x \geq 0$. There are two parameters, $\alpha > 0$ and $\beta > 0$.

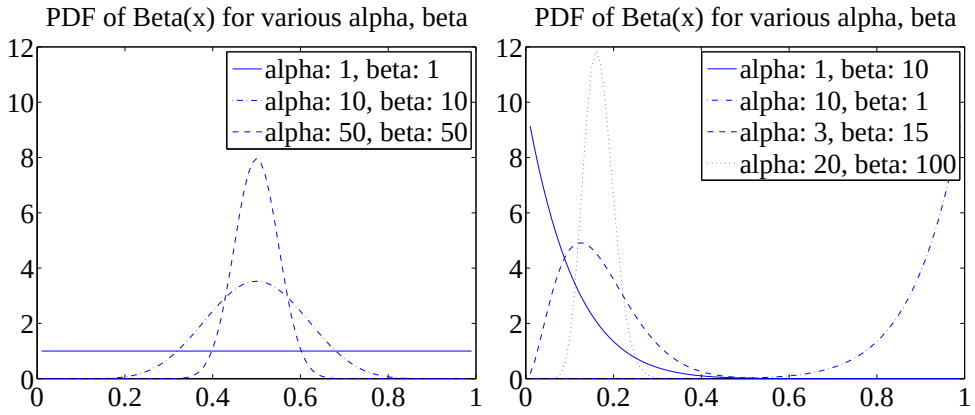

FIGURE 6.1: *Probability density functions for the Beta distribution with a variety of different choices of α and β.*

Definition: 6.7 *Gamma distribution*

A non-negative continuous random variable x has a Gamma distribution if its probability density function is

$$P_\gamma(x|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{(\alpha-1)} e^{-\beta x}.$$

where $\alpha > 0$ and $\beta > 0$.

Figure 6.2 shows plots of the probability density function of the Gamma distribution for a variety of different values of α and β.

Useful Facts: 6.6 *Mean and variance of the gamma distribution*

For a Gamma distribution with parameters α, β

1. The mean is $\frac{\alpha}{\beta}$.
2. The variance is $\frac{\alpha}{\beta^2}$.

6.2.4 The Exponential Distribution

Assume we have an infinite interval of time or space, with points distributed on it. Assume these points form a Poisson point process, as above. For example, we

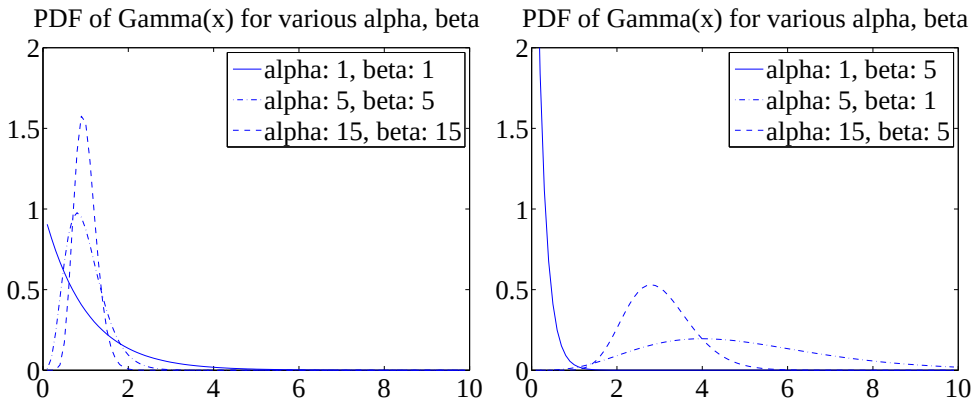

FIGURE 6.2: *Probability density functions for the Gamma distribution with a variety of different choices of α and β.*

might consider the times at which email arrives; or the times at which phone calls arrive at a large telephone exchange; or the locations of roadkill on a road. The distance (or span of time) between two consecutive points is a random variable X. This random variable takes an exponential distribution, defined below. There is a single parameter, $\lambda > 0$. This distribution is often useful in modelling the failure of objects. We assume that failures form a Poisson process in time; then the time to the next failure is exponentially distributed.

Definition: 6.8 *Exponential distribution*

A continuous random variable x has an exponential distribution when its probability density function takes the form

$$P_{\exp}(x|\lambda) = \begin{cases} \lambda \exp^{-\lambda x} & \text{for } x \geq 0 \\ 0 & \text{otherwise} \end{cases} .$$

where $\lambda > 0$ is a parameter.

Useful Facts: 6.7 *Mean and variance of the exponential distribution*

For an exponential distribution with parameter λ

1. The mean is
$$\frac{1}{\lambda}.$$
2. The variance is
$$\frac{1}{\lambda^2}.$$

Notice the relationship between this parameter and the parameter of the Poisson distribution. If (say) the phone calls are distributed with Poisson distribution with intensity λ (per hour), then your expected number of calls per hour is λ. The time between calls will be exponentially distributed with parameter λ, and the expected time to the next call is $1/\lambda$ (in hours).

6.3 THE NORMAL DISTRIBUTION

6.3.1 The Standard Normal Distribution

Definition: 6.9 *Standard Normal distribution*

The probability density function

$$p(x) = \left(\frac{1}{\sqrt{2\pi}}\right) \exp\left(\frac{-x^2}{2}\right).$$

is known as the **standard normal distribution**

The first step is to plot this probability density function (Figure 6.3). You should notice it is quite familiar from work on histograms, etc. in Chapter 13.1. It has the shape of the histogram of standard normal data, or at least the shape that the histogram of standard normal data aspires to.

Useful Facts: 6.8 *Mean and variance of the standard normal distribution*

1. The mean of the standard normal distribution is 0.
2. The variance of the standard normal distribution is 1.

These results are easily established by looking up (or doing!) the relevant integrals; they are relegated to the exercises.

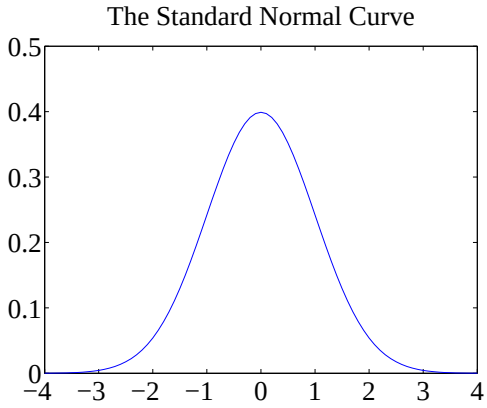

FIGURE 6.3: *A plot of the probability density function of the standard normal distribution. Notice how probability is concentrated around zero, and how there is relatively little probability density for numbers with large absolute values.*

A continuous random variable is a **standard normal random variable** if its probability density function is a standard normal distribution.

6.3.2 The Normal Distribution

Any probability density function that is a standard normal distribution *in standard coordinates* is a **normal distribution**. Now write μ for the mean of a random variable and σ for its standard deviation; we are saying that, if

$$\frac{x-\mu}{\sigma}$$

has a standard normal distribution, then $p(x)$ is a normal distribution. We can work out the form of the probability density function of a general normal distribution in two steps: first, we notice that for any normal distribution, we must have

$$p(x) \propto \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right].$$

But, for this to be a probability density function, we must have $\int_{-\infty}^{\infty} p(x)dx = 1$. This yields the constant of proportionality, and we get

Definition: 6.10 *Normal distribution*

The probability density function

$$p(x) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right) \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right).$$

is a normal distribution.

Useful Facts: 6.9 *Mean and variance of the normal distribution*

The probability density function

$$p(x) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right) \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right).$$

has

1. mean μ
2. and variance σ.

These results are easily established by looking up (or doing!) the relevant integrals; they are relegated to the exercises.

A continuous random variable is a **normal random variable** if its probability density function is a **normal distribution**. Notice that it is quite usual to call normal distributions **gaussian distributions**.

6.3.3 Properties of the Normal Distribution

Normal distributions are important, because one often runs into data that is well described by a normal distribution. It turns out that anything that behaves like a binomial distribution with a lot of trials — for example, the number of heads in many coin tosses; as another example, the percentage of times you get the outcome of interest in a simulation in many runs — should produce a normal distribution (Section 6.4). For this reason, pretty much any experiment where you perform a simulation, then count to estimate a probability or an expectation, should give you an answer that has a normal distribution.

It is a remarkable and deep fact, known as the **central limit theorem**, that adding many independent random variables produces a normal distribution pretty much *whatever* the distributions of those random variables. I've not shown this in detail because it's a nuisance to prove. However, if you add together many random variables, each of pretty much any distribution, then the answer has a

distribution close to the normal distribution. It turns out that many of the processes we observe add up subsidiary random variables. This means that you will see normal distributions very often in practice.

A normal random variable tends to take values that are quite close to the mean, measured in standard deviation units. We can demonstrate this important fact by computing the probability that a standard normal random variable lies between u and v. We form

$$\int_u^v \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du.$$

It turns out that this integral can be evaluated relatively easily using a special function. The **error function** is defined by

$$\mathsf{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp\left(-t^2\right) dt$$

so that

$$\frac{1}{2}\mathsf{erf}\left((\frac{x}{\sqrt{2}})\right) = \int_0^x \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du.$$

Notice that $\mathsf{erf}(x)$ is an odd function (i.e. $\mathsf{erf}(-x) = \mathsf{erf}(x)$). From this (and tables for the error function, or Matlab) we get that, for a standard normal random variable

$$\frac{1}{\sqrt{2\pi}} \int_{-1}^1 \exp\left(-\frac{x^2}{2}\right) dx \approx 0.68$$

and

$$\frac{1}{\sqrt{2\pi}} \int_{-2}^2 \exp\left(-\frac{x^2}{2}\right) dx \approx 0.95$$

and

$$\frac{1}{\sqrt{2\pi}} \int_{-2}^2 \exp\left(-\frac{x^2}{2}\right) dx \approx 0.99.$$

These are very strong statements. They measure how often a standard normal random variable has values that are in the range $-1, 1$, $-2, 2$, and $-3, 3$ respectively. But these measurements apply to normal random variables if we recognize that they now measure how often the normal random variable is some number of standard deviations away from the mean. In particular, it is worth remembering that:

Useful Facts: 6.10 *Normal Random Variables*

- About 68% of the time, a normal random variable takes a value within one standard deviation of the mean.
- About 95% of the time, a normal random variable takes a value within one standard deviation of the mean.
- About 99% of the time, a normal random variable takes a value within one standard deviation of the mean.

6.4 APPROXIMATING BINOMIALS WITH LARGE N

The Binomial distribution appears to be a straightforward thing. We assume we flip a coin N times, where N is a very large number. The coin has probability p of coming up heads, and so probability $q = 1 - p$ of coming up tails. The number of heads h follows the binomial distribution, so

$$P(h) = \frac{N!}{h!(N-h)!} p^h q^{(N-h)}$$

The mean of this distribution is Np, the variance is Npq, and the standard deviation is $\sqrt{Npq}$.

Evaluating this probability distribution for large N is very difficult, because factorials grow fast. We will construct an approximation to the binomial distribution for large N that allows us to evaluate the probability that h lies in some range. This approximation will show that the probability that h is within one standard deviation of the mean is approximately 68%.

This is important, because it shows that our model of probability as frequency is consistent. Consider the probability that the number of heads you see lies within one standard deviation of the mean. The size of that interval is $2\sqrt{Npq}$. As N gets bigger, the size of that interval, *relative to the total number of flips*, gets smaller. If I flip a coin N times, in principle I could see a number of heads h that ranges from 0 to N. However, we will establish that about 68% of the time, h will lie in the interval within one standard deviation of the mean. The size of this interval, *relative to the total number of flips* is

$$2\frac{\sqrt{Npq}}{N} = 2\sqrt{\frac{pq}{N}}.$$

As a result, as $N \to \infty$,

$$\frac{h}{N} \to p$$

because h will tend to land in an interval around pN that gets narrower as N gets larger.

The main difficulty with Figure 6.4 (and with the argument above) is that the mean and standard deviation of the binomial distribution tends to infinity as the number of coin flips tends to infinity. This can confuse issues. For example, the plots of Figure 6.4 show narrowing probability distributions — but is this because the scale is compacted, or is there a real effect? It turns out there is a real effect, and a good way to see it is to consider the normalized number of heads.

6.4.1 Large N

Recall that to normalize a dataset, you subtract the mean and divide the result by the standard deviation. We can do the same for a random variable. We now consider

$$x = \frac{h - Np}{\sqrt{Npq}}.$$

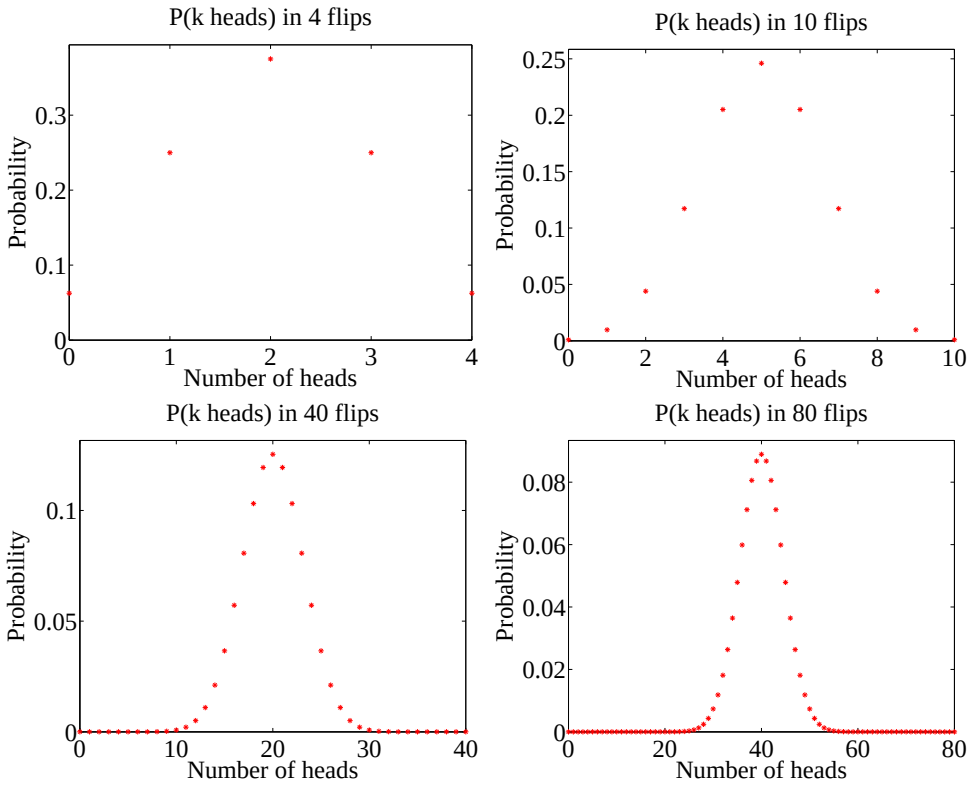

FIGURE 6.4: *Plots of the binomial distribution for $p = q = 0.5$ for different values of N. You should notice that the set of values of h (the number of heads) that have substantial probability is quite narrow compared to the range of possible values. This set gets narrower as the number of flips increases. This is because the mean is pN and the standard deviation is $\sqrt{Npq}$ — so the fraction of values that is within one standard deviation of the mean is $O(1/\sqrt{N})$.*

The probability distribution of x can be obtained from the probability distribution for h, because $h = Np + x\sqrt{Npq}$, so

$$P(x) = \left(\frac{N!}{(Np + x\sqrt{Npq})!(Nq - x\sqrt{Npq})!} \right) p^{(Np + x\sqrt{Npq})} q^{(Nq - x\sqrt{Npq})}.$$

I have plotted this probability distribution for various values of N in Figure 6.5.

But it is hard to work with this distribution for very large N. The factorials become very difficult to evaluate. Second, it is a discrete distribution on N points, spaced $1/\sqrt{Npq}$ apart. As N becomes very large, the number of points that have non-zero probability becomes very large, and x can be very large, or very small. For example, there is some probability, though there may be very little indeed, on the point where $h = N$, or, equivalently, $x = N(p + \sqrt{Npq})$. For sufficiently large N, we think of this probability distribution as a probability density function. We can

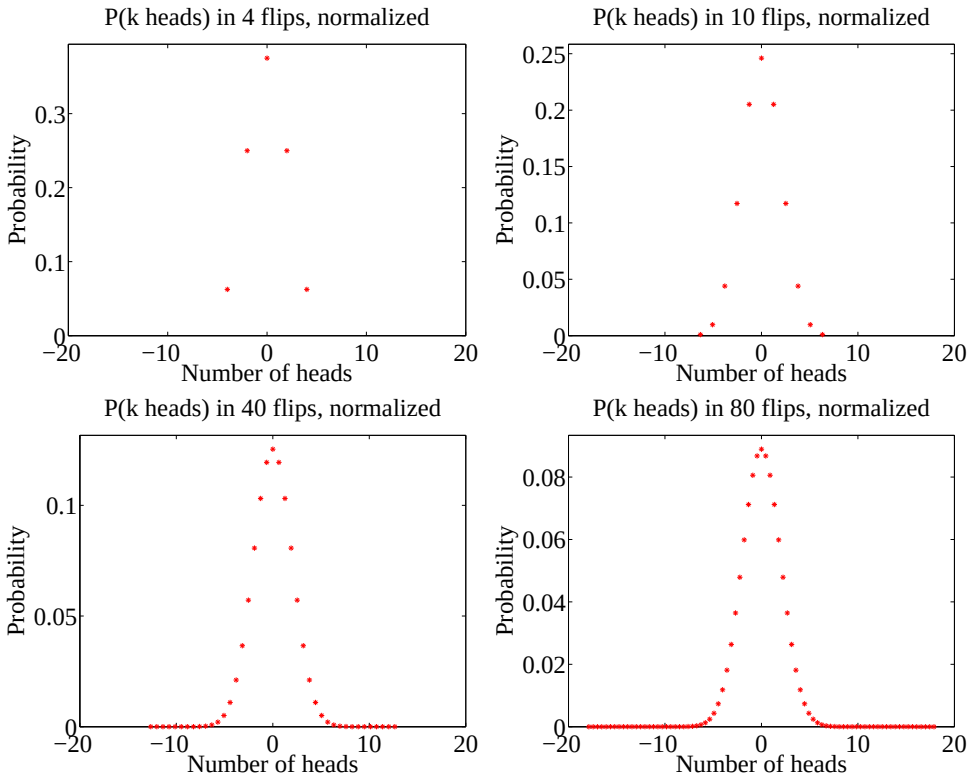

FIGURE 6.5: *Plots of the distribution for the normalized variable x, with $P(x)$ given in the text, obtained from the binomial distribution with $p = q = 0.5$ for different values of N. These distributions are normalized (mean 0, variance 1. They look increasingly like a standard normal distribution EXCEPT that the value at their mode gets smaller as N gets bigger (there are more possible outcomes). In the text, we will establish that the standard normal distribution is a limit, in a useful sense.*

do so, for example, by spreading the probability for x_i (the i'th value of x) evenly over the interval between x_i and x_{i+1}. We then have a probability density function that looks like a histogram, with bars that become narrower as N increases. But what is the limit?

6.4.2 Getting Normal

To proceed, we need Stirling's approximation, which says that, for large N,

$$N! \approx \sqrt{2\pi}\sqrt{N}\left(\frac{N}{e}\right)^N.$$

This yields

$$P(h) \approx \left(\frac{Np}{h}\right)^h \left(\frac{Nq}{N-h}\right)^{(N-h)} \sqrt{\frac{N}{2\pi h(N-h)}}$$

Recall we used the normalized variable

$$x = \frac{h - Np}{\sqrt{Npq}}.$$

We will encounter the term $\sqrt{Npq}$ often, and we use $\sigma = \sqrt{Npq}$ as a shorthand. We can compute h and $N-h$ from x by the equalities

$$h = Np + \sigma x \qquad\qquad N - h = Nq - \sigma x.$$

So the probability distribution written in this new variable x is

$$P(x) \approx \left(\frac{Np}{(Np+\sigma x)}\right)^{(Np+\sigma x)} \left(\frac{Nq}{(Nq-\sigma x)}\right)^{(Nq-\sigma x)} \sqrt{\frac{N}{2\pi(Np+\sigma x)(Nq-\sigma x)}}$$

There are three terms to deal with here. It is easiest to work with $\log P$. Now

$$\log(1+x) = x - \frac{1}{2}x^2 + O(x^3)$$

so we have

$$\begin{aligned} \log\left(\frac{Np}{(Np+\sigma x)}\right) &= -\log\left(1 + \frac{\sigma x}{Np}\right) \\ &\approx -\frac{\sigma x}{Np} + (\frac{1}{2})(\frac{\sigma x}{Np})^2 \end{aligned}$$

and

$$\log\left(\frac{Nq}{(Nq-\sigma x)}\right) \approx \frac{\sigma x}{Nq} + (\frac{1}{2})(\frac{\sigma x}{Nq})^2.$$

From this, we have that

$$\begin{aligned} \log\left[\left(\frac{Np}{Np+\sigma x}\right)^{(Np+\sigma x)} \left(\frac{Nq}{Nq-\sigma x}\right)^{(Nq-\sigma x)}\right] &\approx [Np+\sigma x]\left[-\frac{\sigma x}{Np} + \left(\frac{1}{2}\right)\left(\frac{\sigma x}{Np}\right)^2\right] + \\ &\quad [Nq-\sigma x]\left[\frac{\sigma x}{Nq} + \left(\frac{1}{2}\right)\left(\frac{\sigma x}{Nq}\right)^2\right] \\ &= -\left(\frac{1}{2}\right)x^2 + O((\sigma x)^3) \end{aligned}$$

(recall $\sigma = \sqrt{Npq}$ if you're having trouble with the last step). Now we look at the square-root term. We have

$$\begin{aligned} \log\sqrt{\frac{N}{2\pi(Np+\sigma x)(Nq-\sigma x)}} &= -\frac{1}{2}\left(\log[Np+\sigma x] + \log[Nq-\sigma x] - \log N + \log 2\pi\right) \\ &= -\frac{1}{2}\begin{pmatrix} \log Np + O\left(\left(\frac{\sigma x}{Np}\right)\right) \\ +\log Nq - O\left(\left(\frac{\sigma x}{Nq}\right)\right) \\ -\log N + \log 2\pi \end{pmatrix} \end{aligned}$$

but, since N is very large compared to σx, we can ignore the $O(\left(\frac{\sigma x}{Np}\right))$ terms. Then this term is not a function of x. So we have

$$\log P(x) \approx \frac{-x^2}{2} + \text{constant}.$$

Now because N is very large, our probability distribution P limits to a probability density function p, with

$$p(x) \propto \exp\left(\frac{-x^2}{2}\right).$$

We can get the constant of proportionality from integrating, to

$$p(x) = \left(\frac{1}{\sqrt{2\pi}}\right)\exp\left(\frac{-x^2}{2}\right).$$

This constant of proportionality deals with the effect in figure 6.5, where the mode of the distribution gets smaller as N gets bigger. It does so because there are more points with non-zero probability to be accounted for. But we are interested in the limit where N tends to infinity. This must be a probability density function, so it must integrate to one.

Review this blizzard of terms. We started with a binomial distribution, but standardized the variables so that the mean was zero and the standard deviation was one. We then assumed there was a very large number of coin tosses, so large that that the distribution started to look like a continuous function. The function we get is the standard normal distribution.

6.4.3 So What?

I have proven an extremely useful fact, which I shall now put in a box.

Useful Fact: 6.11 *Binomial distribution for large N*

Assume h follows the binomial distribution with parameters p and q. Write $x = \frac{h-Np}{\sqrt{Npq}}$. Then, for sufficiently large N, the probability distribution $P(x)$ can be approximated by the probability density function

$$\left(\frac{1}{\sqrt{2\pi}}\right)\exp\left(\frac{-x^2}{2}\right)$$

in the sense that

$$P(\{x \in [a,b]\}) \approx \int_a^b \left(\frac{1}{\sqrt{2\pi}}\right)\exp\left(\frac{-u^2}{2}\right)du$$

This justifies our model of probability as frequency. I interpreted an event having probability p to mean that, if I had a large number N of independent repetitions of the experiment, the number that produced the event would be close to Np, and would get closer as N got larger. We know that, for example, 68% of the time a standard normal random variable takes a value between 1 and -1. In this case, the standard normal random variable is

$$\frac{h-(Np)}{\sqrt{Npq}}$$

so that 68% of the time, h must take a value in the range $[Np-\sqrt{Npq}, Np+\sqrt{Npq}]$. Equivalently, the relative frequency h/N must take a value in the range

$$[p-\frac{pq}{\sqrt{N}}, p+\frac{pq}{\sqrt{N}}]$$

but as $N \to \infty$ this range gets smaller and smaller, and h/N limits to p. So our view of probability as a frequency is consistent.

To obtain h, we added N independent Bernoulli random variables. So you can interpret the box as saying that the sum of many independent Bernoulli random variables has a probability distribution that limits to the normal distribution as the number added together gets larger. Remember that I have stated, though not precisely, but not proved the deep and useful fact that the sum of pretty much any independent random variables has a distribution that gets closer to a normal distribution as the number added together gets larger.

6.5 YOU SHOULD

6.5.1 remember these definitions:

6.5.2 remember these terms:

6.5.3 remember these facts:

PROBLEMS

Sums and Differences of Discrete Random Variables

6.1. Assume X and Y are discrete random variables which take integer values in the range $1 \ldots 100$ (inclusive). Write $S = X + Y$.

(a) Show that

$$P(S = k) = \sum_{u=1}^{u=100} P(\{\{X = k - u\} \cap \{Y = u\}\}).$$

(b) Show that

$$P(D = k) = \sum_{u=1}^{u=100} P(\{X = k + u\})P(\{Y = u\}).$$

(c) Now assume that both X and Y are uniform random variables. Show that S is not uniform by considering $P(S = 2)$, $P(S = 3)$, and $P(S = 100)$.

The Geometric Distribution

6.2. Write $S_\infty = \sum_{i=0}^{\infty} r^i$. Show that $(1 - r)S_\infty = 1$, so that

$$S_\infty = \frac{1}{1 - r}$$

6.3. Write $P(\{X = n\})$ for the probability that an experiment requires n repeats for success under the geometric distribution model with probability of success in one experiment p. Use the result of the previous exercise to show that

$$\begin{aligned} \sum_{n=1}^{\infty} P(\{X = n\}) &= p \sum_{n=1}^{\infty} (1 - p)^{(n-1)} \\ &= 1 \end{aligned}$$

6.4. Show that

$$\sum_{i=0}^{\infty} ir^i = (\sum_{i=1}^{\infty} r^i) + r(\sum_{i=1}^{\infty} r^i) + r^2(\sum_{i=1}^{\infty} r^i) + \ldots$$

(look carefully at the limits of the sums!) and so show that

$$\sum_{i=0}^{\infty} ir^i = \frac{r}{(1 - r)^2}.$$

6.5. Write $S_\infty = \sum_{i=0}^{\infty} r^i$. Show that

$$\sum_{i=0}^{\infty} i^2 r^i = (S_\infty - 1) + 3r(S_\infty - 1) + 5r^2(S_\infty - 1) + 7r^3(\S_\infty - 1) + \ldots$$

and so that

$$\sum_{i=0}^{\infty} i^2 r^i = \frac{r(1 + r)}{(1 - r)^3}$$

6.6. Show that, for a geometric distribution with parameter p, the mean is

$$\sum_{i=1}^{\infty} i(1-p)^{(i-1)}p = \sum_{u=0}^{\infty}(u+1)(1-p)^u p.$$

Now by rearranging and using the previous results, show that the mean is

$$\sum_{i=1}^{\infty} i(1-p)^{(i-1)}p = \frac{1}{p}$$

6.7. Show that, for a geometric distribution with parameter p, the variance is $\frac{1-p}{p^2}$. To do this, note the variance is $\mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2$. Now use the results of the previous exercises to show that

$$\mathbb{E}\left[X^2\right] = \sum_{i=1}^{\infty} i^2(1-p)^{(i-1)}p = \frac{p}{1-p}\frac{(1-p)(2-p)}{p^3},$$

then rearrange to get the expression for variance.

6.8. You have a coin with unknown probability p of coming up heads. You wish to generate a random variable which takes the values 0 and 1, each with probability 1/2. Assume $0 < p < 1$. You adopt the following procedure. You start by flipping the coin twice. If both flips produce the same side of the coin, you start again. If the result of the first flip is different from the result of the second flip, you report the result of the first flip and you are finished (this is a trick originally due to John von Neumann).

(a) Show that, in this case, the probability of reporting heads is 1/2.

(b) What is the expected number of flips you must make before you report a result?

Bernoulli Random Variables

6.9. Write X for a Bernoulli random variable which takes the value 1 with probability p (and 0 with probability $(1-p)$).

(a) Show that $\mathbb{E}[X] = p$.

(b) Show that $\mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 = p(1-p)$

The Binomial Distribution

6.10. Show that $P_b(N-i; N, p) = P_b(i; N, p)$ for all i.

6.11. Write h_r for the number of heads obtained in r flips of a coin which has probability p of coming up heads. Compare the following two ways to compute the probability of getting i heads in five coin flips:

- Flip the coin three times, count h_3, then flip the coin twice, count h_2, then form $w = h_3 + h_2$.
- Flip the coin five times, and count h_5.

Show that the probability distribution for w is the same as the probability distribution for h_5. Do this by showing that

$$P(\{w = i\}) = \sum_{j=0}^{5} P(\{h_3 = j\} \cap \{h_2 = i - j\}) = P(\{h_5 = i\}).$$

6.12. Now we will do the previous exercise in a more general form. Again, write h_r for the number of heads obtained in r flips of a coin which has probability p of coming up heads. Compare the following two ways to compute the probability of getting i heads in N coin flips:

- Flip the coin t times, count h_t, then flip the coin $N-t$ times, count h_{N-t}, then form $w = h_t + h_{N-t}$.
- Flip the coin N times, and count h_N.

Show that the probability distribution for w is the same as the probability distribution for h_N. Do this by showing that

$$P(\{w = i\}) = \sum_{j=0}^{N} P(\{h_t = j\} \cap \{h_{N-t} = i - j\}) = P(\{h_N = i\}).$$

You will likely find the recurrence relation

$$P_b(i; N, p) = pP_b(i-1; N-1, p) + (1-p)P_b(i; N-1, p).$$

is useful.

6.13. An airline runs a regular flight with six seats on it. The airline sells six tickets. The gender of the passengers is unknown at time of sale, but women are as common as men in the population. All passengers always turn up for the flight. The pilot is eccentric, and will not fly a plane unless at least one passenger is female. What is the probability that the pilot flies?

6.14. An airline runs a regular flight with s seats on it. The airline always sells t tickets for this flight. The probability a passenger turns up for departure is p, and passengers do this independently. What is the probability that the plane travels with exactly 3 empty seats?

6.15. An airline runs a regular flight with s seats on it. The airline always sells t tickets for this flight. The probability a passenger turns up for departure is p, and passengers do this independently. What is the probability that the plane travels with 1 or more empty seats?

The Multinomial Distribution

6.16. Show that the multinomial distribution

$$P_m(n_1, \ldots, n_k; N, p_1, \ldots, n_k) = \frac{N!}{n_1!n_2!\ldots n_k!} p_1^{n_1} p_2^{n_2} \ldots p_k^{n_k}$$

must satisfy the recurrence relation

$$\begin{aligned} P_m(n_1, \ldots, n_k; N, p_1, \ldots, p_k) \quad = \quad & p_1 P_m(n_1 - 1, \ldots, n_k; N-1, p_1, \ldots, p_k) + \\ & p_2 P_m(n_1, n_2 - 1, \ldots, n_k; N-1, p_1, \ldots, p_k) + \ldots \\ & p_k P_m(n_1, n_2, \ldots, n_k - 1; N-1, p_1, \ldots, p_k) \end{aligned}$$

The Poisson Distribution

6.17. Compute the Taylor series for xe^x around $x = 0$. Use this and pattern matching to show that the mean of the Poisson distribution with intensity parameter λ is λ.

6.18. Compute the Taylor series for $(x^2 + x)e^x$ around $x = 0$. Use this and pattern matching to show that the variance of the Poisson distribution with intensity parameter λ is λ.

Sums of Continuous Random Variables

6.19. Write p_x for the probability density function of a continuous random variable X and p_y for the probability density function of a continuous random variable Y. Show that the probability density function of $S = X + Y$ is

$$p(s) = \int_{-\infty}^{\infty} p_x(s-u)p_y(u)du = \int_{-\infty}^{\infty} p_x(u)p_y(s-u)du$$

The Normal Distribution

6.20. Write

$$f(x) = \left(\frac{1}{\sqrt{2\pi}}\right)\exp\left(\frac{-x^2}{2}\right).$$

(a) Show that $f(x)$ is non-negative for all x.
(b) By integration, show that

$$\int_{-\infty}^{\infty} f(x)dx = 1,$$

so that $f(x)$ is a probability density function (you can look up the integral; few people remember how to do this integral these days).
(c) Show that

$$\int_{-\infty}^{\infty} xf(x)dx = 0.$$

The easiest way to do this is to notice that $f(x) = f(-x)$
(d) Show that

$$\int_{-\infty}^{\infty} xf(x-\mu)dx = \mu.$$

The easiest way to do this is to change variables, and use the previous two exercises.
(e) Show that

$$\int_{-\infty}^{\infty} x^2 f(x)dx = 1.$$

You'll need to either do, or look up, the integral to do this exercise.

6.21. Write

$$g(x) = \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

Show that

$$\int_{-\infty}^{\infty} g(x)dx = \sqrt{2\pi}\sigma.$$

You can do this by a change of variable, and the results of the previous exercises.

6.22. Write

$$p(x) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)\exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right).$$

(a) Show that

$$\int_{-\infty}^{\infty} xp(x)dx = \mu$$

using the results of the previous exercises.

(b) Show that

$$\int_{-\infty}^{\infty} (x-\mu)^2 p(x) dx = \sigma^2$$

using the results of the previous exercises.

The Binomial Distribution for Large N

6.23. I flip a fair coin N times and count heads. We consider the probability that h, the fraction of heads, is in some range of numbers. For each of these questions, you should just write an expression, rather than evaluate the integral. *Hint:* If you know the range of numbers for h, you know the range for h/N.

(a) For $N = 1e6$, use the normal approximation to estimate $P(\{h \in [49500, 50500]\})$.

(b) For $N = 1e4$, use the normal approximation to estimate $P(\{h > 9000\})$.

(c) For $N = 1e2$, use the normal approximation to estimate $P(\{h > 60\} \cup \{h < 40\})$?

PROGRAMMING EXERCISES

6.24. An airline runs a regular flight with 10 seats on it. The probability that a passenger turns up for the flight is 0.95. What is the smallest number of seats the airline should sell to ensure that the probability the flight is full (i.e. 10 or more passengers turn up) is bigger than 0.99? You'll need to write a simple simulation; estimate the probability by counting.

6.25. You will plot a series of figures showing how the binomial distribution for large N increasingly "looks like" the normal distribution. We will consider the number of heads h in N flips of an unbiased coin (so $P(H) = P(T) = 1/2 = p$, and in this case $q = 1 - p = 1/2$). Write $x = \frac{h - Np}{\sqrt{Npq}}$.

(a) Prepare plots of the probability distribution of x for $N = 10$, $N = 30$, $N = 60$, and $N = 100$. These should be superimposed on the same set of axes. On this set of axes, you should also plot the normal probability distribution.

(b) Evaluate $P(\{x \geq 2\})$ for each case by summing over the appropriate terms in the binomial distribution. Now compare this to the prediction that the approximation would make, which is

$$\int_2^{\infty} \frac{1}{\sqrt{2\pi}} e^{[-u^2/2]} du.$$

You can obtain this number by appropriate evaluation of error functions.

(c) Now you will write a program to simulate coin flips and evaluate the variance of the simulated value of x for different numbers of flips. Again, the coin should be fair. For each N from $10, 40, 90, 160, 250, 490, 640, 810, 1000$, estimate the value of x by simulating that number of flips. You should run each simulation 100 times, and use the set of estimates to evaluate the variance of your estimate of x. Plot this variance against N and against $1/sqrtN$ - what do you see?

CHAPTER 7

Markov Chains and Hidden Markov Models

There are many situations where one must work with sequences. Here is a simple, and classical, example. We see a sequence of words, but the last word is missing. I will use the sequence "I had a glass of red wine with my grilled xxxx". What is the best guess for the missing word? You could obtain one possible answer by counting word frequencies, then replacing the missing word with the most common word. This is "the", which is not a particularly good guess because it doesn't fit with the previous word. Instead, you could find the most common pair of words matching "grilled xxxx", and then choose the second word. If you do this experiment (I used Google Ngram viewer, and searched for "grilled *"), you will find mostly quite sensible suggestions (I got "meats", "meat", "fish", "chicken", in that order). If you want to produce random sequences of words, the next word should depend on some of the words you have already produced.

7.1 MARKOV CHAINS

A sequence of random variables X_n is a **Markov chain** if it has the property that,

$$P(X_n = j|\text{values of all previous states}) = P(X_n = j|X_{n-1}),$$

or, equivalently, only the last state matters in determining the probability of the current state. The probabilities $P(X_n = j|X_{n-1} = i)$ are the **transition probabilities**. We will always deal with discrete random variables here, and we will assume that there is a finite number of states. For all our Markov chains, we will assume that

$$P(X_n = j|X_{n-1} = i) = P(X_{n-1} = j|X_{n-2} = i).$$

Formally, we focus on *discrete time, time homogenous Markov chains in a finite state space*. With enough technical machinery one can construct many other kinds of Markov chain.

One natural way to build Markov chains is to take a finite directed graph and label each directed edge from node i to node j with a probability. We interpret these probabilities as $P(X_n = j|X_{n-1} = i)$ (so the sum of probabilities over *outgoing* edges at any node must be 1). The Markov chain is then a **biased random walk** on this graph. A bug (or any other small object you prefer) sits on one of the graph's nodes. At each time step, the bug chooses one of the outgoing edges at random. The probability of choosing an edge is given by the probabilities on the drawing of the graph (equivalently, the transition probabilities). The bug then follows that edge. The bug keeps doing this until it hits an end state.

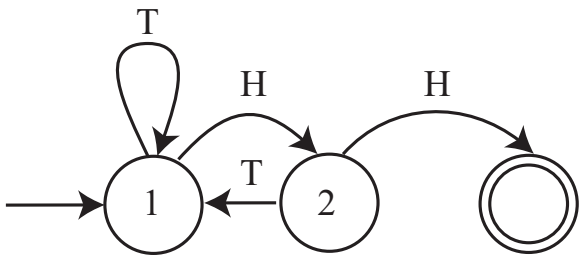

FIGURE 7.1: *A directed graph representing the coin flip example. By convention, the end state is a double circle, and the start state has an incoming arrow. I've labelled the arrows with the event that leads to the transition, but haven't bothered to put in the probabilities, because each is 0.5.*

Worked example 7.1 *Multiple Coin Flips*

You choose to flip a fair coin until you see two heads in a row, and then stop. Represent the resulting sequence of coin flips with a Markov chain. What is the probability that you flip the coin four times?

Solution: Figure 7.1 shows a simple drawing of the directed graph that represents the chain. The last three flips must have been THH (otherwise you'd go on too long, or end too early). But, because the second flip must be a T, the first could be either H or T. This means there are two sequences that work: $HTHH$ and $TTHH$. So $P(4 \text{ flips}) = 2/8 = 1/4$. We might want to answer significantly more interesting questions. For example, what is the probability that we must flip the coin more than 10 times? It is often possible to answer these questions by analysis, but we will use simulations.

Worked example 7.2 *Umbrellas*

I own one umbrella, and I walk from home to the office each morning, and back each evening. If it is raining (which occurs with probability p, and my umbrella is with me), I take it; if it is not raining, I leave the umbrella where it is. We exclude the possibility that it starts raining while I walk. Where I am, and whether I am wet or dry, forms a Markov chain. Draw a state machine for this Markov chain.

Solution: Figure 7.2 gives this chain. A more interesting question is with what probability I arrive at my destination wet? Again, we will solve this with simulation.

Notice an important difference between examples 7.1 and 7.2. In the coin flip case, the sequence of random variables can end (and your intuition likely tells you

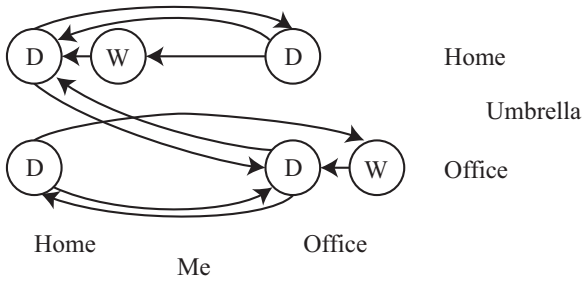

FIGURE 7.2: *A directed graph representing the umbrella example. Notice you can't arrive at the office wet with the umbrella at home (you'd have taken it), and so on. Labelling the edges with probabilities is left to the reader.*

it should do so reliably). We say the Markov chain has an **absorbing state** – a state that it can never leave. In the example of the umbrella, there is an infinite sequence of random variables, each depending on the last. Each state of this chain is **recurrent** – it will be seen repeatedly in this infinite sequence. One way to have a state that is not recurrent is to have a state with outgoing but no incoming edges.

Worked example 7.3 *The gambler's ruin*

Assume you bet 1 a tossed coin will come up heads. If you win, you get 1 and your original stake back. If you lose, you lose your stake. But this coin has the property that $P(H) = p < 1/2$. You have s when you start. You will keep betting until either (a) you have 0 (you are ruined; you can't borrow money) or (b) the amount of money you have accumulated is j, where $j > s$. The coin tosses are independent. The amount of money you have is a Markov chain. Draw the underlying state machine. Write $P(\text{ruined, starting with s}|p) = p_s$. It is straightforward that $p_0 = 1$, $p_j = 0$. Show that

$$p_s = pp_{s+1} + (1-p)p_{s-1}.$$

Solution: Figure 7.3 illustrates this example. The recurrence relation follows because the coin tosses are independent. If you win the first bet, you have $s+1$ and if you lose, you have $s-1$.

The gambler's ruin example illustrates some points that are quite characteristic of Markov chains. You can often write recurrence relations for the probability of various events. Sometimes you can solve them in closed form, though we will not pursue this thought further.

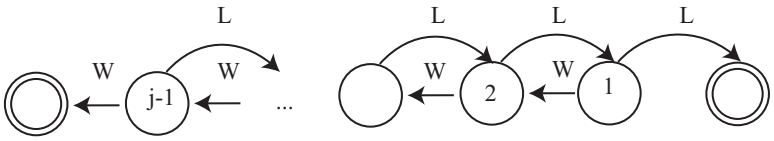

FIGURE 7.3: *A directed graph representing the gambler's ruin example. I have labelled each state with the amount of money the gambler has at that state. There are two end states, where the gambler has zero (is ruined), or has j and decides to leave the table. The problem we discuss is to compute the probability of being ruined, given the start state is s. This means that any state except the end states could be a start state. I have labelled the state transitions with "W" (for win) and "L" for lose, but have omitted the probabilities.*

Useful Facts: 7.1 *Markov chains*

A Markov chain is a sequence of random variables X_n with the property that,

$$P(X_n = j|\text{values of all previous states}) = P(X_n = j|X_{n-1}).$$

7.1.1 Transition Probability Matrices

Define the matrix $\mathcal{P}$ with $p_{ij} = P(X_n = j|X_{n-1} = i)$. Notice that this matrix has the properties that $p_{ij} \geq 0$ and

$$\sum_j p_{ij} = 1$$

because at the end of each time step the model must be in some state. Equivalently, the sum of transition probabilities for outgoing arrows is one. Non-negative matrices with this property are **stochastic matrices**. By the way, you should look very carefully at the i's and j's here — Markov chains are usually written in terms of *row* vectors, and this choice makes sense in that context.

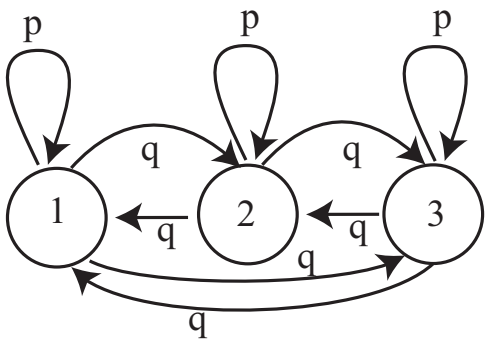

FIGURE 7.4: *A virus can exist in one of* 3 *strains. At the end of each year, the virus mutates. With probability* α*, it chooses uniformly and at random from one of the* 2 *other strains, and turns into that; with probability* $1-\alpha$*, it stays in the strain it is in. For this figure, we have transition probabilities* $p = (1-\alpha)$ *and* $q = (\alpha/2)$*.*

Worked example 7.4 *Viruses*

Write out the transition probability matrix for the virus of Figure 7.4, assuming that $\alpha = 0.2$.

Solution: We have $P(X_n = 1|X_{n-1} = 1) = (1-\alpha) = 0.8$, and $P(X_n = 2|X_{n-1} = 1) = \alpha/2 = P(X_n = 3|X_{n-1} = 1)$; so we get

$$\begin{pmatrix} 0.8 & 0.1 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.1 & 0.8 \end{pmatrix}$$

Now imagine we do not know the initial state of the chain, but instead have a probability distribution. This gives $P(X_0 = i)$ for each state i. It is usual to take these k probabilities and place them in a k-dimensional *row vector*, which is usually written π. From this information, we can compute the probability distribution over the states at time 1 by

$$\begin{aligned} P(X_1 = j) &= \sum_i P(X_1 = j, X_0 = i) \\ &= \sum_i P(X_1 = j|X_0 = i)P(X_0 = i) \\ &= \sum_i p_{ij}\pi_i. \end{aligned}$$

If we write $\mathbf{p}^{(n)}$ for the row vector representing the probability distribution of the

state at step n, we can write this expression as

$$\mathbf{p}^{(1)} = \pi\mathcal{P}.$$

Now notice that

$$\begin{aligned} P(X_2 = j) &= \sum_i P(X_2 = j, X_1 = i) \\ &= \sum_i P(X_2 = j | X_1 = i) P(X_1 = i) \\ &= \sum_i p_{ij} \left(\sum_{ki} p_{ki} \pi_k \right). \end{aligned}$$

so that

$$\mathbf{p}^{(n)} = \pi\mathcal{P}^n.$$

This expression is useful for simulation, and also allows us to deduce a variety of interesting properties of Markov chains.

Useful Facts: 7.2 *Transition probability matrices*

A finite state Markov chain can be represented with a matrix $\mathcal{P}$ of transition probabilities, where the i, j'th element $p_{ij} = P(X_n = j | X_{n-1} = i)$. This matrix is a stochastic matrix. If the probability distribution of state X_{n-1} is represented by π_{n-1}, then the probability distribution of state X_n is given by $\pi_{n-1}^T \mathcal{P}$.

7.1.2 Stationary Distributions

Worked example 7.5 *Viruses*

We know that the virus of Figure 7.4 started in strain 1. After two state transitions, what is the distribution of states when $\alpha = 0.2$? when $\alpha = 0.9$? What happens after 20 state transitions? If the virus starts in strain 2, what happens after 20 state transitions?

Solution: If the virus started in strain 1, then $\pi = [1, 0, 0]$. We must compute $\pi(\mathcal{P}(\alpha))^2$. This yields $[0.66, 0.17, 0.17]$ for the case $\alpha = 0.2$ and $[0.4150, 0.2925, 0.2925]$ for the case $\alpha = 0.9$. Notice that, because the virus with small α tends to stay in whatever state it is in, the distribution of states after two years is still quite peaked; when α is large, the distribution of states is quite uniform. After 20 transitions, we have $[0.3339, 0.3331, 0.3331]$ for the case $\alpha = 0.2$ and $[0.3333, 0.3333, 0.3333]$ for the case $\alpha = 0.9$; you will get similar numbers even if the virus starts in strain 2. After 20 transitions, the virus has largely "forgotten" what the initial state was.

In example 7.5, the distribution of virus strains after a long interval appears not to depend much on the initial strain. This property is true of many Markov chains. Assume that our chain has a finite number of states. Assume that any state can be reached from any other state, by some sequence of transitions. Such chains are called **irreducible**. Notice this means there is no absorbing state, and the chain cannot get "stuck" in a state or a collection of states. Then there is a unique vector $\mathbf{s}$, usually referred to as the **stationary distribution**, such that for *any* initial state distribution π,

$$\lim_{n \to \infty} \pi \mathcal{P}^{(n)} = \mathbf{s}.$$

Equivalently, if the chain has run through many steps, it no longer matters what the initial distribution is. The probability distribution over states will be $\mathbf{s}$.

The stationary distribution can often be found using the following property. Assume the distribution over states is $\mathbf{s}$, and the chain goes through one step. Then the new distribution over states must be $\mathbf{s}$ too. This means that

$$\mathbf{s}\mathcal{P} = \mathbf{s}$$

so that $\mathbf{s}$ is an eigenvector of $\mathcal{P}^T$, with eigenvalue 1. It turns out that, for an irreducible chain, there is exactly one such eigenvector.

The stationary distribution is a useful idea in applications. It allows us to answer quite natural questions, without conditioning on the initial state of the chain. For example, in the umbrella case, we might wish to know the probability I arrive home wet. This could depend on where the chain starts (example 7.6). If you look at the figure, the Markov chain is irreducible, so there is a stationary distribution and (as long as I've been going back and forth long enough for the chain to "forget" where it started), the probability it is in a particular state doesn't

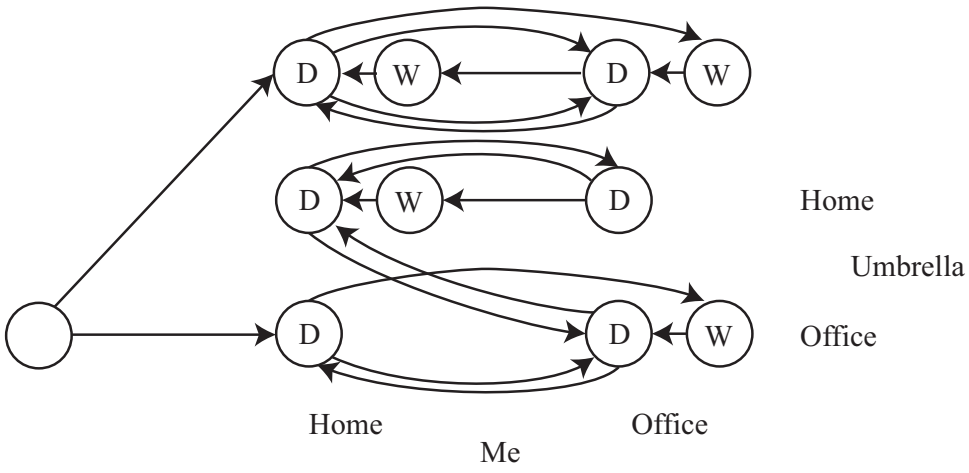

FIGURE 7.5: *In this umbrella example, there can't be a stationary distribution; what happens depends on the initial, random choice of buying/not buying an umbrella.*

depend on where it started. So the most sensible interpretation of this probability is the probability of a particular state in the stationary distribution.

Worked example 7.6 *Umbrellas, but without a stationary distribution*

This is a different version of the umbrella problem, but with a crucial difference. When I move to town, I decide randomly to buy an umbrella with probability 0.5. I then go from office to home and back. If I have bought an umbrella, I behave as in example 7.2. If I have not, I just get wet. Illustrate this Markov chain with a state diagram.

Solution: Figure 7.5 does this. Notice this chain *isn't* irreducible. The state of the chain in the far future depends on where it started (i.e. did I buy an umbrella or not).

Useful Facts: 7.3 *Many Markov chains have stationary distributions*

If a Markov chain has a finite set of states, and if it is possible to get from any state to any other state, then the chain will have a stationary distribution. A sample state of the chain taken after it has been running for a long time will be a sample from that stationary distribution. Once the chain has run for long enough, it will visit states with a frequency corresponding to that stationary distribution, though it may take many state transitions to move from state to state.

7.1.3 Example: Markov Chain Models of Text

Imagine we wish to model English text. The very simplest model would be to estimate individual letter frequencies (most likely, by counting letters in a large body of example text). We might count spaces and punctuation marks as letters. We regard the frequencies as probabilities, then model a sequence by repeatedly drawing a letter from that probability model. You could even punctuate with this model by regarding punctuation signs as letters, too. We expect this model will produce sequences that are poor models of English text – there will be very long strings of 'a's, for example. This is clearly a (rather dull) Markov chain. It is sometimes referred to as a 0-th order chain or a 0-th order model, because each letter depends on the 0 letters behind it.

A slightly more sophisticated model would be to work with pairs of letters. Again, we would estimate the frequency of pairs by counting letter pairs in a body of text. We could then draw a first letter from the letter frequency table. Assume this is an 'a'. We would then draw the second letter by drawing a sample from the conditional probability of encountering each letter after 'a', which we could compute from the table of pair frequencies. Assume this is an 'n'. We get the third letter by drawing a sample from the conditional probability of encountering each letter after 'n', which we could compute from the table of pair frequencies, and so on. This is a first order chain (because each letter depends on the one letter behind it).

Second and higher order chains (or models) follow the general recipe, but the probability of a letter depends on more of the letters behind it. You may be concerned that conditioning a letter on the two (or k) previous letters means we don't have a Markov chain, because I said that the n'th state depends on only the $n-1$'th state. The cure for this concern is to use states that represent two (or k) letters, and adjust transition probabilities so that the states are consistent. So for a second order chain, the string "abcde" is a sequence of four states, "ab", "bc", "cd", and "de".

Worked example 7.7 *Modelling short words*

Obtain a text resource, and use a trigram letter model to produce four letter words. What fraction of bigrams (resp. trigrams) do not occur in this resource? What fraction of the words you produce are actual words?

Solution: I used the text of a draft of this chapter. I ignored punctuation marks, and forced capital letters to lower case letters. I found 0.44 of the bigrams and 0.90 of the trigrams were not present. I built two models. In one, I just used counts to form the probability distributions (so there were many zero probabilities). In the other, I split a probability of 0.1 between all the cases that had not been observed. A list of 20 word samples from the first model is: "ngen", "ingu", "erms", "isso", "also", "plef", "trit", "issi", "stio", "esti", "coll", "tsma", "arko", "llso", "bles", "uati", "namp", "call", "riat", "eplu"; two of these are real English words (three if you count "coll", which I don't; too obscure), so perhaps 10% of the samples are real words. A list of 20 word samples from the second model is: "hate", "ther", "sout", "vect", "nces", "ffer", "msua", "ergu", "blef", "hest", "assu", "fhsp", "ults", "lend", "lsoc", "fysj", "uscr", "ithi", "prow", "lith"; four of these are real English words (you might need to look up "lith", but I refuse to count "hest" as being too archaic), so perhaps 20% of the samples are real words. In each case, the samples are too small to take the fraction estimates all that seriously.

Letter models can be good enough for (say) evaluating communication devices, but they're not great at producing words (example 7.7). More effective language models are obtained by working with words. The recipe is as above, but now we use words in place of letters. It turns out that this recipe applies to such domains as protein sequencing, dna sequencing and music synthesis as well, but now we use amino acids (resp. base pairs; notes) in place of letters. Generally, one decides what the basic item is (letter, word, amino acid, base pair, note, etc.). Then individual items are called **unigrams** and 0'th order models are **unigram models**; pairs are **bigrams** and first order models are **bigram models**; triples are **trigrams**, second order models **trigram models**; and for any other n, groups of n in sequence are **n-grams** and $n-1$'th order models are **n-gram models**.

Worked example 7.8 *Modelling text with n-grams of words*

Build a text model that uses bigrams (resp. trigrams, resp. n-grams) of words, and look at the paragraphs that your model produces.

Solution: This is actually a fairly arduous assignment, because it is hard to get good bigram frequencies without working with enormous text resources. Google publishes n-gram models for English words with the year in which the n-gram occurred and information about how many different books it occurred in. So, for example, the word "circumvallate" appeared 335 times in 1978, in 91 distinct books – some books clearly felt the need to use this term more than once. This information can be found starting at http://storage.googleapis.com/books/ngrams/books/datasetsv2.html. The raw dataset is huge, as you would expect. There are numerous n-gram language models on the web. Jeff Attwood has a brief discussion of some models at https://blog.codinghorror.com/markov-and-you/; Sophie Chou has some examples, and pointers to code snippets and text resources, at http://blog.sophiechou.com/2013/how-to-model-markov-chains/. Fletcher Heisler, Michael Herman, and Jeremy Johnson are authors of RealPython, a training course in Python, and give a nice worked example of a Markov chain language generator at https://realpython.com/blog/python/lyricize-a-flask-app-to-create-lyrics-using-markov-chains/. Markov chain language models are effective tools for satire. Garkov is Josh Millard's tool for generating comics featuring a well-known cat (at http://joshmillard.com/garkov/). There's a nice Markov chain for reviewing wines by Tony Fischetti at http://www.onthelambda.com/2014/02/20/how-to-fake-a-sophisticated-knowledge-of-wine-with-markov-chains/

It is usually straightforward to build a unigram model, because it is usually easy to get enough data to estimate the frequencies of the unigrams. There are many more bigrams than unigrams, many more trigrams than bigrams, and so on. This means that estimating frequencies can get tricky. In particular, you might need to collect an immense amount of data to see every possible n-gram several times. Without seeing every possible n-gram several times, you will need to deal with estimating the probability of encountering rare n-grams *that you haven't seen*. Assigning these n-grams a probability of zero is unwise, because that implies that they *never* occur, as opposed to occur seldom.

There are a variety of schemes for **smoothing** data (essentially, estimating the probability of rare items that have not been seen). The simplest one is to assign some very small fixed probability to every n-gram that has a zero count. It turns out that this is not a particularly good approach, because, for even quite small n, the fraction of n-grams that have zero count can be very large. In turn, you can find that most of the probability in your model is assigned to n-grams you have never seen. An improved version of this model assigns a fixed probability to unseen n-grams, then divides that probability up between all of the n-grams that

have never been seen before. This approach has its own characteristic problems. It ignores evidence that some of the unseen n-grams are more common than others. Some of the unseen n-grams have (n-1) leading terms that are (n-1)-grams that we *have* observed. These (n-1)-grams likely differ in frequency, suggesting that n-grams involving them should differ in frequency, too. More sophisticated schemes are beyond our scope, however.

7.2 ESTIMATING PROPERTIES OF MARKOV CHAINS

Many problems in probability can be worked out in closed form if one knows enough combinatorial mathematics, or can come up with the right trick. Textbooks are full of these, and we've seen some. Explicit formulas for probabilities are often extremely useful. But it isn't always easy or possible to find a formula for the probability of an event in a model. Markov chains are a particularly rich source of probability problems that might be too much trouble to solve in closed form. An alternative strategy is to build a simulation, run it many times, and count the fraction of outcomes where the event occurs. This is a simulation experiment.

7.2.1 Simulation

Imagine we have a random variable X with probability distribution $P(X)$ that takes values in some domain D. Assume that we can easily produce independent simulations, and that we wish to know $\mathbb{E}[f]$, the expected value of the function f under the distribution $P(X)$.

The weak law of large numbers tells us how to proceed. Define a new random variable $F = f(X)$. This has a probability distribution $P(F)$, which might be difficult to know. We want to estimate $\mathbb{E}[f]$, the expected value of the function f under the distribution $P(X)$. This is the same as $\mathbb{E}[F]$. Now if we have a set of IID samples of X, which we write x_i, then we can form a set of IID samples of F by forming $f(x_i) = f_i$. Write

$$F_N = \frac{\sum_{i=1}^{N} f_i}{N}.$$

This is a random variable, and the weak law of large numbers gives that, for any positive number ϵ

$$\lim_{N \to \infty} P(\{\|F_N - \mathbb{E}[F]\| > \epsilon\}) = 0.$$

You can interpret this as saying that, that for a set of IID random samples x_i, the probability that

$$\frac{\sum_{i=1}^{N} f(x_i)}{N}$$

is very close to $\mathbb{E}[f]$ is high for large N

Worked example 7.9 *Computing an Expectation*

Assume the random variable X is uniformly distributed in the range $[0-1]$, and the random variable Y is uniformly distributed in the range $[0-10]$. X and Z are independent. Write $Z = (Y - 5X)^3 - X^2$. What is $\mathsf{var}(\{Z\})$?

Solution: With enough work, one could probably work this out in closed form. An easy program will get a good estimate. We have that $\mathsf{var}(\{Z\}) = \mathbb{E}\left[Z^2\right] - \mathbb{E}[Z]^2$. My program computed 1000 values of Z (by drawing X and Y from the appropriate random number generator, then evaluating the function). I then computed $\mathbb{E}[Z]$ by averaging those values, and $\mathbb{E}[Z]^2$ by averaging their squares. For a run of my program, I got $\mathsf{var}(\{Z\}) = 2.76 \times 10^4$.

You can compute a probability using a simulation, too, because a probability can be computed by taking an expectation. Recall the property of indicator functions that

$$\mathbb{E}\left[\mathbb{I}_{[\mathcal{E}]}\right] = P(\mathcal{E})$$

(Section 5.3.3). This means that computing the probability of an event $\mathcal{E}$ involves writing a function that is 1 when the event occurs, and 0 otherwise; we then estimate the expected value of that function.

Worked example 7.10 *Computing a Probability for Multiple Coin Flips*

You flip a fair coin three times. Use a simulation to estimate the probability that you see three H's.

Solution: You really should be able to work this out in closed form. But it's amusing to check with a simulation. I wrote a simple program that obtained a 1000x3 table of uniformly distributed random numbers in the range $[0-1]$. For each number, if it was greater than 0.5 I recorded an H and if it was smaller, I recorded a T. Then I counted the number of rows that had 3 H's (i.e. the expected value of the relevant indicator function). This yielded the estimate 0.127, which compares well to the right answer.

Worked example 7.11 *Computing a Probability*

Assume the random variable X is uniformly distributed in the range $[0-1]$, and the random variable Y is uniformly distributed in the range $[0-10]$. Write $Z = (Y - 5X)^3 - X^2$. What is $P(\{Z > 3\})$?

Solution: With enough work, one could probably work this out in closed form. An easy program will get a good estimate. My program computed 1000 values of Z (by drawing X and Y from the appropriate random number generator, then evaluating the function) and counted the fraction of Z values that was greater than 3 (which is the relevant indicator function). For a run of my program, I got $P(\{Z > 3\}) \approx 0.619$

For all the examples we will deal with, producing an IID sample of the relevant probability distribution will be straightforward. You should be aware that it can be very hard to produce an IID sample from an arbitrary distribution, particularly if that distribution is over a continuous variable in high dimensions.

7.2.2 Simulation Results as Random Variables

The estimate of a probability or of an expectation that comes out of a simulation experiment is a random variable, because it is a function of random numbers. If you run the simulation again, you'll get a different value, unless you did something silly with the random number generator. Generally, you should expect this random variable to have a normal distribution. You can check this by constructing a histogram over a large number of runs. The mean of this random variable is the parameter you are trying to estimate. It is useful to know that this random variable tends to be normal, because it means the standard deviation of the random variable tells you a lot about the likely values you will observe.

Another helpful rule of thumb, which is almost always right, is that the standard deviation of this random variable behaves like

$$\frac{C}{\sqrt{N}}$$

where C is a constant that depends on the problem and can be very hard to evaluate, and N is the number of runs of the simulation. What this means is that if you want to (say) double the accuracy of your estimate of the probability or the expectation, you have to run four times as many simulations. Very accurate estimates are tough to get, because they require immense numbers of simulation runs.

Figure 7.6 shows how the result of a simulation behaves when the number of runs changes. I used the simulation of example 7.11, and ran multiple experiments for each of a number of different samples (i.e. 100 experiments using 10 samples; 100 using 100 samples; and so on).

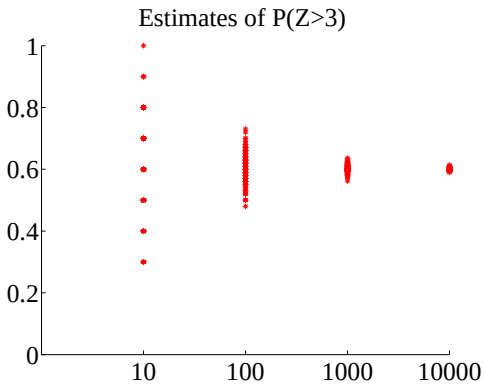

FIGURE 7.6: *Estimates of the probability from example 7.11, obtained from different runs of my simulator using different numbers of samples. In each case, I used 100 runs; the number of samples is shown on the horizontal axis. You should notice that the estimate varies pretty widely when there are only 10 samples in each run, but the variance (equivalently, the size of the spread) goes down sharply as the number of samples per run increases to 1000. Because we expect these estimates to be roughly normally distributed, the variance gives a good idea of how accurate the original probability estimate is.*

Worked example 7.12 *Getting 14's with 20-sided dice*

You throw 3 fair 20-sided dice. Estimate the probability that the sum of the faces is 14 using a simulation. Use $N = [1e1, 1e2, 1e3, 1e4, 1e5, 1e6]$. Which estimate is likely to be more accurate, and why?

Solution: You need a fairly fast computer, or this will take a long time. I ran ten versions of each experiment for $N = [1e1, 1e2, 1e3, 1e4, 1e5, 1e6]$, yielding ten probability estimates for each N. These were different for each version of the experiment, because the simulations are random. I got means of $[0, 0.0030, 0.0096, 0.0100, 0.0096, 0.0098]$, and standard deviations of [00.00670.00330.00090.00020.0001]. This suggests the true value is around 0.0098, and the estimate from $N = 1e6$ is best. The reason that the estimate with $N = 1e1$ is 0 is that the probability is very small, so you don't usually observe this case at all in only ten trials.

Small probabilities can be rather hard to estimate, as we would expect. In the case of example 7.11, let us estimate $P(\{Z > 950\})$. A few moments with a computer will show that this probability is of the order of 1e-3 to 1e-4. I obtained a million different simulated values of Z from my program, and saw 310 where $Z > 950$. This means that to know this probability to, say, three digits of numerical accuracy might involve a daunting number of samples. Notice that this does not contradict the rule of thumb that the standard deviation of the random variable

defined by a simulation estimate behaves like $\frac{C}{\sqrt{N}}$; it's just that in this case, C is very large indeed.

Useful Facts: 7.4 *The properties of simulations*

You should remember that

- The weak law of large numbers means you can estimate expectations and probabilities with a simulation.
- The result of a simulation is usually a normal random variable.
- The expected value of this random variable is usually the true value of the expectation or probability you are trying to simulate.
- The standard deviation of this random variable is usually $\frac{C}{\sqrt{N}}$, where N is the number of examples in the simulation and C is a number usually too hard to estimate.

Worked example 7.13 *Comparing simulation with computation*

You throw 3 fair six-sided dice. You wish to know the probability the sum is 3. Compare the true value of this probability with estimates from six runs of a simulation using $N = 10000$. What conclusions do you draw?

Solution: I ran six simulations with $N = 10000$, and got $[0.0038, 0.0038, 0.0053, 0.0041, 0.0056, 0.0049]$. The mean is 0.00458, and the standard deviation is 0.0007, which suggests the estimate isn't that great, but the right answer should be in the range $[0.00388, 0.00528]$ with probability about 0.68. The true value is $1/216 \approx 0.00463$. The estimate is tolerable, but not super accurate.

7.2.3 Simulating Markov Chains

We will always assume that we know the states and transition probabilities of the Markov chain. Properties that might be of interest in this case include: the probability of hitting an absorbing state; the expected time to go from one state to another; the expected time to hit an absorbing state; and which states have high probability under the stationary distribution.

Worked example 7.14 *Coin Flips with End Conditions*

I flip a coin repeatedly until I encounter a sequence HTHT, at which point I stop. What is the probability that I flip the coin nine times?

Solution: You might well be able to construct a closed form solution to this if you follow the details of example 13.1 and do quite a lot of extra work. A simulation is really straightforward to write; notice you can save time by not continuing to simulate coin flips once you've flipped past nine times. I got 0.0411 as the mean probability over 10 runs of a simulation of 1000 experiments each, with a standard deviation of 0.0056.

Worked example 7.15 *A Queue*

A bus is supposed to arrive at a bus stop every hour for 10 hours each day. The number of people who arrive to queue at the bus stop each hour has a Poisson distribution, with intensity 4. If the bus stops, everyone gets on the bus and the number of people in the queue becomes zero. However, with probability 0.1 the bus driver decides not to stop, in which case people decide to wait. If the queue is ever longer than 15, the waiting passengers will riot (and then immediately get dragged off by the police, so the queue length goes down to zero). What is the expected time between riots?

Solution: I'm not sure whether one could come up with a closed form solution to this problem. A simulation is completely straightforward to write. I get a mean time of 441 hours between riots, with a standard deviation of 391. It's interesting to play around with the parameters of this problem; a less conscientious bus driver, or a higher intensity arrival distribution, lead to much more regular riots.

Worked example 7.16 *Inventory*

A store needs to control its stock of an item. It can order stocks on Friday evenings, which will be delivered on Monday mornings. The store is old-fashioned, and open only on weekdays. On each weekday, a random number of customers comes in to buy the item. This number has a Poisson distribution, with intensity 4. If the item is present, the customer buys it, and the store makes 100; otherwise, the customer leaves. Each evening at closing, the store loses 10 for each unsold item on its shelves. The store's supplier insists that it order a fixed number k of items (i.e. the store must order k items each week). The store opens on a Monday with 20 items on the shelf. What k should the store use to maximise profits?

Solution: I'm not sure whether one could come up with a closed form solution to this problem, either. A simulation is completely straightforward to write. To choose k, you run the simulation with different k values to see what happens. I computed accumulated profits over 100 weeks for different k values, then ran the simulation five times to see which k was predicted. Results were $21, 19, 23, 20, 21$. I'd choose 21 based on this information.

For example 7.16, you should plot accumulated profits. If k is small, the store doesn't lose money by storing items, but it doesn't sell as much stuff as it could; if k is large, then it can fill any order but it loses money by having stock on the shelves. A little thought will convince you that k should be near 20, because that is the expected number of customers each week, so $k = 20$ means the store can expect to sell all its new stock. It may not be exactly 20, because it must depend a little on the balance between the profit in selling an item and the cost of storing it. For example, if the cost of storing items is very small compared to the profit, an very large k might be a good choice. If the cost of storage is sufficiently high, it might be better to never have anything on the shelves; this point explains the absence of small stores selling PC's.

Quite substantial examples are possible. The game "snakes and ladders" involves random walk on Markov chain. If you don't know this game, look it up; it's sometimes called "chutes and ladders", and there is an excellent Wikipedia page. The state is given by where each players' token is on the board, so on a 10x10 board one player involves 100 states, two players 100^2 states, and so on. The set of states is finite, though big. Transitions are random, because each player throws dice. The snakes (resp. ladders) represent extra edges in the directed graph. Absorbing states occur when a player hits the top square. It is straightforward to compute the expected number of turns for a given number of players by simulation, for example. For one commercial version, the Wikipedia page gives the crucial numbers: for two players, expected number of moves to a win is 47.76, and the first player wins with probability 0.509. Notice you might need to think a bit about how to write the program if there were, say, 8 players on a 12x12 board – you would likely avoid storing the entire state space.

7.3 EXAMPLE: RANKING THE WEB BY SIMULATING A MARKOV CHAIN

Perhaps the most valuable technical question of the last thirty years has been: Which web pages are interesting? Some idea of the importance of this question is that it was only really asked about 20 years ago, and at least one gigantic technology company has been spawned by a partial answer. This answer, due to Larry Page and Sergey Brin, and widely known as PageRank, revolves around simulating the stationary distribution of a Markov chain.

You can think of the world wide web as a directed graph. Each page is a state. Directed edges from page to page represent links. Count only the first link from a page to another page. Some pages are linked, others are not. We want to know how important each page is.

One way to think about importance is to think about what a random web surfer would do. The surfer can either (a) choose one of the outgoing links on a page at random, and follow it or (b) type in the URL of a new page, and go to that instead. This is a random walk on a directed graph. We expect that this random surfer should see a lot of pages that have lots of incoming links from other pages that have lots of incoming links that (and so on). These pages are important, because lots of pages have linked to them.

For the moment, ignore the surfer's option to type in a URL. Write $r(i)$ for the importance of the i'th page. We model importance as leaking from page to page across outgoing links (the same way the surfer jumps). Page i receives importance down each incoming link. The amount of importance is proportional to the amount of importance at the other end of the link, and inversely proportional to the number of links leaving that page. So a page with only one outgoing link transfers all its importance down that link; and the way for a page to receive a lot of importance is for it to have a lot of important pages link to it alone. We write

$$r(j) = \sum_{i \to j} \frac{r(i)}{|\,i\,|}$$

where $|\,i\,|$ means the total number of links pointing *out* of page i. We can stack the $r(j)$ values into a *row* vector $\mathbf{r}$, and construct a matrix $\mathcal{P}$, where

$$p_{ij} = \begin{cases} \frac{1}{|i|} & \text{if } i \text{ points to } j \\ 0 & \text{otherwise} \end{cases}$$

With this notation, the importance vector has the property

$$\mathbf{r} = \mathbf{r}\mathcal{P}$$

and should look a bit like the stationary distribution of a random walk to you, except that $\mathcal{P}$ isn't stochastic — there may be some rows where the row sum of $\mathcal{P}$ is zero, because there are *no* outgoing links from that page. We can fix this easily by replacing each row that sums to zero with $(1/n)\mathbf{1}$, where n is the total number of pages. Call the resulting matrix $\mathcal{G}$ (it's quite often called the **raw Google matrix**).

The web has pages with no outgoing links (which we've dealt with), pages with no incoming links, and even pages with no links at all. A random walk could

get trapped by moving to a page with no outgoing links. Allowing the surfer to randomly enter a URL sorts out all of these problems, because it inserts an edge of small weight from every node to every other node. Now the random walk cannot get trapped.

There are a variety of possible choices for the weight of these inserted edges. The original choice was to make each inserted edge have the same weight. Write $\mathbf{1}$ for the n dimensional column vector containing a 1 in each component, and let $0 < \alpha < 1$. We can write the matrix of transition probabilities as

$$\mathcal{G}(\alpha) = \alpha\frac{(\mathbf{1}\mathbf{1}^T)}{n} + (1-\alpha)\mathcal{G}$$

where $\mathcal{G}$ is the original Google matrix. An alternative choice is to choose a weight for each web page. This weight could come from a query; from advertising revenues; or from page visit statistics. Google keeps quiet about the details. Write this weight vector $\mathbf{v}$, and require that $\mathbf{1}^T\mathbf{v} = 1$ (i.e. the coefficients sum to one). Then we could have

$$\mathcal{G}(\alpha, \mathbf{v}) = \alpha\frac{(\mathbf{1}\mathbf{v}^T)}{n} + (1-\alpha)\mathcal{G}.$$

Now the importance vector $\mathbf{r}$ is the (unique, though I won't prove this) *row* vector $\mathbf{r}$ such that

$$\mathbf{r} = \mathbf{r}\mathcal{G}(\alpha, \mathbf{v}).$$

How do we compute this vector? One natural algorithm is to estimate $\mathbf{r}$ with a random walk, because $\mathbf{r}$ is the stationary distribution of a Markov chain. If we simulate this walk for many steps, the probability that the simulation is in state j should be $r(j)$, at least approximately.

This simulation is easy to build. Imagine our random walking bug sits on a web page. At each time step, it transitions to a new page by either (a) picking from all existing pages at random, using $\mathbf{v}$ as a probability distribution on the pages (which it does with probability α); or (b) chooses one of the outgoing links uniformly and at random, and follows it (which it does with probability $1-\alpha$). The stationary distribution of this random walk is $\mathbf{r}$. Another fact that I shall not prove is that, when α is sufficiently large, this random walk very quickly "forgets" it's initial distribution. As a result, you can estimate the importance of web pages by starting this random walk in a random location; letting it run for a bit; then stopping it, and collecting the page you stopped on. The pages you see like this are independent, identically distributed samples from $\mathbf{r}$; so the ones you see more often are more important, and the ones you see less often are less important.

7.4 HIDDEN MARKOV MODELS AND DYNAMIC PROGRAMMING

Imagine we wish to build a program that can transcribe speech sounds into text. Each small chunk of text can lead to one, or some, sounds, and some randomness is involved. For example, some people pronounce the word "fishing" rather like "fission". As another example, the word "scone" is sometimes pronounced rhyming with "stone", and sometimes rhyming with "gone". A Markov chain supplies a model of all possible text sequences, and allows us to compute the probability of any particular sequence. We will use a Markov chain to model text sequences, but

what we observe is sound. We must have a model of how sound is produced by text. With that model and the Markov chain, we want to produce text that (a) is a likely sequence of words and (b) is likely to have produced the sounds we hear.

Many applications contain the main elements of this example. We might wish to transcribe music from sound. We might wish to understand American sign language from video. We might wish to produce a written description of how someone moves from video observations. We might wish to break a substitution cipher. In each case, what we want to recover is a sequence that can be modelled with a Markov chain, but we don't see the states of the chain. Instead, we see noisy measurements that *depend* on the state of the chain, and we want to recover a state sequence that is (a) likely under the Markov chain model and (b) likely to have produced the measurements we observe.

7.4.1 Hidden Markov Models

Assume we have a finite state, time homogenous Markov chain, with S states. This chain will start at time 1, and the probability distribution $P(X_1 = i)$ is given by the vector π. At time u, it will take the state X_u, and its transition probability matrix is $p_{ij} = P(X_{u+1} = j | X_u = i)$. We do not observe the state of the chain. Instead, we observe some Y_u. We will assume that Y_u is also discrete, and there are a total of O possible states for Y_u for any u. We can write a probability distribution for these observations $P(Y_u | X_u = i) = q_i(Y_u)$. This distribution is the **emission distribution** of the model. For simplicity, we will assume that the emission distribution does not change with time.

We can arrange the emission distribution into a matrix $\mathcal{Q}$. A **hidden Markov model** consists of the transition probability distribution for the states, the relationship between the state and the probability distribution on Y_u, and the initial distribution on states, that is, $(\mathcal{P}, \mathcal{Q}, \pi)$. These models are often dictated by an application. An alternative is to build a model that best fits a collection of observed data, but doing so requires technical machinery we cannot expound here.

I will sketch how one might build a model for transcribing speech, but you should keep in mind this is just a sketch of a very rich area. We can obtain the probability of a word following some set of words using n-gram resources, as in section 7.1.3. We then build a model of each word in terms of small chunks of word that are likely to correspond to common small chunks of sound. We will call these chunks of sound **phonemes**. We can look up the different sets of phonemes that correspond to a word using a pronunciation dictionary. We can combine these two resources into a model of how likely it is one will pass from one phoneme inside a word to another, which might either be inside this word or inside another word. We now have $\mathcal{P}$. We will not spend much time on π, and might even model it as a uniform distribution. We can use a variety of strategies to build $\mathcal{Q}$. One is to build discrete features of a sound signal, then count how many times a particular set of features is produced when a particular phoneme is played.

7.4.2 Picturing Inference with a Trellis

Assume that we have a sequence of N measurements Y_i that we believe to be the output of a known hidden Markov model. We wish to recover the "best"

corresponding sequence of X_i. Doing so is inference. We will choose to recover a sequence X_i that maximises

$$\log P(X_1, X_2, \ldots, X_N | Y_1, Y_2, \ldots, Y_N, \mathcal{P}, \mathcal{Q}, \pi)$$

which is

$$\log \left(\frac{P(X_1, X_2, \ldots, X_N, Y_1, Y_2, \ldots, Y_N | \mathcal{P}, \mathcal{Q}, \pi)}{P(Y_1, Y_2, \ldots, Y_N)} \right)$$

and this is

$$\log P(X_1, X_2, \ldots, X_N, Y_1, Y_2, \ldots, Y_N | \mathcal{P}, \mathcal{Q}, \pi) - \log P(Y_1, Y_2, \ldots, Y_N).$$

Notice that $P(Y_1, Y_2, \ldots, Y_N)$ doesn't depend on the sequence of X_u we choose, and so the second term can be ignored. What is important here is that we can decompose $\log P(X_1, X_2, \ldots, X_N, Y_1, Y_2, \ldots, Y_N | \mathcal{P}, \mathcal{Q}, \pi)$ in a very useful way, because the X_u form a Markov chain. We want to maximise

$$\log P(X_1, X_2, \ldots, X_N, Y_1, Y_2, \ldots, Y_N | \mathcal{P}, \mathcal{Q}, \pi)$$

but this is

$$\begin{aligned} &\log P(X_1) + \log P(Y_1|X_1) + \\ &\log P(X_2|X_1) + \log P(Y_2|X_2) + \\ &\ldots \\ &\log P(X_N|X_{n-1}) + \log P(Y_N|X_N). \end{aligned}$$

Notice that this cost function has an important structure. It is a sum of terms. There are terms that depend on a single X_i (unary terms) and terms that depend on two (binary terms). Any state X_i appears in at most two binary terms.

We can illustrate this cost function in a structure called a **trellis**. This is a weighted, directed graph consisting of N copies of the state space, which we arrange in columns. There is a column corresponding to each measurement. We add a directed arrow from any state in the u'th column to any state in the $u+1$'th column if the transition probability between the states isn't 0. This represents the fact that there is a possible transition between these states. We then label the trellis with weights. We weight the node representing the case that state $X_u = j$ in the column corresponding to Y_u with $\log P(Y_u|X_u = j)$. We weight the arc from the node representing $X_u = i$ to that representing $X_{u+1} = j$ with $\log P(X_{u+1} = j|X_u = i)$.

The trellis has two crucial properties. Each directed path through the trellis from the start column to the end column represents a legal sequence of states. Now for some directed path from the start column to the end column, sum all the weights for the nodes and edges along this path. This sum is the log of the joint probability of that sequence of states with the measurements. You can verify each of these statements easily by reference to a simple example (try Figure 7.7)

There is an efficient algorithm for finding the path through a trellis which maximises the sum of terms. The algorithm is usually called **dynamic programming** or the **Viterbi algorithm**. I will describe this algorithm both in narrative, and as a recursion. We want to find the best path from each node in the first

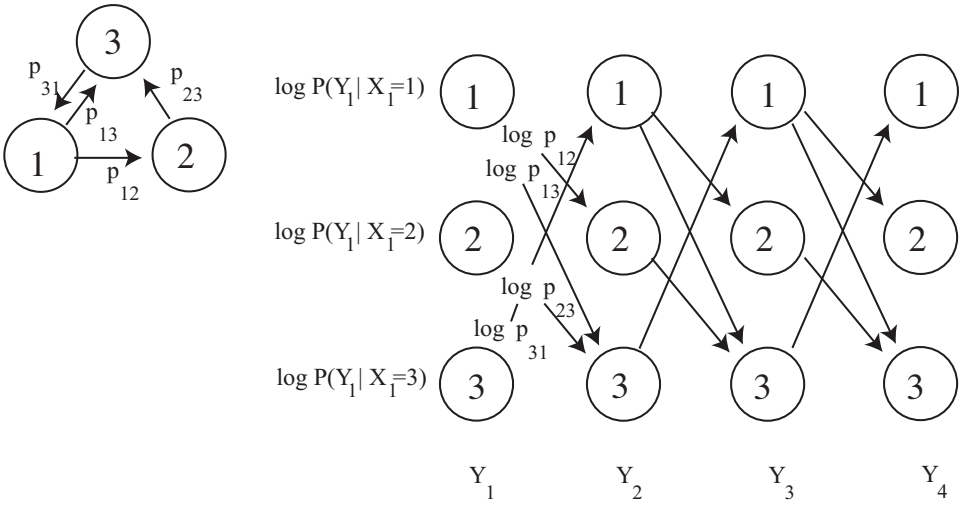

FIGURE 7.7: *At the* **top left**, *a simple state transition model. Each outgoing edge has some probability, though the topology of the model forces two of these probabilities to be* 1. *Below, the trellis corresponding to that model. Each path through the trellis corresponds to a legal sequence of states, for a sequence of three measurements. We weight the arcs with the log of the transition probabilities, and the nodes with the log of the emission probabilities. I have shown some weights*

column to each node in the last. There are S such paths, one for each node in the first column. Once we have these paths, we can choose the one with highest log joint probability. Now consider one of these paths. It passes through the i'th node in the u'th column. The path segment from this node to the end column must, itself, be the best path from this node to the end. If it wasn't, we could improve the original path by substituting the best. This is the key insight that gives us an algorithm.

Start at the *final* column of the tellis. We can evaluate the best path from each node in the final column to the final column, because that path is just the node, and the value of that path is the node weight. Now consider a two-state path, which will start at the second last column of the trellis (look at panel I in Figure 7.8). We can easily obtain the value of the best path leaving each node in this column. Consider a node: we know the weight of each arc leaving the node and the weight of the node at the far end of the arc, so we can choose the path segment with the largest value of the sum; this arc is the best we can do leaving that node. This sum is the best value obtainable on leaving that node—which is often known as the **cost to go function**.

Now, because we know the best value obtainable on leaving each node in the second-last column, we can figure out the best value obtainable on leaving each node in the third-last column (panel II in Figure 7.8). At each node in the third-last column, we have a choice of arcs. Each of these reaches a node *from which we know the value of the best path.* So we can choose the best path leaving a node in

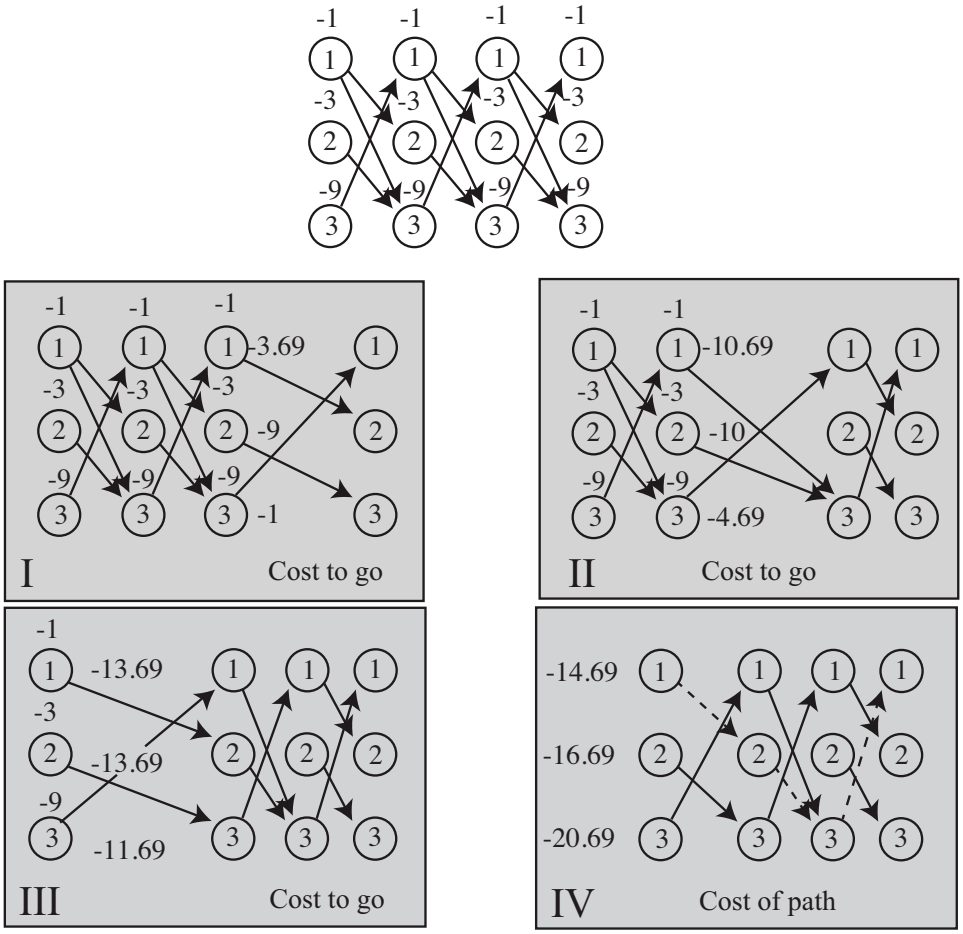

FIGURE 7.8: *An example of finding the best path through a trellis. The probabilities of leaving a node are uniform (and remember,* $\ln 2 \approx -0.69$*). Details in the text.*

the third-last column by finding the path that has the best value of: the arc weight leaving the node; the weight of the node in the second-last column the arc arrives at; and the value of the path leaving that node. This is much more easily done than described. All this works just as well for the fourth-last column, etc. (panel III in Figure 7.8) so we have a recursion. To find the value of the best path with $X_1 = i$, we go to the corresponding node in the first column, then add the value of the node to the value of the best path leaving that node (panel IV in Figure 7.8). Finally, to find the value of the best path leaving the first column, we compute the maximum value over all nodes in the first column.

We can also get the path with the maximum likelihood value. When we compute the value of a node, we erase all but the best arc leaving that node. Once we reach the first column, we simply follow the path from the node with the best value. This path is illustrated by dashed edges in Figure 7.8.

7.4.3 Dynamic Programming for HMM's: Formalities

We will formalize the recursion of the previous section with two ideas. First, we define $C_w(j)$ to be the cost of the best path segment to the end of the trellis *leaving* the node representing $X_w = j$. Second, we define $B_w(j)$ to be the node in column $w+1$ that lies on the best path *leaving* the node representing $X_w = j$. So $C_w(j)$ tells you the cost of the best path, and $B_w(j)$ tells you what node is next on the best path.

Now it is straightforward to find the cost of the best path leaving each node in the second last column, and also the path. In symbols, we have

$$C_{N-1}(j) = \max_u \left[\log P(X_N = u | X_{N-1} = j) + \log P(Y_N | X_N = u)\right]$$

and

$$B_{N-1}(j) = \operatorname*{argmax}_u \left[\log P(X_N = u | X_{N-1} = j) + \log P(Y_N | X_N = u)\right].$$

You should check this against step I of Figure 7.8

Once we have the best path leaving each node in the $w+1$'th column and its cost, it's straightforward to find the best path leaving the w'th column and its cost. In symbols, we have

$$C_w(j) = \max_u \left[\log P(X_{w+1} = u | X_w = j) + \log P(Y_{w+1} | X_{w+1} = u) + C_{w+1}(u)\right]$$

and

$$B_w(j) = \operatorname*{argmax}_u \left[\log P(X_{w+1} = u | X_w = j) + \log P(Y_{w+1} | X_{w+1} = u) + C_{w+1}(u)\right].$$

Check this against steps II and III in Figure 7.8.

7.4.4 Example: Simple Communication Errors

Hidden Markov models can be used to correct text errors. We will simplify somewhat, and assume we have text that has no punctuation marks, and no capital letters. This means there are a total of 27 symbols (26 lower case letters, and a space). We send this text down some communication channel. This could be a telephone line, a fax line, a file saving procedure or anything else. This channel makes errors independently at each character. For each location, with probability $1-p$ the output character at that location is the same as the input character. With probability p, the channel chooses randomly between the character one ahead or one behind in the character set, and produces that instead. You can think of this as a simple model for a mechanical error in one of those now ancient printers where a character strikes a ribbon to make a mark on the paper. We must reconstruct the transmission from the observations.

I built a unigram model, a bigram model, and a trigram model. I stripped the text of this chapter of punctuation marks and mapped the capital letters to lower case letters. I used an HMM package (in my case, for Matlab; but there's a good one for R as well) to perform inference. The main programming here is housekeeping

*	e	t	i	a	o	s	n	r	h
1.9e-1	9.7e-2	7.9e-2	6.6e-2	6.5e-2	5.8e-2	5.5e-2	5.2e-2	4.8e-2	3.7e-2

TABLE 7.1: *The most common single letters (unigrams) that I counted from a draft of this chapter, with their probabilities. The '*' stands for a space. Spaces are common in this text, because I have tended to use short words (from the probability of the '*', average word length is between five and six letters).*

Lead char					
*	*t (2.7e-2)	*a (1.7e-2)	*i (1.5e-2)	*s (1.4e-2)	*o (1.1e-2)
e	e* (3.8e-2)	er (9.2e-3)	es (8.6e-3)	en (7.7e-3)	el (4.9e-3)
t	th (2.2e-2)	t* (1.6e-2)	ti (9.6e-3)	te (9.3e-3)	to (5.3e-3)
i	in (1.4e-2)	is (9.1e-3)	it (8.7e-3)	io (5.6e-3)	im (3.4e-3)
a	at (1.2e-2)	an (9.0e-3)	ar (7.5e-3)	a* (6.4e-3)	al (5.8e-3)
o	on (9.4e-3)	or (6.7e-3)	of (6.3e-3)	o* (6.1e-3)	ou (4.9e-3)
s	s* (2.6e-2)	st (9.4e-3)	se (5.9e-3)	si (3.8e-3)	su (2.2e-3)
n	n* (1.9e-2)	nd (6.7e-3)	ng (5.0e-3)	ns (3.6e-3)	nt (3.6e-3)
r	re (1.1e-2)	r* (7.4e-3)	ra (5.6e-3)	ro (5.3e-3)	ri (4.3e-3)
h	he (1.4e-2)	ha (7.8e-3)	h* (5.3e-3)	hi (5.1e-3)	ho (2.1e-3)

TABLE 7.2: *The most common bigrams that I counted from a draft of this chapter, with their probabilities. The '*' stands for a space. For each of the 10 most common letters, I have shown the five most common bigrams with that letter in the lead. This gives a broad view of the bigrams, and emphasizes the relationship between unigram and bigram frequencies. Notice that the first letter of a word has a slightly different frequency than letters (top row; bigrams starting with a space are first letters). About* 40% *of the possible bigrams do not appear in the text.*

to make sure the transition and emission models are correct. About 40% of the bigrams and 86% of the trigrams did not appear in the text. I smoothed the bigram and trigram probabilities by dividing the probability 0.01 evenly between all unobserved bigrams (resp. trigrams). The most common unigrams, bigrams and trigrams appear in the tables. As an example sequence, I used

> "the trellis has two crucial properties each directed path through the trellis from the start column to the end column represents a legal sequence of states now for some directed path from the start column to the end column sum all the weights for the nodes and edges along this path this sum is the log of the joint probability of that sequence of states with the measurements you can verify each of these statements easily by reference to a simple example"

(which is text you could find in a draft of this chapter). There are 456 characters in this sequence.

When I ran this through the noise process with $p = 0.0333$, I got

> "theztrellis has two crucial properties each directed path through the tqdllit from the start column to the end coluln represents a legal se-

*th	the	he*	is*	*of	of*	on*	es*	*a*	ion
1.7e-2	1.2e-2	9.8e-3	6.2e-3	5.6e-3	5.4e-3	4.9e-3	4.9e-3	4.9e-3	4.9e-3
tio	e*t	in*	*st	*in	at*	ng*	ing	*to	*an
4.6e-3	4.5e-3	4.2e-3	4.1e-3	4.1e-3	4.0e-3	3.9e-3	3.9e-3	3.8e-3	3.7e-3

TABLE 7.3: *The most frequent 10 trigrams in a draft of this chapter, with their probabilities. Again, '*' stands for space. You can see how common 'the' and '*a*' are; 'he*' is common because '*the*' is common. About* 80% *of possible trigrams do not appear in the text.*

> quencezof states now for some directed path from the start column to thf end column sum aml the veights for the nodes and edges along this path this sum is the log of the joint probability oe that sequence of states wish the measurements youzcan verify each of these statements easily by reference to a simple examqle"

which is mangled but not too badly (13 of the characters are changed, so 443 locations are the same).

The unigram model produces

> "the trellis has two crucial properties each directed path through the tqdllit from the start column to the end coluln represents a legal sequence of states now for some directed path from the start column to thf end column sum aml the veights for the nodes and edges along this path this sum is the log of the joint probability oe that sequence of states wish the measurements you can verify each of these statements easily by reference to a simple examqle"

which fixes three errors. The unigram model only changes an observed character when the probability of encountering that character on its own is less than the probability it was produced by noise. This occurs only for "z", which is unlikely on its own and is more likely to have been a space. The bigram model produces

> "she trellis has two crucial properties each directed path through the trellit from the start column to the end coluln represents a legal sequence of states now for some directed path from the start column to the end column sum aml the veights for the nodes and edges along this path this sum is the log of the joint probability oe that sequence of states wish the measurements you can verify each of these statements easily by reference to a simple example"

This is the same as the correct text in 449 locations, so somewhat better than the noisy text. The trigram model produces

> "the trellis has two crucial properties each directed path through the trellit from the start column to the end column represents a legal sequence of states now for some directed path from the start column to the end column sum all the weights for the nodes and edges along this path this sum is the log of the joint probability of that sequence of states

with the measurements you can verify each of these statements easily by reference to a simple example"

which corrects all but one of the errors (look for "trellit").

7.5 YOU SHOULD

7.5.1 remember these definitions:

7.5.2 remember these terms:

7.5.3 remember these facts:

7.5.4 be able to:

- Estimate various probabilities and expectations for a Markov chain by simulation.
- Evaluate the results of multiple runs of a simple simulation.

- Set up a simple HMM and use it to solve problems.

PROBLEMS

7.1. Multiple die rolls: You roll a fair die until you see a 5, then a 6; after that, you stop. Write $P(N)$ for the probability that you roll the die N times.
- **(a)** What is $P(1)$?
- **(b)** Show that $P(2) = (1/36)$.
- **(c)** Draw a directed graph encoding all the sequences of die rolls that you could encounter. Don't write the events on the edges; instead, write their probabilities. There are 5 ways not to get a 5, but only one probability, so this simplifies the drawing.
- **(d)** Show that $P(3) = (1/36)$.
- **(e)** Now use your directed graph to argue that $P(N) = (5/6)P(N-1) + (25/36)P(N-2)$.

7.2. More complicated multiple coin flips: You flip a fair coin until you see either HTH or THT, and then you stop. We will compute a recurrence relation for $P(N)$.
- **(a)** Draw a directed graph for this chain.
- **(b)** Think of the directed graph as a finite state machine. Write Σ_N for some string of length N accepted by this finite state machine. Use this finite state machine to argue that $Sigma_N$ has one of four forms:

  1. $TT\Sigma_{N-2}$
  2. $HH\Sigma_{N-3}$
  3. $THH\Sigma_{N-2}$
  4. $HTT\Sigma_{N-3}$

- **(c)** Now use this argument to show that $P(N) = (1/2)P(N-2)+(1/4)P(N-3)$.

7.3. For the umbrella example of worked example 7.2, assume that with probability 0.7 it rains in the evening, and 0.2 it rains in the morning. I am conventional, and go to work in the morning, and leave in the evening.
- **(a)** Write out a transition probability matrix.
- **(b)** What is the stationary distribution? (you should use a simple computer program for this).
- **(c)** What fraction of evenings do I arrive at home wet?
- **(d)** What fraction of days do I arrive at my destination dry?

PROGRAMMING EXERCISES

7.4. A dishonest gambler has two dice and a coin. The coin and one die are both fair. The other die is unfair. It has $P(n) = [0.5, 0.1, 0.1, 0.1, 0.1, 0.1]$ (where n is the number displayed on the top of the die). The gambler starts by choosing a die. Choosing a die is by flipping a coin; if the coin comes up heads, the gambler chooses the fair die, otherwise, the unfair die. The gambler rolls the chosen die repeatedly until a 6 comes up. When a 6 appears, the gambler chooses again (by flipping a coin, etc), and continues.
- **(a)** Model this process with a hidden markov model. The emitted symbols should be $1, \ldots, 6$. Doing so requires only two hidden states (which die is

in hand). Simulate a long sequence of rolls using this model. What is the probability the emitted symbol is 1?

(b) Use your simulation to produce 10 sequences of 100 symbols. Record the hidden state sequence for each of these. Now recover the hidden state using dynamic programming (you should likely use a software package for this; there are many good ones for R and Matlab). What fraction of the hidden states is correctly identified by your inference procedure?

(c) Does inference accuracy improve when you use sequences of 1000 symbols?

7.5. Warning: this exercise is fairly elaborate, though straightforward. We will correct text errors using a hidden Markov model.

(a) Obtain the text of a copyright-free book in plain characters. One natural source is Project Gutenberg, at https://www.gutenberg.org. Simplify this text by dropping all punctuation marks except spaces, mapping capital letters to lower case, and mapping groups of many spaces to a single space. The result will have 27 symbols (26 lower case letters and a space). From this text, count unigram, bigram and trigram letter frequencies.

(b) Use your counts to build models of unigram, bigram and trigram letter probabilities. You should build both an unsmoothed model, and at least one smoothed model. For the smoothed models, choose some small amount of probability ϵ and split this between all events with zero count. Your models should differ only by the size of ϵ.

(c) Construct a corrupted version of the text by passing it through a process that, with probability p_c, replaces a character with a randomly chosen character, and otherwise reports the original character.

(d) For a reasonably sized block of corrupted text, use an HMM inference package to recover the best estimate of your true text. Be aware that your inference will run more slowly as the block gets bigger, but you won't see anything interesting if the block is (say) too small to contain any errors.

(e) For $p_c = 0.01$ and $p_c = 0.1$, estimate the error rate for the corrected text for different values of ϵ. Keep in mind that the corrected text could be worse than the corrupted text.

PART THREE

INFERENCE

CHAPTER 8

Inference: Making Point Estimates

Inference is the process of drawing conclusions from data. One form of inference is to estimate a number, or set of numbers, that describes a dataset. The result is known as a **point estimate**. An alternative is to estimate an interval within which a number lies, with some degree of certainty. Such estimates are known as **interval estimates**. Finally, one might wish to assess the extent to which a body of evidence supports rejecting an hypothesis — known as **hypothesis testing**. In this chapter, we deal with point estimates. In the following chapter, we deal with interval estimates and hypothesis testing, which require an understanding of point estimates. There are two, somewhat distinct, situations in which we could make point estimates.

In the first, we have a dataset $\{\mathbf{x}\}$, and a probability model we believe applies to that dataset. But we need to select appropriate values of the parameters to ensure that the model describes the data. For example, we might have a set of N coin flips which we believe to be independent and identically distributed. Of these, k flips came up H. We know that a binomial distribution with is a good model. But, to date, we've known $p(H)$ (which I'll call θ) when we've built this model. What value of θ is suggested by our observations? Your intuition is likely to suggest using k/N, but we'd like a better procedure than guessing. We need an inference procedure to obtain the unknown parameter or parameters from the data. Notice that we will get an estimate, rather than the "true" value. As we shall see, there is more than one possible procedure to apply, depending to some extent on the problem. In some cases (section 8.1), we estimate parameter values based solely on data; in others (section 8.2), we are able to use prior information about the parameters to affect the estimate.

In the second situation, we want to know some property of a population. For example, we may wish to know the mean weight of a person, or the mean response of a mouse to a drug. It would be a stretch to describe this population with one of the probability models that we have seen. In principle, the number we want is not even necessarily random; in principle, we could measure everyone on the planet and average the weights. In practice, this doesn't make sense, for quite straightforward reasons. You can't really weigh every person, or dose every mouse, on the planet. Instead, to estimate this property, we obtain a sample (some people; some mice; etc.) of the population, and estimate the property from the sample. There is now an important problem. Different samples lead to different estimates of the property. We will arrange to have the sample drawn randomly from the population, so the sample we see represents the value of a set of random variables. If you followed the proof of the weak law of large numbers, you should suspect that the mean of this sample could be a good estimate of the population mean. This turns out to be the case (section 8.3). However, there is some random error in the estimate, and we

can tell (on average) how large the error caused by random sampling could be.

8.1 ESTIMATING MODEL PARAMETERS WITH MAXIMUM LIKELIHOOD

Assume we have a dataset $\mathcal{D} = \{\mathbf{x}\}$, and a probability model we believe applies to that dataset. Generally, application logic suggests the type of model (i.e. normal probability density; Poisson probability; geometric probability; and so on). But usually, we do not know the parameters of the model — for example, the mean and standard deviation of a normal distribution; the intensity of a poisson distribution; and so on. Notice that this situation is unlike what we have seen to date. In chapter 6, we assumed that we knew parameters, and could then use the model to assign a probability to a set of data items $\mathcal{D}$. Here we *know* the value of $\mathcal{D}$, but don't know the parameters. Our model will be better or worse depending on how well we choose the parameters. We need a strategy to estimate the parameters of a model from a sample dataset. Notice how each of the following examples fits this pattern. There is an important, and widespread, convention that I shall adhere to. Unknown parameters are widely referred to as θ (which could be a scalar, or a vector, or, if you're lucky, much more interesting than that).

Example: 8.1 *Inferring p from repeated flips — binomial*

We could flip the coin N times, and count the number of heads h. We know that an appropriate probability model for a set of independent coin flips is the binomial model $P_b(h; N, \theta)$, where θ is the probability a flipped coin comes up heads (which was written $p(H)$ or p in the binomial model). But we do not know $p(H)$, which is the parameter. I wrote this as θ because we do not know it. We need a strategy to extract a value of θ from the data.

Example: 8.2 *Inferring p from repeated flips — geometric*

We could flip the coin repeatedly until we see a head. We know that, in this case, the number of flips has the geometric distribution with parameter $p(H)$. In this case, the data is a sequence of T's with a final H from the coin flips. There are N flips (or terms) and the last flip is a head. We know that an appropriate probability model is the geometric distribution $P_g(N; \theta)$. But we do not know $p(H)$, which is the parameter I wrote as θ.

Example: 8.3 *Inferring the intensity of spam — poisson*

It is reasonable to assume that the number of spam emails one gets in an hour has a Poisson distribution. But what is the intensity parameter, written λ in the definition, of that distribution? We could count the number of spam emails that arrive in each of a set of distinct hours, giving a dataset of counts $\mathcal{D}$. We need a strategy to wrestle an estimate of the intensity parameter, which I shall write θ because we don't know it, from this dataset.

Example: 8.4 *Inferring the mean and standard deviation of normal data*

Imagine we know for some reason that our data is well described by a normal distribution. We could ask what is the mean and standard deviation of the normal distribution that best represents the data?

8.1.1 The Maximum Likelihood Principle

We need a "reasonable" procedure to choose a value of θ to report. The **maximum likelihood principle** asks for a choice of θ that makes the data observed "most probable". It is important to understand that there are other ways of proceeding (we'll see one in section 8.2). With that said, the maximum likelihood principle has served extremely well since the start of the 20'th century.

Write θ for the parameters of our model. If we knew θ, then the probability of observing the data $\mathcal{D}$ would be $P(\mathcal{D}|\theta)$. We can construct an expression for $P(\mathcal{D}|\theta)$ using our model. Now we know $\mathcal{D}$, and we don't know θ, so the value of $P(\mathcal{D}|\theta)$ is a function of θ.

Definition: 8.1 *Likelihood*

The function $P(\mathcal{D}|\theta)$, which is a function of θ, is known as the **likelihood** of the data D, and is often written $\mathcal{L}(\theta)$ (or $\mathcal{L}(\theta; \mathcal{D})$ if you want to remember that data is involved).

The maximum likelihood principle asks for a choice of θ such that the probability of observing the data you actually see, is maximised. This should strike you as being a reasonable choice.

Definition: 8.2 *The maximum likelihood principle*

Choose θ such that $\mathcal{L}(\theta) = P(\mathcal{D}|\theta)$ is maximised, as a function of θ.

For the examples we work with, the data will be **independent and identically distributed** or **IID**. This means that each data item is an idependently obtained sample from the same probability distribution (see section **??**). In turn, this means that the likelihood is a product of terms, one for each data item, which we can write as

$$\mathcal{L}(\theta) = P(\mathcal{D}|\theta) = \prod_{i \in \text{dataset}} P(d_i|\theta).$$

There are two, distinct, important concepts we must work with. One is the unknown parameter(s), which we will write θ. The other is the *estimate* of the value of that parameter, which we will write $\hat{\theta}$. This estimate is the best we can do — it may not be the "true" value of the parameter.

Worked example 8.1 *Inferring $p(H)$ for a coin from flips using a binomial model*

In N independent coin flips, you observe k heads. Use the maximum likelihood principle to infer $p(H)$.

Solution: The coin has $\theta = p(H)$, which is the unknown parameter. We know that an appropriate probability model is the binomial model $P_b(k; N, \theta)$. We have that

$$\mathcal{L}(\theta) = P(\mathcal{D}|\theta) = P_b(k; N, \theta) = \left(\begin{array}{c} N \\ k \end{array} \right) \theta^k (1-\theta)^{(N-k)}$$

which is a function of θ — the unknown probability that a coin comes up heads; k and N are known. We must find the value of θ that maximizes this expression. Now the maximum occurs when

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = 0.$$

We have

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = \left(\begin{array}{c} N \\ k \end{array} \right) \left(k\theta^{k-1}(1-\theta)^{(N-k)} - \theta^k (N-k)(1-\theta)^{(n-k-1)} \right)$$

and this is zero when

$$k\theta^{k-1}(1-\theta)^{(N-k)} = \theta^k (N-k)(1-\theta)^{(N-k-1)}$$

so the maximum occurs when

$$k(1-\theta) = \theta(N-k).$$

This means the maximum likelihood estimate is

$$\hat{\theta} = \frac{k}{N}.$$

This answer seems natural to most people, and it's very likely that you would make this guess without knowing the maximum likelihood principle. But now we have a procedure we can apply to other problems.

Worked example 8.2 *Inferring $p(H)$ from coin flips using a geometric model*

You flip a coin N times, stopping when you see a head. Use the maximum likelihood principle to infer $p(H)$ for the coin.

Solution: The coin has $\theta = p(H)$, which is the unknown parameter. We know that an appropriate probability model is the geometric model $P_g(N;\theta)$. We have that

$$\mathcal{L}(\theta) = P(\mathcal{D}|\theta) = P_g(N;\theta) = (1-\theta)^{(N-1)}\theta$$

which is a function of θ — the unknown probability that a coin comes up heads; N is known. We must find the value of θ that maximizes this expression. Now the maximum occurs when

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = 0 = ((1-\theta)^{(N-1)} - (N-1)(1-\theta)^{(N-2)}\theta)$$

So the maximum likelihood estimate is

$$\hat{\theta} = \frac{1}{N}.$$

Most people don't guess this estimate, though it usually seems reasonable in retrospect.

Worked example 8.3 *Inferring die probabilities from multiple rolls and a multinomial distribution*

You throw a die N times, and see n_1 ones, ... and n_6 sixes. Write $p_1, \ldots, p_6$ for the probabilities that the die comes up one, ..., six. Use the maximum likelihood principle to estimate $p_1, \ldots, p_6$.

Solution: The data are n, $n_1, \ldots, n_6$. The parameters are $\theta = (p_1, \ldots, p_6)$. $P(\mathcal{D}|\theta)$ comes from the multinomial distribution. In particular,

$$\mathcal{L}(\theta) = P(\mathcal{D}|\theta) = \frac{n!}{n_1! \ldots n_6!} p_1^{n_1} p_2^{n_2} \ldots p_6^{n_6}$$

which is a function of $\theta = (p_1, \ldots, p_6)$. Now we want to maximize this function by choice of θ. Notice that we could do this by simply making all p_i very large — but this omits a fact, which is that $p_1 + p_2 + p_3 + p_4 + p_5 + p_6 = 1$. So we substitute using $p_6 = 1 - p_1 - p_2 - p_3 - p_4 - p_5$ (there are other, neater, ways of dealing with this issue, but they take more background knowledge). At the maximum, we must have that for all i,

$$\frac{\partial \mathcal{L}(\theta)}{\partial p_i} = 0$$

which means that, for p_i, we must have

$$n_i p_i^{(n_i - 1)} (1 - p_1 - p_2 - p_3 - p_4 - p_5)^{n_6} - p_i^{n_i} n_6 (1 - p_1 - p_2 - p_3 - p_4 - p_5)^{(n_6 - 1)} = 0$$

so that, for each p_i, we have

$$n_i(1 - p_1 - p_2 - p_3 - p_4 - p_5) - n_6 p_i = 0$$

or

$$\frac{p_i}{1 - p_1 - p_2 - p_3 - p_4 - p_5} = \frac{n_i}{n_6}.$$

You can check that this equation is solved by

$$\hat{\theta} = \frac{1}{(n_1 + n_2 + n_3 + n_4 + n_5 + n_6)} (n_1, n_2, n_3, n_4, n_5, n_6)$$

There is one problem here, which is we need to take derivatives of products, which can quickly lead to quite unmanageable expressions. There is a cure. The logarithm is a monotonic function (i.e. if $x > 0$, $y > 0$, $x > y$, then $\log(x) > \log(y)$). This means that the values of θ that maximise the log-likelihood are the same as the values that maximise the likelihood. This observation allows us to transform a

product into a sum, and the derivative of a sum is easy. We have

$$\log P(\mathcal{D}|\theta) = \log \prod_{i \in \text{dataset}} P(d_i|\theta) = \sum_{i \in \text{dataset}} \log P(d_i|\theta)$$

and in some cases, $\log P(d_i|\theta)$ takes a convenient, easy form.

Definition: 8.3 *The log-likelihood of a dataset under a model*

is a function of the unknown parameters, and you will often see it written as

$$\log \mathcal{L}(\theta) = \log P(\mathcal{D}|\theta) = \sum_{i \in \text{dataset}} \log P(d_i|\theta).$$

Worked example 8.4 *Poisson distributions*

You observe N intervals, each of the same, fixed length (in time, or space). You know that, in these intervals, events occur with a Poisson distribution (for example, you might be observing Prussian officers being kicked by horses, or telemarketer calls...). You know also that the intensity of the Poisson distribution is the same for each observation. The number of events you observe in the i'th interval is n_i. What is the intensity of the Poisson distribution?

Solution: Write θ for the unknown intensity. The likelihood is

$$\mathcal{L}(\theta) = \prod_{i \in \text{intervals}} P(\{n_i \text{ events}\} | \theta) = \prod_{i \in \text{intervals}} \frac{\theta^{n_i} e^{-\theta}}{n_i!}.$$

It will be easier to work with logs. The log-likelihood is

$$\log \mathcal{L}(\theta) = \sum_i (n_i \log \theta - \theta - \log n_i!)$$

so that we must solve

$$\frac{\partial \log \mathcal{L}(\theta)}{\partial \theta} = \sum_i (\frac{n_i}{\theta} - 1) = 0$$

which yields a maximum likelihood estimate of

$$\hat{\theta} = \frac{\sum_i n_i}{N}$$

Worked example 8.5 *The intensity of swearing*

A famously sweary politician gives a talk. You listen to the talk, and for each of 30 intervals 1 minute long, you record the number of swearwords. You record this as a histogram (i.e. you count the number of intervals with zero swear words, with one, etc.). For the first 10 intervals, you see

no. of swear words	no. of intervals
0	5
1	2
2	2
3	1
4	0

and for the following 20 intervals, you see

no. of swear words	no. of intervals
0	9
1	5
2	3
3	2
4	1

Assume that the politician's use of swearwords is Poisson. What is the intensity using the first 10 intervals? the second 20 intervals? all the intervals? why are they different?

Solution: Use the expression from worked example 8.4 to find

$$\begin{aligned}\hat{\theta}_{10} &= \frac{\text{total number of swearwords}}{\text{number of intervals}} \\ &= \frac{9}{10}\end{aligned}$$

$$\begin{aligned}\hat{\theta}_{20} &= \frac{\text{total number of swearwords}}{\text{number of intervals}} \\ &= \frac{21}{20}\end{aligned}$$

$$\begin{aligned}\hat{\theta}_{30} &= \frac{\text{total number of swearwords}}{\text{number of intervals}} \\ &= \frac{30}{30}.\end{aligned}$$

These are different because the maximum likelihood estimate is an *estimate* — we can't expect to recover the exact value from a dataset. Notice, however, that the estimates are quite close.

Worked example 8.6 *Normal distributions*

Assume we have $x_1, \ldots, x_N$, and we wish to model these data with a normal distribution. Use the maximum likelihood principle to estimate the mean of that normal distribution.

Solution: The likelihood of a set of data values under the normal distribution with unknown mean θ and standard deviation σ is

$$\begin{aligned}\mathcal{L}(\theta) &= P(x_1, \ldots x_N|\theta, \sigma) \\ &= P(x_1|\theta, \sigma)P(x_2|\theta, \sigma)\ldots P(x_N|\theta, \sigma) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x_i-\theta)^2}{2\sigma^2}\right)\end{aligned}$$

and this expression is a moderate nuisance to work with. The log of the likelihood is

$$\log \mathcal{L}(\theta) = \left(\sum_{i=1}^{N} -\frac{(x_i-\theta)^2}{2\sigma^2}\right) + \quad \text{term not depending on } \theta.$$

We can find the maximum by differentiating wrt θ and setting to zero, which yields

$$\begin{aligned}\frac{\partial \log \mathcal{L}(\theta)}{\partial \theta} &= \sum_{i=1}^{N} \frac{2(x_i-\theta)}{2\sigma^2} \\ &= 0 \\ &= \frac{1}{\sigma^2}\left(\sum_{i=1}^{N} x_i - N\theta\right)\end{aligned}$$

so the maximum likelihood estimate is

$$\hat{\theta} = \frac{\sum_{i=1}^{N} x_i}{N}$$

which probably isn't all that surprising. Notice we did not have to pay attention to σ in this derivation — we did not assume it was known, it just doesn't do anything.

Worked example 8.7 *Normal distributions -II*

Assume we have $x_1, \ldots, x_N$ which are data that can be modelled with a normal distribution. Use the maximum likelihood principle to estimate the standard deviation of that normal distribution.

Solution: Now we have to write out the log of the likelihood in more detail. Write μ for the mean of the normal distribution and θ for the unknown standard deviation of the normal distribution. We get

$$\log \mathcal{L}(\theta) = \left(\sum_{i=1}^{N} -\frac{(x_i - \mu)^2}{2\theta^2} \right) - N \log \theta + \text{Term not depending on } \theta$$

We can find the maximum by differentiating wrt θ and setting to zero, which yields

$$\frac{\partial \log \mathcal{L}(\theta)}{\partial \theta} = \frac{-2}{\theta^3} \sum_{i=1}^{N} \frac{-(x_i - \mu)^2}{2} - \frac{N}{\theta} = 0$$

so the maximum likelihood estimate is

$$\hat{\theta} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N}}$$

which probably isn't all that surprising, either.

Useful Fact: 8.1 *Maximum likelihood is the go-to inference method for lots of data*

If you have many data items and a model, you really should use maximum likelihood.

You should notice that one could maximize the likelihood of a normal distribution with respect to mean and standard deviation in one go (i.e. I could have done worked examples 8.6 and 8.7 in one worked example, instead of two). I did this example in two parts because I felt it was more accessible that way; if you object, you're likely to be able to fill in the details yourself very easily.

The maximum likelihood principle has a variety of neat properties we cannot expound. One worth knowing about is **consistency**; for our purposes, this means that the maximum likelihood estimate of parameters can be made arbitrarily close to the right answer by having a sufficiently large dataset. Now assume that our data doesn't actually come from the underlying model. This is the usual case, because we can't usually be sure that, say, the data truly is normal or truly comes from a

Poisson distribution. Instead we choose a model that we think will be useful. When the data doesn't come from the model, maximum likelihood produces an estimate of θ that corresponds to the model that is (in quite a strong sense, which we can't explore here) the closest to the source of the data. Maximum likelihood is very widely used because of these neat properties. But there are some difficulties.

8.1.2 Cautions about Maximum Likelihood

One important problem is that it might be hard to find the maximum of the likelihood exactly. There are strong numerical methods for maximizing functions, and these are very helpful, but even today there are likelihood functions where it is very hard to find the maximum.

The second is that small amounts of data can present nasty problems. For example, in the binomial case, if we have only one flip we will estimate p as either 1 or 0. We should find this report unconvincing. In the geometric case, with a fair coin, there is a probability 0.5 that we will perform the estimate and then report that the coin has $p = 1$. This should also worry you. As another example, if we throw a die only a few times, we could reasonably expect that, for some i, $n_i = 0$. This doesn't necessarily mean that $p_i = 0$, though that's what the maximum likelihood inference procedure will tell us.

This creates a very important technical problem — how can I estimate the probability of events that haven't occurred? This might seem like a slightly silly question to you, but it isn't. Solving this problem has really significant practical consequences. As one example, consider a biologist trying to count the number of butterfly species on an island. The biologist catches and classifies a lot of butterflies, then leaves. But are there more butterfly species on the island? To get some sense that we can reason successfully about this problem, compare two cases. In the first, the biologist catches many individuals of each of the species observed. In this case, you should suspect that catching more butterflies is unlikely to yield more species. In the second case, there are many species where the biologist sees only one individual of that species. In this case, you should suspect that catching more butterflies might very well yield new species.

8.2 INCORPORATING PRIORS WITH BAYESIAN INFERENCE

Another important issue with maximum likelihood is that there is no mechanism to incorporate prior beliefs. For example, imagine you get a new die from a reliable store, roll it six times and see a one once. You would be happy to believe that $p(6) = 1/6$ for this die. Now imagine you borrow a die from a friend with a long history of making weighted dice. Your friend tells you this die is weighted so that $p(1) = 1/2$. You roll the die six times and see a one once; in this case, you might worry that $p(6)$ isn't $1/6$, and you just happened to get a slightly unusual set of rolls. You'd worry because you have good reason to believe the die isn't fair, and you'd want more evidence to believe $p(6) = 1/6$. Maximum likelihood can't distinguish between these two cases.

The difference lies in prior information — information we possess before we look at the data. We would like to take this information into account when we estimate the model. One way to do so is to place a **prior probability distribution**

$p(\theta)$ on the parameters θ. Then, rather than working with the likelihood $p(\mathcal{D}|\theta)$, we could apply Bayes' rule, and form the **posterior** $p(\theta|\mathcal{D})$. This posterior represents the probability that θ takes various values, given the data $\mathcal{D}$.

Definition: 8.4 *Bayesian inference*

Extracting information from the posterior $p(\theta|\mathcal{D})$ is usually called **Bayesian inference**.

Bayes' rule tells us that

$$p(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})}.$$

You should notice that having this probability distribution immediately allows us to answer quite sophisticated questions (some more detail in Chapter 9). For example, we could immediately compute $P(\{\theta \in [0.2, 0.4]\}|\mathcal{D})$ in a straightforward way (though the reason to care is put off till chapter 9). Quite often, we just want to extract an estimate of θ. For this, we can use the θ that maximizes the posterior.

Definition: 8.5 *MAP estimate*

A natural estimate of θ is the value that maximizes the posterior $p(\theta|\mathcal{D})$. This estimate is known as a **maximum a posteriori estimate** or **MAP estimate**.

To get this θ, we do not need to know the *value* of the posterior, and it is enough to work with

$$p(\theta|\mathcal{D}) \propto \P(\mathcal{D}|\theta)P(\theta).$$

This form exposes similarities and differences between Bayesian and maximum likelihood inference. To reason about the posterior, you need to have a prior $P(\theta)$. If you assume that this prior has the same value for every θ, you'll end up doing maximum likelihood, because $P(\theta)$ is some constant. Using a prior that isn't constant means that the MAP estimate may be different from the maximum likelihood estimate. This difference is occuring because you have prior beliefs that some θ are more likely to occur than others. Supplying these prior beliefs is part of the modelling process.

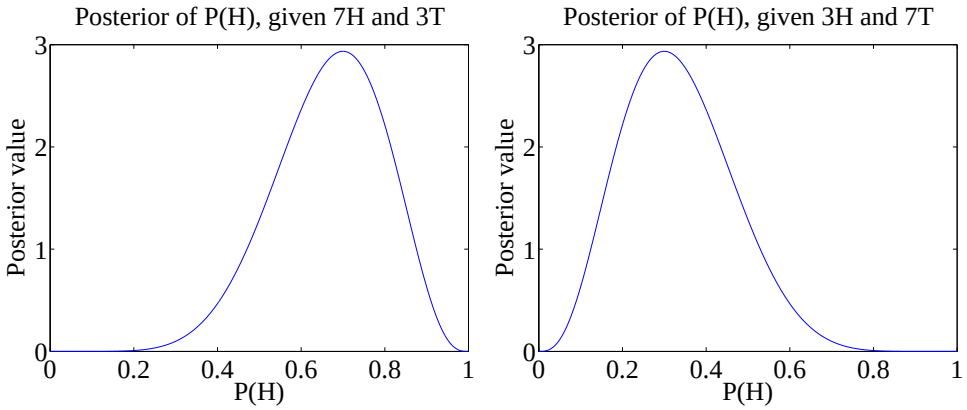

FIGURE 8.1: *The curves show a function* proportional to *the posterior on θ, for the two cases of example 8.8. Notice that this information is rather richer than the single value we would get from maximum likelihood inference.*

Worked example 8.8 *Flipping a coin*

We have a coin with probability θ of coming up heads when flipped. We start knowing nothing about θ. We then flip the coin 10 times, and see 7 heads (and 3 tails). Plot a function proportional to $p(\theta|\{7 \text{ heads and } 3 \text{ tails}\})$. What happens if there are 3 heads and 7 tails?

Solution: We know nothing about p, except that $0 \leq \theta \leq 1$, so we choose a uniform prior on p. We have that $p(\{7 \text{ heads and } 3 \text{ tails}\}|\theta)$ is binomial. The joint distribution is $p(\{7 \text{ heads and } 3 \text{ tails}\}|\theta) \times p(\theta)$ but $p(\theta)$ is uniform, so doesn't depend on θ. So the posterior is *proportional* to: $\theta^7(1-\theta)^3$ which is graphed in figure 8.1. The figure also shows $\theta^3(1-\theta)^7$ which is *proportional* to the posterior for 3 heads and 7 tails. In each case, the evidence does not rule out the possibility that $\theta = 0.5$, but tends to discourage the conclusion. Maximum likelihood would give $\theta = 0.7$ or $\theta = 0.3$, respectively.

8.2.1 Conjugate priors

Figure 8.1 shows curves *proportional to* the posterior. The functions are not the true posterior, because they do not integrate to one. The fact that we are missing this constant of proportionality means that, for example, we cannot compute $P(\{\theta \in [0.3, 0.6]\}|\mathcal{D})$. For curves like these, it is fairly easy to estimate the constant of proportionality (using either numerical integration or the fact that they're proportional to a binomial distribution). Figure 8.2 shows a set of true posteriors for different sets of evidence.

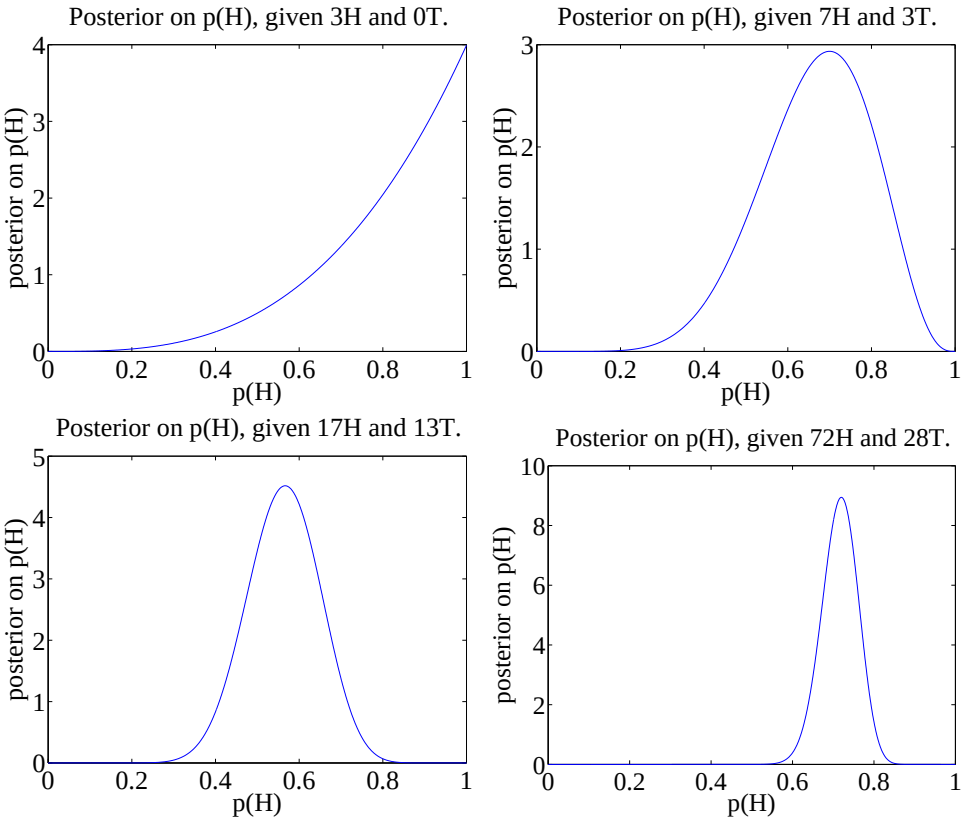

FIGURE 8.2: *The probability that an unknown coin will come up heads when flipped is $p(H)$. For these figures, I simulated coin flips from a coin with $p = 0.75$. I then plotted the posterior for various data. Notice how, as we see more flips, we get more confident about p. The vertical axis changes substantially between plots in this figure.*

The constant of proportionality can be computed by noticing that

$$P(\mathcal{D}) = \int_{\theta} P(\mathcal{D}|\theta)P(\theta)d\theta.$$

It is usually impossible to do this integral in closed form, so we would have to use a numerical integral or a trick. There is one really useful trick: in some cases, $P(\theta)$ and $P(\mathcal{D}|\theta)$, when multiplied together, take a familiar form. This happens when $P(\mathcal{D}|\theta)$ and $P(\theta)$ each belong to parametric families where there is a special relationship between the families. When a prior has this property, it is called a **conjugate** prior. There are some cases worth knowing, given in the worked examples.

Worked example 8.9 *Flipping a coin - II*

We have a coin with probability θ of coming up heads when flipped. We model the prior on θ with a Beta distribution, with parameters $\alpha > 0$, $\beta > 0$. We then flip the coin N times, and see h heads. What is $P(\theta|N, h, \alpha, \beta)$?

Solution: We have that $P(N, h|\theta)$ is binomial, and that $P(\theta|N, h, \alpha, \beta) \propto P(N, h|\theta)P(\theta|\alpha, \beta)$. This means that

$$P(\theta|N, h, \alpha, \beta) \propto \left(\begin{array}{c} N \\ h \end{array} \right) \theta^h (1-\theta)^{(N-h)} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{(\alpha-1)} (1-\theta)^{(\beta-1)}.$$

and we can write

$$P(\theta|N, h, \alpha, \beta) \propto \theta^{(\alpha+h-1)} (1-\theta)^{(\beta+N-h-1)}.$$

Notice this has the form of a Beta distribution, so it is easy to recover the constant of proportionality. We have

$$P(\theta|N, h, \alpha, \beta) = \frac{\Gamma(\alpha+\beta+N)}{\Gamma(\alpha+h)\Gamma(\beta+N-h)} \theta^{(\alpha+h-1)} (1-\theta)^{(\beta+N-h-1)}.$$

Worked example 8.10 *More sweary politicians*

Example 8.5 gives some data from a sweary politician. Assume we have only the first 10 intervals of observations, and we wish to estimate the intensity using a Poisson model. Write θ for this parameter. Use a Gamma distribution as a prior, and write out the posterior.

Solution: We have that

$$\begin{aligned} p(\mathcal{D}|\theta) &= \left(\frac{\theta^0 e^{-\theta}}{0!}\right)^5 \left(\frac{\theta^1 e^{-\theta}}{1!}\right)^2 \times \\ &\quad \left(\frac{\theta^2 e^{-\theta}}{2!}\right)^2 \left(\frac{\theta^3 e^{-\theta}}{3!}\right)^1 \\ &= \frac{\theta^9 e^{-10\theta}}{12} \end{aligned}$$

and

$$p(\theta|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\theta^{(\alpha-1)}e^{-\beta\theta}$$

This means that

$$p(\theta|\mathcal{D}) \propto \theta^{(\alpha-1+9)}e^{-(\beta+10)\theta}.$$

Notice this has the form of another Gamma distribution, so we can write

$$p(\theta|\mathcal{D}) = \frac{(\beta+10)^{(\alpha+9)}}{\Gamma(\alpha+9)}\theta^{(\alpha-1+9)}e^{-(\beta+10)\theta}$$

8.2.2 Normal Prior and Normal Likelihood Yield Normal Posterior

When both $P(\mathcal{D}|\theta)$ and $P(\theta)$ are normal the posterior is normal, too. The mean and standard deviation of the posterior take a simple form. We start with an example. Assume we drop a measuring device down a borehole. It is designed to stop falling and catch onto the side of the hole after it has fallen μ_π meters. On board is a device to measure its depth. This device reports a known constant times the correct depth plus a zero mean normal random variable, which we call "noise". The device reports depth every second.

The first question to ask is what depth do we believe the device is at *before* we receive any measurement? We designed the device to stop at μ_π meters, so we are not completely ignorant about where it is. However, it may not have worked absolutely correctly. We choose to model the depth at which it stops as μ_π meters plus a zero mean normal random variable. The second term could be caused by error in the braking system, etc. We could estimate the standard deviation of the second term (which we write σ_π) either by dropping devices down holes, then measuring with tape measures, or by analysis of likely errors in our braking system. The depth

of the object is the unknown parameter of the model; we write this depth θ. Now the model says that θ is a normal random variable with mean μ_π and standard deviation σ_π.

Notice that this model probably isn't exactly right — for example, there must be some probability in the model that the object falls beyond the bottom of the hole, which it can't do — but it captures some important properties of our system. The device should stop at or close to μ_π meters most of the time, and it's unlikely to be too far away.

Now assume we receive a single measurement — what do we now know about the device's depth? The first thing to notice is that there is something to do here. Ignoring the prior and taking the measurement might not be wise. For example, imagine that the noise in the wireless system is large, so that the measurement is often corrupted — our original guess about the device's location might be better than the measurement. Write x for the measurement. Notice that the scale of the measurement may not be the same as the scale of the depth, so the mean of the measurement is $c\theta$, where c is a change of scale (for example, from inches to meters). We have that $p(x|\theta)$ is normal with mean $c\theta$ and standard deviation σ_m. We would like to know $p(\theta|x)$.

We have that

$$\begin{aligned}
\log p(\theta|x) &= \log p(\theta, x) + \text{terms not depending on } \theta \\
&= \log p(x|\theta) + \log p(\theta) + \text{terms not depending on } \theta \\
&= -\frac{(x - c\theta)^2}{2\sigma_m^2} - \frac{(\theta - \mu_\pi)^2}{2\sigma_\pi^2} \\
&+ \text{terms not depending on } \theta \text{ or } x.
\end{aligned}$$

Now some trickery will get us an expression for $p(\theta|x)$. Notice first that $\log p(\theta|x)$ is of degree 2 in θ (i.e. it has terms θ^2, θ and things that don't depend on θ). This means that $p(\theta|x_1)$ must be a normal distribution, because we can rearrange its log into the form of the log of a normal distribution. This yields a fact of crucial importance.

Useful Fact: 8.2 *Normal distributions are conjugate*

A normal prior and a normal likelihood yield a normal posterior.

Write μ_p for the mean of this distribution, and σ_p for its standard deviation. The log of the distribution must be

$$-\frac{(\theta - \mu_p)^2}{2\sigma_p^2} + \text{terms not depending on } \theta.$$

The terms not depending on θ are not interesting, because if we know σ_p those terms must add up to

$$\log\left(\frac{1}{\sqrt{2\pi}\sigma_p}\right)$$

so that the probability density function sums to one. Notice that

$$-\frac{(\theta-\mu_p)^2}{2\sigma_p^2} = -\theta^2\left(\frac{1}{2\sigma_p^2}\right) + 2\theta\frac{\mu_p}{2\sigma_p^2} + \text{term not depending on } \theta.$$

Now thrash through algebra to get

$$\sigma_p^2 = \frac{\sigma_m^2\sigma_\pi^2}{\sigma_m^2 + c^2\sigma_\pi^2}$$

and

$$\mu_p = \frac{cx\sigma_\pi^2 + \mu_\pi\sigma_m^2}{\sigma_m^2 + c^2\sigma_\pi^2}.$$

These equations "make sense". Imagine that σ_π is very small, and σ_m is very big; then our new expected value of θ — which is μ_1 — is about μ_π. Equivalently, because our prior was very accurate, and the measurement was unreliable, our expected value is about the prior value. Similarly, if the measurement is reliable (i.e. σ_m is small) and the prior has high variance (i.e. σ_π is large), then our expected value of θ is about x/c — i.e. the measurement, rescaled. I have put these equations, in a more general form, in a box below.

Useful Fact: 8.3 *Simple expressions for the parameters of a normal posterior*

Assume we wish to estimate a parameter θ. The prior distribution for θ is normal, with known mean μ_π and known standard deviation σ_π. We receive a single data item x. The likelihood of this data item is normal with mean $c\theta$ and standard deviation σ_m, where c and σ_m are known. Then the posterior, $p(\theta|x, c, \sigma_m, \mu_\pi, \sigma_\pi)$, is normal, with mean

$$\frac{cx\sigma_\pi^2 + \mu_\pi\sigma_m^2}{\sigma_m^2 + c^2\sigma_\pi^2}$$

and standard deviation

$$\sqrt{\frac{\sigma_m^2\sigma_\pi^2}{\sigma_m^2 + c^2\sigma_\pi^2}}.$$

8.2.3 MAP Inference

Look at example 8.1, where we estimated the probability a coin would come up heads with maximum likelihood. We could not change our estimate just by knowing the coin was fair, but we could come up with a number for $\theta = p(H)$ (rather than, say, a posterior distribution). A natural way to produce a point estimate for θ that

incorporates prior information is to choose $\hat{\theta}$ such that

$$\hat{\theta} = \underset{\theta}{\text{argmax}}\ P(\theta|\mathcal{D}) = \underset{\theta}{\text{argmax}}\ \frac{P(\theta, \mathcal{D})}{P(\mathcal{D})}$$

This is the MAP estimate. If we wish to perform MAP inference, $P(\mathcal{D})$ doesn't matter (it changes the value, but not the location, of the maximum). This means we can work with $P(\theta, \mathcal{D})$, often called the **joint** distribution.

Worked example 8.11 *Flipping a coin - II*

We have a coin with probability θ of coming up heads when flipped. We model the prior on θ with a Beta distribution, with parameters $\alpha > 0$, $\beta > 0$. We then flip the coin N times, and see h heads. What is the MAP estimate of θ?

Solution: We have that

$$P(\theta|N, h, \alpha, \beta) = \frac{\Gamma(\alpha + \beta + N)}{\Gamma(\alpha + h)\Gamma(\beta + N - h)}\theta^{(\alpha+h-1)}(1-\theta)^{(\beta+N-h-1)}.$$

You can get the MAP estimate by differentiating and setting to 0, yielding

$$\hat{\theta} = \frac{\alpha - 1 + h}{\alpha + \beta - 2 + N}.$$

This has rather a nice interpretation. You can see α and β as extra counts of heads (resp. tails) that are added to the observed counts. So, for example, if you were fairly sure that the coin should be fair, you might make α and β large and equal. When $\alpha = 1$ and $\beta = 1$, we have a uniform prior as in the previous examples.

Worked example 8.12 *More sweary politicians*

We observe our swearing politician for N intervals, seeing n_i swear words in the i'th interval. We model the swearing with a Poisson model. We wish to estimate the intensity, which we write θ. We use a Gamma distribution for the prior on θ. What is the MAP estimate of θ?

Solution: Write $T = \sum_{i=1}^{N}$. We have that

$$p(\theta|\mathcal{D}) = \frac{(\beta+N)^{(\alpha+T)}}{\Gamma(\alpha+T)}\theta^{(\alpha-1+T)}e^{-(\beta+T)\theta}$$

and the MAP estimate is

$$\hat{\theta} = \frac{(\alpha-1+T)}{(\beta+N)}$$

(which you can get by differentiating with respect to θ, then setting to zero). Notice that if β is close to zero, you can interpret α as extra counts; if β is large, then it strongly discourages large values of $\hat{\theta}$, even if the counts are large.

Worked example 8.13 *MAP for Normal prior and likelihood*

We wish to estimate a parameter θ. The prior distribution for θ is normal, with known mean μ_π and known standard deviation σ_π. We have a single data item x. What is the MAP estimate of θ?

Solution: The equations are in a box, above (page 242). A normal distribution has a maximum at the mean, so

$$\hat{\theta} = \frac{cx\sigma_\pi^2 + \mu_\pi\sigma_m^2}{\sigma_m^2 + c^2\sigma_\pi^2}$$

Useful Fact: 8.4 *Bayesian inference is particularly good with little data*

If you have few data items and a model and a reasonable choice of prior, you really should use Bayesian inference.

8.2.4 Filtering

We can make online estimates of the maximum likelihood value of mean and standard deviation for a normal distribution. Assume that, rather than seeing N elements of a dataset in one go, you get to see each one once, and you cannot store them. Assume that this dataset is modelled as normal data. Write $\hat{\mu}_k$ for the maximum likelihood estimate of the mean based on data items $1 \dots k$ (and $\hat{\sigma}_k$ for the maximum likelihood estimate of the standard deviation, etc.). Notice that

$$\hat{\mu}_{k+1} = \frac{(k\hat{\mu}_k) + x_{k+1}}{(k+1)}$$

and that

$$\hat{\sigma}_{k+1} = \sqrt{\frac{(k\hat{\sigma}_k^2) + (x_{k+1} - \hat{\mu}_{k+1})^2}{(k+1)}}$$

This means that you can incorporate new data into your estimate as it arrives without keeping all the data. This process of updating a representation of a dataset as new data arrives is known as **filtering**.

This is particularly useful in the case of normal posteriors. Recall that if we have a normal prior and a normal likelihood, the posterior is normal. Now consider a stream of incoming measurements. Our initial representation of the parameters we are trying to estimate is the prior, which is normal. We allow ourselves to see one measurement, which has normal likelihood; then the posterior is normal. You can think of this posterior as a prior for the parameter estimate based on the *next* measurement. But we know what to do with a normal prior, a normal likelihood, and a measurement; so we can incorporate the measurement and go again. This means we can exploit our expression for the posterior mean and standard deviation in the case of normal likelihood and normal prior and a single measurement to deal with multiple measurements very easily.

Useful Fact: 8.5 *Normal posteriors can be updated online*

Assume we wish to estimate a parameter θ. The prior distribution for θ is normal, with known mean μ_π and known standard deviation σ_π. All data is normal conditioned on θ. We have already received k data items. The posterior $p(\theta|x_1, \ldots, x_k, c_1, \ldots, c_k, \sigma_{n1}, \ldots, \sigma_{nk}, \mu_\pi, \sigma_\pi)$ is normal, with mean μ_k and standard deviation σ_k. We receive a new data item x_{k+1}. The likelihood of this data item is normal with mean $c_{k+1}\theta$ and standard deviation $\sigma_{n(k+1)}$, where c_{k+1} and $\sigma_{n(k+1)}$ are known. Then the posterior, $p(\theta|x_1, \ldots, x_{k+1}, c_1, \ldots, c_k, c_{k+1}, \sigma_{n1}, \ldots, \sigma_{n(k+1)}, \mu_\pi, \sigma_\pi)$, is normal, with mean

$$\mu_{k+1} = \frac{c_{k+1}x_{k+1}\sigma_k^2 + \mu_k\sigma_{n(k+1)}^2}{\sigma_{n(k+1)}^2 + c_{k+1}^2\sigma_k^2}$$

and

$$\sigma_{k+1}^2 = \frac{\sigma_{n(k+1)}^2\sigma_k^2}{\sigma_{n(k+1)}^2 + c_{k+1}^2\sigma_k^2}.$$

Again, notice the very useful fact that, if everything is normal, we can update our posterior representation when new data arrives using a very simple recursive form.

Worked example 8.14 *Normal data*

Assume you see N datapoints x_i which are modelled by a normal distribution with unknown mean θ and with known standard deviation σ. You model the prior on θ using a normal distribution with mean μ_0 and standard deviation σ_0. What is the MAP estimate of the mean?

Solution: Recall that the maximum value of a normal distribution occurs at its mean. Now problem is covered by useful fact 8.3, but in this case we have $c_i = 1$ for each data point, and $\sigma_i = \sigma$. We can write

$$\mu_N = \frac{x_N \sigma_{N-1}^2 + \mu_{N-1}\sigma^2}{\sigma^2 + \sigma_{N-1}^2}$$

and

$$\sigma_N^2 = \frac{\sigma^2 \sigma_{N-1}^2}{\sigma^2 + \sigma_{N-1}^2}.$$

and evaluate the recursion down to μ_0, σ_0.

8.2.5 Cautions about Bayesian Inference

Just like maximum likelihood inference, bayesian inference is not a recipe that can be applied without thought. It turns out that, when there is a lot of data, the prior has little influence on the outcome of the inference, and the MAP solution looks a lot like the maximum likelihood solution. So the difference between the two approaches is most interesting when there is little data, where the prior matters. The difficulty is that it might be hard to know what to use as a good prior. In the examples, I emphasized mathematical convenience, choosing priors that lead to clean posteriors. There is no reason to believe that nature uses conjugate priors (even though conjugacy is a neat property). How should one choose a prior for a real problem?

This isn't an easy point. If there is little data, then the choice could really affect the inference. Sometimes we're lucky, and the logic of the problem dictates a choice of prior. Mostly, we have to choose and live with the consequences of the choice. Often, doing so is succesful in applications.

The fact we can't necessarily justify a choice of prior seems to be one of life's inconveniences, but it represents a significant philosophical problem. It's been at the core of a long series of protracted, often quite intense, arguments about the philosophical basis of statistics. I haven't followed these arguments closely enough to summarize them; they seem to have largely died down without any particular consensus being reached.

8.3 SAMPLES, URNS AND POPULATIONS

Very often the data we see is a small part of the data we could have seen, if we'd been able to collect enough data. We need to know how the measurements we make

on the dataset relate to the measurements we could have made, if we had all the data. This situation occurs very often. For example, imagine we wish to know the average weight of a rat. This isn't random; you could weigh every rat on the planet, and then average the answers. But doing so would absurd (among other things, you'd have to weigh them all at the same time, which would be tricky). Instead, we weigh a small set of rats, chosen rather carefully. If we have chosen sufficiently carefully, then the answer from the small set is quite a good representation of the answer from the whole set.

The data we could have observed, if we could have seen everything, is the **population**. The data we actually have is the **sample**. We would like to know the mean of the population, but can see only the sample; surprisingly, we can say a great deal from the sample alone, assuming that it is chosen appropriately.

8.3.1 Estimating the Population Mean from a Sample

Assume we have a population $\{x\}$, for $i = 1, \ldots, N_p$. Notice the subscript here — this is the number of items in the population. The population could be unreasonably big: for example, it could consist of all the people in the world. We want to know the mean of this dataset, but we do not get to see the whole dataset. Instead, we see a sample.

How the sample is obtained is key to describing the population. We will focus on only one model (there are lots of others). In our model, the sample is obtained by choosing a fixed number of data items. Write k for the number of data items in the sample. We expect k is a lot smaller than N_p. Each item is chosen independently, and fairly. This means that each time we choose, we choose one from the entire set of N_p data items, and each has the same probability of being chosen. This is sometimes referred to as "sampling with replacement".

One natural way to think about sampling with replacement is to imagine the data items as being written on tickets, which are placed in an urn (old-fashioned word for a jar, now used mainly by statisticians and morticians). You obtain the sample by repeating the following experiment k times: shake the urn; take a ticket from the urn and write down the data on the ticket; put it back in the urn. Notice that, in this case, each sample is drawn from the same urn. This is important, and makes the analysis easier. If we had not put the ticket back, the urn would change between samples.

We summarize the whole dataset with its mean, which we write $\mathsf{popmean}(\{x\})$. This is known as the **population mean**. The notation is just to drive home the facts that it's the mean of the whole population, and that we don't, and can't, know it. The whole point of this exercise is to *estimate* this mean.

We would like to estimate the mean of the whole dataset from the items that we actually see. Imagine we draw k tickets from the urn as above, and average the values. The result is a random variable, because different draws of k tickets will give us different values. Write $X^{(k)}$ for this random variable, which is referred to as the **sample mean**. Because expectations are linear, we must have that

$$\mathbb{E}\left[X^{(k)}\right] = \frac{1}{k}\left(\mathbb{E}\left[X^{(1)}\right] + \ldots + \mathbb{E}\left[X^{(1)}\right]\right) = \mathbb{E}\left[X^{(1)}\right]$$

(where $X^{(1)}$ is the random variable whose value is obtained by drawing one ticket

from the urn). Now

$$\begin{aligned}
\mathbb{E}\left[X^{(1)}\right] &= \sum_{i \in 1,\ldots N_p} x_i p(i) \\
&= \sum_{i \in 1,\ldots N_p} x_i \frac{1}{N_p} \qquad \text{because we draw fairly from the urn} \\
&= \frac{\sum_{i \in 1,\ldots N_p} x_i}{N_p} \\
&= \mathsf{popmean}\left(\{x\}\right)
\end{aligned}$$

which is the mean value of the items in the urn. This means that

$$\mathbb{E}\left[X^{(k)}\right] = \mathsf{popmean}\left(\{x_i\}\right).$$

Under our sampling model, the expected value of the sample mean is the population mean.

Useful Facts: 8.6 *Properties of sample and population means*

The sample mean is a random variable. It is random, because different samples from the population will have different values of the sample mean. The expected value of this random variable is the population mean.

We will not get the same value of $X^{(k)}$ each time we perform the experiment, because we see different data items in each sample. So $X^{(k)}$ has variance, and this variance is important. If it is large, then each estimate is quite different. If it is small, then the estimates cluster. Knowing the variance of $X^{(k)}$ would tell us how accurate our estimate of the population mean is.

8.3.2 The Variance of the Sample Mean

We write $\mathsf{popsd}\left(\{x\}\right)$ for the standard deviation of the whole population of $\{x\}$. Again, we write it like this to keep track of the facts that (a) it's for the whole population and (b) we don't — and usually can't — know it.

We can compute the variance of $X^{(k)}$ (the sample mean) easily. We have

$$\mathsf{var}\left[X^{(k)}\right] = \mathbb{E}\left[(X^{(k)})^2\right] - \mathbb{E}\left[X^{(k)}\right]^2 = \mathbb{E}\left[(X^{(k)})^2\right] - \left(\mathsf{popmean}\left(\{x\}\right)\right)^2$$

so we need to know $\mathbb{E}\left[(X^{(k)})^2\right]$. We can compute this by writing

$$X^{(k)} = \frac{1}{k}(X_1 + X_2 + \ldots X_k)$$

where X_1 is the value of the first ticket drawn from the urn, etc. We then have

$$X^{(k)2} = \left(\frac{1}{k}\right)^2 \left(\begin{array}{l} {X^2}_1 + {X^2}_2 + \ldots {X^2}_k + X_1X_2 + \ldots \\ X_1X_k + X_2X_1 + \ldots X_2X_k + \ldots X_{k-1}X_k \end{array} \right).$$

Expectations are linear, so we have that

$$\mathbb{E}\left[(X^{(k)})^2\right] = \left(\frac{1}{k}\right)^2 \left(\begin{array}{l} \mathbb{E}\left[{X^2}_1\right] + \mathbb{E}\left[{X^2}_2\right] + \ldots \mathbb{E}\left[{X^2}_k\right] + \mathbb{E}[X_1X_2] + \\ \ldots \mathbb{E}[X_1X_k] + \mathbb{E}[X_2X_1] + \ldots \mathbb{E}[X_2X_k] + \ldots \mathbb{E}[X_{k-1}X_k] \end{array} \right).$$

The *order* in which the tickets are drawn from the urn doesn't matter, because each time we draw a ticket we draw from the same urn. This means that $\mathbb{E}\left[{X^2}_1\right] = \mathbb{E}\left[{X^2}_2\right] = \ldots \mathbb{E}\left[{X^2}_k\right]$. You can think of this term as the expected value of the random variable generated by: drawing a single number out of the urn; squaring that number; and reporting the square. Notice that $\mathbb{E}\left[{X^2}_1\right] = \mathbb{E}\left[(X^{(1)})^2\right]$ (look at the definition of $X^{(1)}$).

Because the *order* doesn't matter, we also have that $\mathbb{E}[X_1X_2] = \mathbb{E}[X_1X_3] = \ldots \mathbb{E}[X_{k-1}X_k]$. You can think of this term as the expected value of the random variable generated by: drawing a number out of the urn; writing it down; returning it to the urn; then drawing a second number from the urn; and reporting the product of these two numbers. So we can write

$$\mathbb{E}\left[X^{(k)2}\right] = (\frac{1}{k})^2 \left(k\mathbb{E}\left[(X^{(1)})^2\right] + k(k-1)\mathbb{E}[X_1X_2]\right)$$

and these two terms are quite easy to evaluate.

Worked example 8.15 *Urn variances*

Show that

$$\mathbb{E}\left[(X^{(1)})^2\right] = \frac{\sum_{i=1}^{N_p} x_i^2}{N_p} = \mathsf{popsd}\left(\{x\}\right)^2 + \mathsf{popmean}\left(\{x\}\right)^2$$

Solution: First, we have $(X^{(1)})^2$ is the number obtained by taking a ticket out of the urn uniformly and at random and squaring its data item. Now

$$\mathsf{popsd}\left(\{x\}\right)^2 = \mathbb{E}\left[(X^{(1)})^2\right] - \mathbb{E}\left[X^{(1)}\right]^2 = \mathbb{E}\left[(X^{(1)})^2\right] - \mathsf{popmean}\left(\{x\}\right)^2$$

so

$$\mathbb{E}\left[(X^{(1)})^2\right] = \mathsf{popsd}\left(\{x\}\right)^2 + \mathsf{popmean}\left(\{x\}\right)^2$$

Worked example 8.16 *Urn variances*

Show that

$$\mathbb{E}[X_1 X_2] = \mathsf{popmean}\left(\{x\}\right)^2$$

Solution: This looks hard, but isn't. Recall from the facts in chapter 5 (useful facts 5.6, page 140) that if X and Y are independent random variables, $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$. But X_1 and X_2 are independent — they are different random draws from the same urn. So

$$\mathbb{E}[X_1 X_2] = \mathbb{E}[X_1]\mathbb{E}[X_2]$$

but $\mathbb{E}[X_1] = \mathbb{E}[X_2]$ (they are draws from the same urn) and $\mathbb{E}[X] = \mathsf{popmean}\left(\{x\}\right)$. So

$$\mathbb{E}[X_1 X_2] = \mathsf{popmean}\left(\{x\}\right)^2.$$

Now

$$\begin{aligned}
\mathbb{E}\left[(X^{(k)})^2\right] &= \frac{k\mathbb{E}\left[(X^{(1)})^2\right] + k(k-1)\mathbb{E}[X_1 X_2]}{k^2} \\
&= \frac{\mathbb{E}\left[(X^{(1)})^2\right] + (k-1)\mathbb{E}[X_1 X_2]}{k} \\
&= \frac{(\mathsf{popsd}\left(\{x\}\right)^2 + \mathsf{popmean}\left(\{x\}\right)^2) + (k-1)\mathsf{popmean}\left(\{x\}\right)^2}{k} \\
&= \frac{\mathsf{popsd}\left(\{x\}\right)^2}{k} + \mathsf{popmean}\left(\{x\}\right)^2
\end{aligned}$$

so we have

$$\begin{aligned}
\mathsf{var}\left[X^{(k)}\right] &= \mathbb{E}\left[(X^{(k)})^2\right] - \mathbb{E}\left[X^{(k)}\right]^2 \\
&= \frac{\mathsf{popsd}\left(\{x\}\right)^2}{k} + \mathsf{popmean}\left(\{x\}\right)^2 - \mathsf{popmean}\left(\{x\}\right)^2 \\
&= \frac{\mathsf{popsd}\left(\{x\}\right)^2}{k}.
\end{aligned}$$

This is a very useful result which is well worth remembering together with our facts on the sample mean, so we'll put them in a box together.

Useful Fact: 8.7 *Mean and variance of the sample mean*

The sample mean is a random variable. Write $X^{(k)}$ for the mean of k samples. We have that:

$$\begin{aligned}
\mathbb{E}\left[X^{(k)}\right] &= \mathsf{popmean}\left(\{x\}\right) \\
\mathsf{var}\left[X^{(k)}\right] &= \frac{\mathsf{popsd}\left(\{x\}\right)^2}{k} \\
\mathsf{std}\left(\left\{X^{(k)}\right\}\right) &= \frac{\mathsf{popsd}\left(\{x\}\right)}{\sqrt{k}}
\end{aligned}$$

The consequence is this: If you draw k samples, the standard deviation of your estimate of the mean is

$$\frac{\mathsf{popsd}\left(\{x\}\right)}{\sqrt{k}}$$

which means that (a) the more samples you draw, the better your estimate becomes and (b) the estimate improves rather slowly — for example, to halve the standard deviation in your estimate, you need to draw four times as many samples. The standard deviation of the estimate of the mean is often known as the **standard error** of the mean. This allows us to draw a helpful distinction: the population has a standard deviation, and our estimate of its mean (or other things — but we won't go into this) has a standard error.

Notice we cannot state the standard error of our estimate exactly, because we do not know $\mathsf{popsd}\left(\{x\}\right)$. But we could make a good estimate of $\mathsf{popsd}\left(\{x\}\right)$, by computing the standard deviation of the examples that we have. It is now helpful to have some notation for the particular sample we have. I will write $\sum_{i \in \text{sample}}$ for a sum over the sample items, and we will use

$$\mathsf{mean}\left(\{x\}\right) = \frac{\sum_{i \in \text{sample}} x_i}{k}$$

for the mean of the sample — that is, the mean of the data we actually see; this is consistent with our old notation, but there's a little reindexing to keep track of the fact we don't see all of the population. Similarly, I will write

$$\mathsf{std}\left(\{x\}\right) = \sqrt{\frac{\sum_{i \in \text{sample}} (x_i - \mathsf{mean}\left(\{x_i\}\right))^2}{k}}$$

for the sample standard deviation. Again, this is the standard deviation of the data we actually see; and again, this is consistent with our old notation, again with a

little reindexing to keep track of the fact we don't see all of the population. We could estimate

$$\mathsf{popsd}\left(\{x\}\right) \approx \mathsf{std}\left(\{x\}\right)$$

and as long as we have enough examples, this estimate is good. If the number of samples k is small, it is better to use

$$\mathsf{popsd}\left(\{x\}\right) \approx \sqrt{\frac{\sum_{i \in \text{sample}} (x_i - \mathsf{mean}\left(\{x\}\right))^2}{k-1}}.$$

In fact, much more is known about the distribution of $X^{(k)}$.

8.3.3 The Probability Distribution of the Sample Mean

The sample mean is a random variable. We know an expression for its mean, and we can estimate its variance. In fact, we can determine its probability distribution, though I won't do this rigorously. In section 6.4.3, I mentioned that adding a number of independent random variables almost always got you a normal random variable, a fact sometimes known as the central limit theorem. I didn't prove it, and I'm not going to now. But when we form $X^{(k)}$, we're adding random variables. This means that $X^{(k)}$ is a normal random variable, for sufficiently big k (for some reason, $k > 30$ is usually seen as right).

This is important, because it has the following consequence. Draw a large number of different samples of k elements from the population. Each is a dataset of k items. Compute $\mathsf{mean}\left(\{x\}\right)$ for each, and regard the resulting numbers $e_1, \ldots, e_r$ as data items. Convert the e_i to standard coordinates s_i, where

$$s_i = \frac{(e_i - \mathsf{mean}\left(\{e_i\}\right))}{\mathsf{std}\left(e_i\right)}$$

(i.e. by subtracting the mean of the e_i, and dividing by their standard deviation). Now construct a construct a histogram of the s. If r is sufficiently large, the histogram will be close to the standard normal curve.

Useful Fact: 8.8 *The sample mean is normally distributed*

Write $X^{(k)}$ for the mean of k samples. For sufficiently large k (30 is usually seen as right), $X^{(k)}$ is a normal random variable, with mean $\mathsf{popmean}\left(\{x\}\right)$ and standard deviation $\frac{\mathsf{popsd}(\{x\})}{\sqrt{k}}$.

8.3.4 When The Urn Model Works

In our model, there was a population of N_p data items x_i, and we saw k of them, chosen at random. In particular, each choice was fair (in the sense that each data

item had the same probability of being chosen) and independent. These assumptions are very important for our analysis to apply. If our data does not have these properties, bad things can happen.

For example, assume we wish to estimate the percentage of the population that has beards. This is a mean (the data items take the value 1 for a person with a beard, and 0 without a beard). If we select people according to our model, then ask them whether they have a beard, then our estimate of the percentage of beards should behave as above.

The first thing that should strike you is that it isn't at all easy to select people according to this model. For example, we might select phone numbers at random, then call and ask the first person to answer the phone whether they have a beard; but many children won't answer the phone because they are too small. The next important problem is that errors in selecting people can lead to massive errors in your estimate. For example, imagine you decide to survey all of the people at a kindergarten on a particular day; or all of the people in a women's clothing store; or everyone attending a beard growing competition (they do exist). In each case, you will get an answer that is a very poor estimate of the right answer, and the standard error might look very small. Of course, it is easy to tell that these cases are a bad choice.

It may not be easy to tell what a good choice is. You should notice the similarity between estimating the percentage of the population that wears a beard, and estimating the percentage that will vote for a particular candidate. There is a famous example of a survey that mispredicted the result of the Dewey-Truman presidential election in 1948; poll-takers phoned random phone numbers, and asked for an opinion. But at that time, telephones tended to be owned by a small percentage of rather comfortable households, who tended to prefer one candidate, and so the polls mispredicted the result rather badly.

Sometimes, we don't really have a choice of samples. For example, we might be presented with a small dataset of (say) human body temperatures. If we can be satisfied that the people were selected rather randomly, we might be able to use this dataset to predict expected body temperature. But if we knew that the subjects had their temperatures measured because they presented themselves at the doctor with a suspected fever, then we most likely cannot use it to predict expected body temperature.

One important and valuable case where this model works is in simulation. If you can guarantee that your simulations are independent (which isn't always easy), this model applies to estimates obtained from a simulation. Notice that it is usually straightforward to build a simulation so that the i'th simulation reports an x_i where $\mathsf{popmean}(\{x\})$ gives you the thing you want to measure. For example, imagine you wish to measure the probability of winning a game; then the simulation should report one when the game is won, and zero when it is lost. As another example, imagine you wish to measure the expected number of turns before a game is won; then your simulation should report the number of turns elapsed before the game was won.

8.4 YOU SHOULD

8.4.1 remember these definitions:

8.4.2 remember these terms:

8.4.3 remember these facts:

8.4.4 be able to:

- Write out the likelihood for a set of independent data items produced by models from chapter 6 (at least Normal, Binomial, Multinomial, Poisson, Beta, Gamma, Exponential).
- Write out the log likelihood for a set of independent data items produced by models from chapter 6 (at least Normal, Binomial, Multinomial, Poisson, Beta, Gamma, Exponential).
- Find maximum likelihood solutions for parameters of these models from a set of independent data items (in this case, ignore Beta and Gamma; finding maximum likelihood estimates for these can be tricky, and isn't important to us).
- Describe situations where maximum likelihood estimates might not be reliable.
- Describe the difference between maximum likelihood estimation and Bayesian inference.
- Write an expression for the posterior or log-posterior of model parameters given a set of independent data items.
- Compute the MAP estimate for the cases shown in the worked examples.
- Compute on-line estimates of the maximum likelihood estimate of the mean and standard deviation of a normal model.
- Compute on-line estimates of the MAP estimate of the mean and standard deviation in the case of a normal prior and a normal likelihood.
- Estimate the population mean from a sample mean.
- Estimate the standard error of the estimate of a population mean.

PROBLEMS

Maximum Likelihood Methods

8.1. Fitting a Normal Distribution: You are given a dataset of N numbers. Write x_i for the i'th number. You wish to model this dataset with a normal distribution.

(a) Show the maximum likelihood estimate of the mean of this distribution is $\mathsf{mean}\left(\{x\}\right)$.

(b) Show the maximum likelihood estimate of the standard deviation of this distribution is $\mathsf{std}\left(x\right)$.

(c) Now assume that all of these numbers take the same value - what happens to your estimate of the standard deviation?

8.2. Fitting an Exponential Distribution: You are given a dataset of N non-negative numbers. Write x_i for the i'th number. You wish to model this dataset with an exponential distribution, with probability density function $P(x|\theta) = \theta e^{-\theta x}$. Show the maximum likelihood estimate of θ is

$$\frac{N}{\sum_i x_i}$$

8.3. Fitting a Poisson Distribution: You count the number of times that the annoying "MacSweeper" popup window appears per hour when you surf the web. You wish to model these counts with a Poisson distribution. On day 1, you surf for 4 hours, and see counts of $3, 1, 4, 2$ (in hours 1 through 4 respectively). On day 2, you surf for 3 hours, and observe counts of $2, 1, 2$. On day 3, you surf for 5 hours, and observe counts of $3, 2, 2, 1, 4$. On day 4, you surf for 6 hours, but keep only the count for all six hours, which is 13. You wish to model the intensity in counts per hour.

(a) What is the maximum likelihood estimate of the intensity for each of days 1, 2, and 3 *separately*?

(b) What is the maximum likelihood estimate of the intensity for day 4?

(c) What is the maximum likelihood estimate of the intensity for all days taken together?

8.4. Fitting a Geometric Model: You wish to determine the number of zeros on a roulette wheel without looking at the wheel. You will do so with a geometric model. Recall that when a ball on a roulette wheel falls into a non-zero slot, odd/even bets are paid; when it falls into a zero slot, they are not paid. There are 36 non-zero slots on the wheel.

(a) Assume you observe a total of r odd/even bets being paid before you see a bet not being paid. What is the maximum likelihood estimate of the number of slots on the wheel?

(b) How reliable is this estimate? Why?

(c) You decide to watch the wheel k times to make an estimate. In the first experiment, you see r_1 odd/even bets being paid before you see a bet not being paid; in the second, r_2; and in the third, r_3. What is the maximum likelihood estimate of the number of slots on the wheel?

8.5. Fitting a Binomial Model: You encounter a deck of Martian playing cards. There are 87 cards in the deck. You cannot read Martian, and so the meaning of the cards is mysterious. However, you notice that some cards are blue, and others are yellow.

(a) You shuffle the deck, and draw one card. It is yellow. What is the maximum likelihood estimate of the fraction of blue cards in the deck?

(b) You repeat the previous exercise 10 times, replacing the card you drew each time before shuffling. You see 7 yellow and 3 blue cards in the deck. What is the maximum likelihood estimate of the fraction of blue cards in the deck?

8.6. Fitting a Least Squares Model: We observe a set of N data items. The i'th data item consists of a vector $\mathbf{x}_i$ and a number y_i. We believe that this data is explained by a model where $P(y|\mathbf{x}, \theta)$ is normal, with mean $\mathbf{x}^T\theta$ and (known) standard deviation σ. Here θ is a vector of unknown parameters. At first sight, this model may strike you as being a bit strange, though we'll do a lot with it later. You can visualize this model by noting that y is generated by (a) forming $\mathbf{x}^T\theta$ then (b) adding a zero-mean normal random variable with standard deviation σ.

(a) Show that a maximum likelihood estimate $\hat{\theta}$ of the value of θ is obtained by solving

$$\sum_i (y_i - \mathbf{x}_i^T\hat{\theta})^2 = 0.$$

(b) Stack the y_i into a vector $\mathbf{y}$ and the $\mathbf{x}_i$ into a matrix $\mathcal{X}$ according to

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{bmatrix} \qquad \mathcal{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \dots \\ \mathbf{x}_N^T \end{bmatrix}.$$

Now show that a maximum likelihood estimate $\hat{\theta}$ of the value of θ is obtained by solving the equation $\mathcal{X}^T\mathcal{X}\hat{\theta} = \mathcal{X}^T\mathbf{y}$.

8.7. Logistic Regression: We observe a set of N data items. The i'th data item consists of a vector $\mathbf{x}_i$ and a discrete y_i, which can take the values 0 or 1. We believe that this data is explained by a model where y_i is a draw from a Bernoulli distribution. The Bernoulli distribution has the property that

$$\log \frac{P(y=1|\mathbf{x}, \theta)}{P(y=0|\mathbf{x}, \theta)} = \mathbf{x}^T\theta.$$

This is known as a logistic model. Here θ is a vector of unknown parameters.

(a) Show that

$$P(y=1|\mathbf{x}, \theta) = \frac{\exp(\mathbf{x}^T\theta)}{1 + \exp(\mathbf{x}^T\theta)}$$

(b) Show that the log-likelihood of a dataset is given by

$$\sum_i \left[y_i \mathbf{x}_i^T\theta - \log\left(1 + \exp(\mathbf{x}_i^T\theta)\right)\right].$$

(c) Now show that a maximum likelihood estimate $\hat{\theta}$ of the value of θ is obtained by solving the equation

$$\sum_i \left[\left(y_i - \frac{\exp(\mathbf{x}_i^T\theta)}{1 + \exp(\mathbf{x}_i^T\theta)} \right)\right] = 0.$$

Bayesian Methods

8.8. Zeros on a Roulette Wheel: We now wish to make a more sophisticated estimate of the number of zeros on a roulette wheel without looking at the wheel. We will do so with Bayesian inference. Recall that when a ball on a roulette wheel falls into a non-zero slot, odd/even bets are paid; when it falls into a zero slot, they are not paid. There are 36 non-zero slots on the wheel. We assume that thenumber of zeros is one of $\{0, 1, 2, 3\}$. We assume that these cases have prior probability $\{0.1, 0.2, 0.4, 0.3\}$.

(a) Write n for the event that, in a single spin of the wheel, an odd/even bet will not be paid (equivalently, the ball lands in one of the zeros). Write z for the number of zeros in the wheel. What is $P(n|z)$ for each of the possible values of z (i.e. each of $\{0, 1, 2, 3\}$)?

(b) Under what circumstances is $P(z = 0|\text{observations})$ NOT 0?

(c) You observe 36 independent spins of the same wheel. A zero comes up in 2 of these spins. What is $P(z|\text{observations})$?

8.9. Which random number generator? You have two random number generators. R1 generates numbers that are distributed uniformly and at random in the range -1 to 1. R2 generates standard normal random variables. A program chooses one of these two random number generators uniformly and at random, then using that generator produces three numbers x_1, x_2 and x_3.

(a) Write $R1$ for the event the program chose R1, etc. When is $P(R1|x_1, x_2, x_3) = 0$?

(b) You observe $x_1 = -.1$, $x_2 = .4$ and $x_3 = -.9$. What is $P(R1|x_1, x_2, x_3)$?

8.10. Which random number generator, again? You have $r > 1$ random number generators. The i'th random number generator generates numbers that are distributed normally, with zero mean and standard deviation i. A program chooses one of these random number generators uniformly and at random, then using that generator produces N numbers $x_1, \ldots, x_N$. Write R_i for the event the program chose the i'th random number generator, etc. Write an expression for $P(R_i|x_1, \ldots, x_N) = 0$

8.11. Which random number generator, yet again? You have $r > 1$ random number generators. The i'th random number generator generates numbers that are distributed normally, with mean i and standard deviation 2. A program chooses one of these random number generators uniformly and at random, then using that generator produces N numbers $x_1, \ldots, x_N$. Write R_i for the event the program chose the i'th random number generator, etc. Write an expression for $P(R_i|x_1, \ldots, x_N) = 0$

8.12. A Normal Distribution: You are given a dataset of 3 numbers, $-1, 0, 20$. You wish to model this dataset with a normal distribution with unknown mean μ and standard deviation 1. You will make an MAP estimate of μ. The prior on μ is normal, with mean 0 and standard deviation 10.

(a) What is the MAP estimate of μ?

(b) A new datapoint, with value 1, arrives. What is the new MAP estimate of μ?

Samples and Populations

8.13. The Average Mouse: You wish to estimate the average weight of a mouse. You obtain 10 mice, sampled uniformly at random and with replacement from the mouse population. Their weights are $21, 23, 27, 19, 17, 18, 20, 15, 17, 22$

grams respectively.

(a) What is the best estimate for the average weight of a mouse, from this data?

(b) What is the standard error of this estimate?

(c) How many mice would you need to reduce the standard error to 0.1?

8.14. Sample Variance and Standard Error: You encounter a deck of Martian playing cards. There are 87 cards in the deck. You cannot read Martian, and so the meaning of the cards is mysterious. However, you notice that some cards are blue, and others are yellow.

(a) You shuffle the deck, and draw one card. You repeat this exercise 10 times, replacing the card you drew each time before shuffling. You see 7 yellow and 3 blue cards in the deck. As you know, the maximum likelihood estimate of the fraction of blue cards in the deck is 0.3. What is the standard error of this estimate?

(b) How many times would you need to repeat the exercise to reduce the standard error to 0.05?

PROGRAMMING EXERCISES

8.15. One interesting way to evaluate maximum likelihood estimation is to use simulations. We will compare maximum likelihood estimates to true parameter values for normal distributions. Write a program that draws s sets of k samples from a normal distribution with mean zero and standard deviation one. Now compute the maximum likelihood estimate of the mean of this distribution from each set of samples. You should see s different values of this estimate, one for each set. How does the variance of the estimate change with k?

8.16. Write a program that draws a sample of a Bernoulli random variable δ with $P(\delta = 1) = 0.5$. If this sample takes the value 1, your program should draw a sample from a normal distribution with mean zero and standard deviation 1. Otherwise, your program should draw a sample from a normal distribution with mean 1 and standard deviation 1. Write x_1 for the resulting sample. We will use x_1 to infer the value of δ_1.

(a) Show that

$$\delta = \begin{cases} 1 & \text{if} x_1 < 0.5 \\ 0 & \text{otherwise} \end{cases}$$

is a maximum likelihood estimate of δ_1.

(b) Now draw 100 sets each of one sample from this program, and infer for each the value of δ_1. How often do you get the right answer?

(c) Show that the true error rate is

$$2 \int_{0.5}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left[\frac{(u-1)^2}{2}\right] du.$$

(Hint: I did a devious change of variables to get the leading 2). Compare the number obtained from your simulation with an estimate of this integral using the error function.

8.17. Logistic regression: We observe a set of N data items. The i'th data item consists of a vector $\mathbf{x}_i$ and a discrete y_i, which can take the values 0 or 1. We believe that this data is explained by a model where y_i is a draw from a

Bernoulli distribution. The Bernoulli distribution has the property that

$$\log \frac{P(y=1|\mathbf{x},\theta)}{P(y=0|\mathbf{x},\theta)} = \mathbf{x}^T\theta.$$

This is known as a logistic model. Here θ is a vector of unknown parameters. Inferring θ is often known as logistic regression. We will investigate logistic regression using simulation.

(a) Write a program that accepts 10 dimensional vector θ and then (a) generates a sample of 1000 samples $\mathbf{x}_i$, each of which is an IID sample from a normal distribution with mean zero and covariance matrix the identity matrix; and (b) for each $\mathbf{x}_i$, forms y_i which is a sample from a Bernoulli distribution where

$$P(y_i = 1|\mathbf{x}_i,\theta) = \frac{\exp \mathbf{x}_i^T\theta}{1+\exp \mathbf{x}_i^T\theta}.$$

This is a sample dataset. Call this program the dataset maker.

(b) Write a program that accepts a sample dataset and estimates $\hat{\theta}$ (the value of θ that was used to generate the sample dataset) using maximum likelihood. This will require that you use some optimization code, or write your own. Call this program the inference engine.

(c) Choose a θ (a sample from a normal distribution with mean zero and covariance matrix the identity matrix is one way to do this), then create 100 sample datasets. For each, apply the inference engine, and obtain a $\hat{\theta}$. How does the mean of these estimates of θ compare to the true value? What are the eigenvalues of the covariance of these estimates like?

(d) Choose a θ (a sample from a normal distribution with mean zero and covariance matrix the identity matrix is one way to do this), then create 2 sample datasets. For the first, apply the inference engine, and obtain a $\hat{\theta}$. Now use this $\hat{\theta}$ to predict the y_i values of the second, using the logistic regression model. How do your predictions compare to the true values? should the difference be zero? why?

CHAPTER 9

Inference: Intervals and Testing

Although a point estimate is the best estimate available, it could still be wrong. This doesn't always matter. But in some applications, it can be important to know what range a parameter could take, and still be consistent with the data. We have seen this possibility before, in example 8.8, where I graphed the posterior — you could look at that graph and see that p could be in a fairly large range of values.

Being able to estimate an interval of values is important, particularly when there are safety or legal considerations to worry about. Imagine you have a machine that fills cereal boxes. Each box gets a quantity of cereal that is random, but has low variance. If the weight of cereal in any box is below the amount printed on the label, you might be in trouble. When you choose the amount to print on the label, a point estimate of the mean weight of cereal might not be particularly helpful. If that estimate is a little low, you could have problems. Instead, what you'd like to know is an interval that the mean lies in with very high probability. Then you can print a label number that is smaller than the smallest in the interval, and be confident that the amount in the box is more than the amount on the label.

Some more detail is required to understand another application. Imagine I want to test the idea that the mean human body weight is 72kg. The mean human weight isn't a random number, but it's very hard to measure directly. I am forced to take a sample, and compute the sample mean. This sample mean is a random number, and it will have different values for different samples. I need to know how to tell whether the difference between the observed value and 72kg is just be an effect of sampling variance, or is because the mean weight actually isn't 72kg. It turns out that I can construct an interval around the sample mean within which the true value will lie for (say) 99% of possible samples. If 72kg is outside that interval, then very few samples are consistent with my idea; in turn, if I want to believe the mean human body weight is 72kg, I have to believe that I obtained a very odd sample.

This reasoning can be extended to compare populations. Imagine I want to know whether mice weigh more than rats. For practical reasons, I will estimate the weights using a sample. But this means that two different estimates will have different values purely because I used different samples. I am now in a difficult position – perhaps my observed sample mean for the weight of mice is smaller than that for rats because of random variation in the sample value. A very large difference may be hard to explain with random variation unless you are willing to believe in very odd samples. This leads to a procedure that can be used to decide whether (say) mice weigh more than rats.

9.1 CONFIDENCE INTERVALS

A **statistic** is a function of a dataset. One example of a statistic is the mean of a sample. Yet another is the MAP estimate of a parameter (which you get by computing some numbers from the dataset alone). We observe the value of a statistic, which we can compute from our dataset. But the dataset is random, because it is either a sample from a population or an IID sample from a distribution mode. This means that we should think of the statistic as the observed value of a random variable — if we had a different sample from the same population (resp. IID sample from the same distribution) we would compute a different value.

We would like to know an interval such that the true value of the statistic lies in that interval with high probability. For example, assume my dataset is a single draw from a standard normal distribution. Then I know that the mean of my dataset lies in the range $[-1, 1]$ with probability about 0.68. As another example, assume that my dataset consists of 16 IID samples from a standard normal distribution. Then I know the mean of my dataset lies in the range $[-.25, .25]$ with probability about 0.68 (if you're not sure about this, you should think it through). Such intervals are usually known as **confidence intervals**.

Usually, we don't know precisely what model the dataset comes from, which means that constructing confidence intervals can take some ingenuity. There are two cases that are quite straightforward: confidence intervals for population means, and confidence intervals for Bayesian inference.

9.1.1 Confidence Intervals for Population Means

Assume we have a population, and we wish to know a confidence interval for its mean. We draw a sample $\{x\}$ of k items, and compute the sample mean. This is the value of a random variable — random, because it depends on the randomly drawn sample — whose probability distribution we know. This knowledge is very powerful, because it tells us how close to the true mean our estimate is likely to be. The reasoning looks like this. Our estimate of the unknown number $\mathsf{popmean}(\{x\})$ is the mean of the sample we have, which we write $\mathsf{mean}(\{x\})$. We know that the error — which is

$$\mathsf{mean}(\{x\}) - \mathsf{popmean}(\{x\})$$

— is a normal random variable, with mean 0 and standard deviation

$$\frac{\mathsf{popsd}(\{x\})}{\sqrt{k}}.$$

We cannot evaluate this standard deviation because we do not — and, in practice, can not — know $\mathsf{popsd}(\{x\})$. But we can *estimate* $\mathsf{popsd}(\{x\})$. The population standard deviation is the square root of the population variance, and the population variance is the average of $(x_i - \mathsf{mean}(\{x\}))^2$ over the population. As a result, when we sample the population, the sample variance behaves in a familiar way. The sample variance is a random variable (because the sample was chosen randomly). Its expected value is the population variance (because it's the mean of something). Rather than determine the standard deviation of the sample variance, we will assume that it is small (which is true for a reasonably sized sample), and then

pretend that there is no *significant* variation in the value from sample to sample. If we make these assumptions, then we can estimate $\mathsf{popsd}\left(\{x\}\right) \approx \mathsf{std}\left(\{x\}\right)$.

With this approximation, we estimate the standard deviation of the sample mean to be

$$\frac{\mathsf{std}\left(\{x\}\right)}{\sqrt{k}}.$$

We can scale the error by the standard deviation to get

$$M = \frac{\mathsf{mean}\left(\{x\}\right) - \mathsf{popmean}\left(\{x\}\right)}{\left(\frac{\mathsf{popsd}(\{x\})}{\sqrt{k}}\right)} \approx \frac{\mathsf{mean}\left(\{x\}\right) - \mathsf{popmean}\left(\{x\}\right)}{\left(\frac{\mathsf{std}(\{x\})}{\sqrt{k}}\right)}$$

which is a standard normal random variable. But we know rather a lot about the behaviour of standard normal random variables. For about 68% of samples, the M will lie between -1 and 1, and so on. In turn, this means that for about 68% of samples, the sample mean will lie in the interval between $\mathsf{mean}\left(\{x\}\right) - \frac{\mathsf{std}(\{x\})}{\sqrt{k}}$ and $\mathsf{mean}\left(\{x\}\right) + \frac{\mathsf{std}(\{x\})}{\sqrt{k}}$, and so on. Remember that

$$\frac{\mathsf{std}\left(\{x\}\right)}{\sqrt{k}}$$

is called the **standard error**; I will write

$$\mathsf{stderr}\left(\{x\}\right) = \frac{\mathsf{std}\left(\{x\}\right)}{\sqrt{k}}.$$

In particular, we have

Useful Facts: 9.1 *How often normal random variables lie close to the mean*

- About 68% of the time, the value of a standard normal random variable is within one standard deviation of the mean.
- About 95% of the time, the value of a standard normal random variable is within two standard deviations of the mean.
- About 99% of the time, the value of a standard normal random variable is within three standard deviations of the mean.

In turn, this means that

- About 68% of the time:

$$\mathsf{abs}\left(\mathsf{mean}\left(\{x\}\right) - \mathsf{popmean}\left(\{x\}\right)\right) \leq \frac{\mathsf{popsd}\left(\{x\}\right)}{\sqrt{k}} \approx \mathsf{stderr}\left(\{x\}\right).$$

- About 95% of the time:

$$\mathsf{abs}\left(\mathsf{mean}\left(\{x\}\right) - \mathsf{popmean}\left(\{x\}\right)\right) \leq 2\frac{\mathsf{popsd}\left(\{x\}\right)}{\sqrt{k}} \approx 2\mathsf{stderr}\left(\{x\}\right).$$

- About 99% of the time:

$$\mathsf{abs}\left(\mathsf{mean}\left(\{x\}\right) - \mathsf{popmean}\left(\{x\}\right)\right) \leq 3\frac{\mathsf{popsd}\left(\{x\}\right)}{\sqrt{k}} \approx 3\mathsf{stderr}\left(\{x\}\right).$$

We can plot the confidence interval by drawing **error bars** — draw a bar one (or two, or three) standard errors up and down from the estimate. We interpret this interval as representing the effect of sampling uncertainty on our estimate. If the urn model really did apply, then the confidence intervals have the property that the true mean lies inside the interval for about 68% of possible samples (for one standard error error bars; or 95% for two; etc.).

Worked example 9.1 *The weight of female mice eating chow*

Give a 95% confidence interval for the weight of a female mouse who ate chow, based on the dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml.

Solution: There are great datasets dealing with a wide range of genotype and phenotype variations in mice at this URL. The one to look at is Churchill.Mamm.Gen.2012.phenotypes.csv, which has information about 150 mice. 100 were fed with chow, and 50 with a high fat diet. There's lots of phenotype information here. You should look at Weight2, which seems to be a weight computed later in life (as far as I can tell, around the time the mouse was sacrificed). Notice some values are missing. If we focus on the female mice who ate chow and whose weights are available (48 by my count), we find a mean weight of 27.78 gr, and a standard error of 0.70 gr (remember to divide by the square root of 48). This means that the interval we want runs from 26.38 to 29.18 gr.
The authors ask that anyone using this data should cite the papers: *High-Resolution Genetic Mapping Using the Mouse Diversity Outbred Population*, Svenson KL, Gatti DM, Valdar W, Welsh CE, Cheng R, Chesler EJ, Palmer AA, McMillan L, Churchill GA. **Genetics**. 2012 Feb;190(2):437-47; and *The Diversity Outbred Mouse Population* Churchill GA, Gatti DM, Munger SC, Svenson KL **Mammalian Genome 2012**, Aug 15.

It is quite straightforward, and informative, to verify these ideas with a simulation. I used the heights column from the bodyfat dataset (from http://www2.stetson.edu/~jrasp/data.htm; look for bodyfat.xls). I removed the single height outlier. I simulated the population using the whole dataset (251 items), then drew numerous samples of various sizes, with replacement. I computed the mean of each of these sets of samples. Figure 9.1 shows a scatter plot of sample means for different samples, using a set of sizes $(9, 16, \ldots, 81)$. I have also plotted the population mean, and the true 1-standard error bars (i.e. using the population standard deviation) for each of these sample sizes. Notice how most sample means lie within the 1-standard error bars, as they should.

Now we might reasonably ask another question. Choose a number $0 < f < 0.5$. We should like to know a confidence interval $[a, b]$ for the true mean M such that $P(M < a) = f$ and $P(M > b) = f$. Notice that the true mean lies in this interval for a fraction $1 - 2f$ of the samples. The interval I have defined is known as a **centered interval**. This is because there are many intervals such that the true mean lies in the interval for this fraction of samples. For example, imagine we write the observed sample mean s. Then the true mean should lie in the range $[-\infty, s]$ for half of the samples we draw. It should also lie in the range $[s, \infty]$ for half the samples we draw. But these intervals are not as interesting as a centered interval.

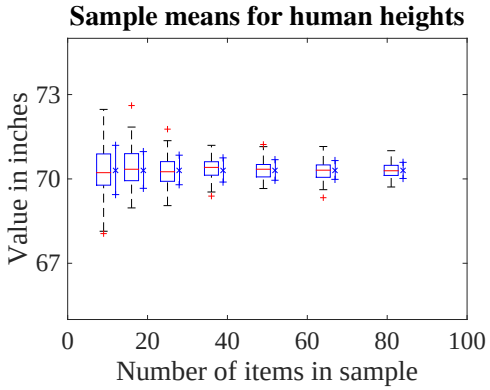

FIGURE 9.1: *I took the heights dataset. I then sampled elements with replacement to form random subsets of sizes* $(9, 16, \ldots, 81)$*. For each of 100 subsets of each size, I computed the sample mean. This means that there are 100 sample means for each sample size. I have represented these means by a boxplot. I then computed the population mean, and the standard error as measured by the* population *standard deviation. The x to the side of each column is the population mean, and the vertical bars are one standard error above and below the population mean. Notice how (a) the sample means vary less as the sample gets bigger and (b) the sample means largely lie within the error bars, as they should.*

Worked example 9.2 *The weight of female mice eating chow, again*

Give a centered 80% confidence interval for the weight of a female mouse who ate chow, based on the dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml.

Solution: From worked example 9.1, the sample mean is 27.78 gr and the standard error is 0.70 gr. Write M for the true mean. We have

$$x = \frac{M - 27.78}{0.70}$$

is a standard normal random variable. So the question is asking for a $u > 0$ such that

$$\int_{-u}^{u} \frac{1}{\sqrt{2\pi}} \exp\left(-x^2/2\right) dx = 0.8$$

(i.e. what is the range of values about zero such that 80% of standard normal random variables lies in this range). Such numbers can be extracted from the inverse of the error function (which is known as the **inverse error function**). An alternative is to look in statistical tables, which usually give u values for a useful set of p's. I found $u = 1.2815$, which gives an interval of 26.88 to 28.68. I've rounded to 1/100'th of a gram, to avoid being silly about how accurately I expect to weigh a mouse.

9.1.2 Bayesian Confidence Intervals

We can use the posterior to evaluate the probability that the true value of a parameter lies within an interval. Assume we want to know for $a < b$, the probability that the parameter θ lies in that interval. In many cases, we can compute

$$\int_a^b p(\theta|\mathcal{D})d\theta,$$

which supplies the relevant probability.

Worked example 9.3 *Flipping a coin*

We have a coin with probability θ of coming up heads when flipped. We start knowing nothing about θ except that, being a probability, $0 \leq \theta \leq 1$. We use a uniform prior on θ. We then flip the coin 10 times, and see 7 heads (and 3 tails). What is $P(\{\theta \in [0.5, 0.8]\}|\mathcal{D})$?

Solution: We could answer this question by computing

$$\int_{0.5}^{0.8} p(\theta|\mathcal{D})d\theta.$$

Recall that a Beta distribution with $\alpha = \beta = 1$ is uniform, and look at example 8.11. The posterior is

$$P(\theta|10, 7, 1, 1) = \frac{\Gamma(12)}{\Gamma(8)\Gamma(4)}\theta^7(1-\theta)^3.$$

I evaluated this integral numerically, and got 0.73.

In example 9.3, I determined that $P(\{\theta \in [0.5, 0.8]\}|\mathcal{D})$ was 0.73. It is often more useful to find an interval $[a, b]$ such that the probability θ is in the interval takes some value. The problem is that there are many such intervals. Usually, what we want is to choose the interval so that $P(\{\theta < a\})$ is the same as $P(\{\theta > b\})$. We choose some u so that $0 \leq u < 0.5$, and then choose a such that

$$P(\{\theta \leq a\}) = \int_{-\infty}^{a} P(\theta|\mathcal{D})d\theta = u$$

and b such that

$$P(\{\theta \geq b\}|\mathcal{D}) = \int_{b}^{\infty} P(\theta|\mathcal{D})d\theta = u.$$

Actually obtaining values for a and b might be quite difficult. One strategy is to search a range of values.

Worked example 9.4 *Flipping a coin - II*

We have a coin with probability θ of coming up heads when flipped. We start knowing nothing about θ except that, being a probability, $0 \leq \theta \leq 1$. We use a uniform prior on θ. We then flip the coin 10 times, and see 7 heads (and 3 tails). Construct an interval $[a, b]$ such that $P(\{\theta \leq a\}|\mathcal{D}) \approx 0.05$ and $P(\{\theta \geq b\}|\mathcal{D}) \approx 0.05$. This is a 90% confidence interval.

Solution: The way to do this is to find an a such that $\int_{-\infty}^{a} p(\theta|\mathcal{D})d\theta \approx 0.05$, and a b such that $\int_{b}^{\infty} p(\theta|\mathcal{D})d\theta \approx 0.05$. I was only able to do these integrals numerically. I wrote a brief program to compute the relevant integrals numerically, then searched for a and b using interval halving. I found $P(\{\theta \leq 0.435\}|\mathcal{D}) \approx 0.05$ and $P(\{\theta \geq 0.865\}|\mathcal{D}) \approx 0.05$; this means the interval is $[0.435, 0.865]$.

9.2 USING THE STANDARD ERROR TO EVALUATE HYPOTHESES

Assume we have a **hypothesis** to work with. For example, we believe that the average human body temperature is 98.4^o in Farenheit. It isn't practical to evaluate the average human body temperature, so we must draw a sample. Even if our hypothesis is true, the average temperature for the sample is likely not to be 98.4^o, because the sample is random, and the sample mean is a random variable. We need to be able to tell whether the difference between 98.4^o and the number we observe can be explained by the randomness produced by sampling.

You should think of this as assessing the extent to which the evidence we have contradicts the hypothesis. At first glance, this may seem strange to you — surely one wants to assess the extent to which the evidence *supports* the hypothesis — but in fact it's natural. You can't prove that a scientific hypothesis is true; you can only fail to show that it's false. Just one piece of evidence can destroy a scientific hypothesis, but no amount of evidence can remove all doubt.

The sample mean is a random variable. But we have a good model of these sample means as random variables. Assuming the hypothesis is true, we know the distribution of the sample mean. It is normal, the expected value is 98.4^o, and its standard deviation is the standard error. We can use this to assess the extent to which the observed mean temperature can be explained solely by the choice of sample. Assume we observe a number many standard errors away from 98.4^o. Then to believe that the hypothesis is true, we also have to believe we have a very unusual sample. There is an important, quite general, line of reasoning here. It is a bad idea to try and explain data using a hypothesis that makes the data you observed a rare event.

Example: 9.1 *Patriot missiles*

I got this example from "Dueling idiots", a nice book by P.J. Nahin, Princeton University Press. Apparently in 1992, the Boston Globe of Jan 24 reported on this controversy. The pentagon claimed that the patriot missile successfully engaged SCUD missiles in 80% of encounters. An MIT physicist, Theodore Postol, pointed out there was a problem. He viewed tapes of 14 patriot/SCUD encounters, with one hit and 13 misses. We can reasonably assume each encounter is independent. We can extract the probability of getting one hit and 13 misses if $P(\text{hit}) = 0.8$ from the binomial model, to get a number around 1e-8. Now you could look at this information and make several arguments: (a) the pentagon is right and the probability really is 0.8, but Postol looked at a really unlucky set of videotapes; (b) the probability is not 0.8, because you would need to fire 14 patriots at 14 SCUD missiles about 1e8 times to see this set of videotapes once; (c) for some reason, the videotapes are not independent — perhaps only unsuccessful encounters get filmed. If Postol viewed tapes at random (i.e. he didn't select only unsuccessful tapes, etc.), then argument (a) is easily dismissed, because the pentagon would have had to be unreasonably unlucky — it's a bad idea to try to explain data with a hypothesis that makes the data very unlikely.

9.2.1 Does This Population have This Mean?

Imagine we hypothesize that the average human body temperature is 95^o. Write T for the random variable evaluated by: collecting a random sample of people, measuring their temperatures, and computing the average of these temperatures. The mean of this random variable is the population mean, and the standard deviation of this random variable is the standard error, which we write s. Now consider the random variable

$$G = \frac{(T - 95^o)}{s}.$$

This is a standard normal random variable, if our hypothesis is true.

We now have a way to tell whether the evidence supports our hypothesis. We assess how strange the sample would have to be to yield the value that we actually see, *if* the hypothesis is true. We can do this, because we know the value we observe is the value of a standard normal random variable *if* the hypothesis is true. If that value is far away from zero, for the hypothesis to be true the sample would have to be extremely unlikely.

We can evaluate our standard normal random variable by computing the mean temperature of the sample, which we write t. This represents an observed value of T. If you're puzzled by the relationship between the two, use an analogy with rolling a die. The random variable is what you could get if you rolled a die; the value is the face that came up when you did roll it. Similarly, T is what you could get by choosing a sample, and t is the value you did get with the sample you have.

In turn, we obtain a value for G, which is

$$g = \frac{(t - 95^o)}{s}.$$

Now G is a standard normal random variable, if our hypothesis is true. We can interpret this to mean that for 68% of possible samples, the value of g will lie between -1 and 1 (etc.), if our hypothesis is true. Equivalently G will take a value between $|g|$ and $-|g|$, for a fraction f of all possible samples of the population, where

$$f = \frac{1}{\sqrt{2\pi}} \int_{-|g|}^{|g|} \exp\left(\frac{-u^2}{2}\right) du.$$

Now assume we see a very large (or very small) value of g. The value of f will be close to one, which implies that most samples from the population will have a g value closer to zero if the hypothesis were true. Equivalently, this says that, if the hypothesis were true, our sample is highly unusual, which implies the data fails to support the hypothesis.

Worked example 9.5 *Samples of 44 male chow eating mice*

Assume the mean weight of a male chow eating mouse is 35 gr. and the standard error of a sample of 44 such mice is 0.827 gr. What fraction of samples of 44 such mice will have a sample mean in the range 33-37 grams?

Solution: You could use the data at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml, but you don't really need it to answer this question. Write S for the sample mean weight of a sample of 44 male chow eating mice. Because we assumed that the true mean weight is 35gr, we have

$$x = \frac{S - 35}{0.827}$$

is a standard normal random variable. The question is asking for the probability that x takes a value in the range $[(33 - 35)/0.827, (37 - 35)/0.827]$, which is $[-2.41, 2.41]$. This is

$$\int_{-2.41}^{2.41} \frac{1}{\sqrt{2\pi}} \exp\left(-x^2/2\right) dx \approx 0.984$$

a number I found in tables. In turn, this means about 98.4% of samples of 44 chow eating male mice will give a mean weight in this range, if the population mean is truly 35 gr.

Worked example 9.6 *Samples of 48 chow-eating female mice.*

Assume the population mean of the weight of a chow-eating female mouse is 27.8 gr. Use the data at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml to estimate the fraction of samples that will have mean weight greater than 29gr.

Solution: From worked example 9.1, the standard error of a sample of 48 chow-eating female mice is 0.70 gr. Write S for a sample mean of a sample of 48 chow-eating female mice. Because we assumed that the true mean was 27.8gr, we have

$$x = \frac{S - 27.8}{0.70}$$

is a standard normal random variable. The question is asking for the probability that x takes a value greater than $(29 - 27.8)/0.7 = 1.7143$, which is

$$\int_{1.7143}^{\infty} \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)dx \approx 0.043$$

a number I found in tables. In turn, this means about 4% of samples of 48 chow eating female mice will give a mean weight greater than 29gr, if the population mean is truly 27.8gr

9.2.2 P-values

It is easy to think about $p = 1 - f$ than about f. You should think of p as representing the fraction of samples that would give a larger absolute value of g, *if* the null hypothesis was true. If this fraction is very small, then there is significant evidence against the null hypothesis. We have that p — the fraction of experiments that would give us a larger absolute value of g than the one we saw — is given by

$$p = (1 - f) = (1 - \frac{1}{\sqrt{2\pi}} \int_{-|g|}^{|g|} \exp\left(\frac{-u^2}{2}\right) du) = P(\{G > |g|\}) \cup P(\{G < -|g|\})$$

and we can compute this number easily, because we know the distribution of G is normal (or nearly so). The fraction is sometimes referred to as a **p-value**.

Worked example 9.7 *If the mean length of an adult male mouse is 10cm, how unusual is the sample in the mouse dataset?*

The mouse dataset is the one at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml. The variable to look at is Length2 (which appears to be the length of the mouse at the point it is sacrificed). We need to compute the p value.

Solution: The mean length of male mice in this data is 9.5cm, and the standard error is 0.045cm. Write L for the sample mean length of some sample of male mice. This (unknown) value is a random variable. Assuming that the mean length really is 10, we have that

$$G = \frac{L - 10}{0.045}$$

is a normal random variable. The value we observe is $g = (9.5 - 10)/0.045 = -11.1$ We are asking for the probability that $G \leq |g|$ AND $G \geq -|g|$. This is so close to 1 that the difference is of no interest to us. In turn, we can interpret this as overwhelming evidence that the mean length isn't 10cm – if it were, the sample in the mouse dataset is quite implausibly unlikely.

Once we have the p-value, testing is straightforward. A small value of the fraction means that the outcome we see would be rare *if* the null hypothesis is true. The fraction that we compute should be interpreted an assessment of the significance of the evidence against the null hypothesis. The p-value is smaller when the evidence against the null hypothesis is stronger; we get to decide how small a p-value means we should reject the null hypothesis.

It is conventional to reject the null hypothesis when the p-value is less than 0.05. This is sometimes called "a significance level of 5%". The phrase can mislead: the term "significance" seems to imply the result is important, or meaningful. Instead, you should interpret a p-value of 0.05 as meaning that you would see evidence this unusual in about one experiment in twenty if the null hypothesis was true. It's quite usual to feel that this means the hypothesis is unlikely to be true. However, notice there is an important danger here. The more samples you look at, the better your chance of seeing one which has a small p-value. If you look at lots of samples or do lots of experiments, looking for one with a small p-value, then use that to argue the hypothesis is false, you are fooling yourself.

Sometimes, the p-value is even smaller, and this can be interpreted as very strong evidence the null hypothesis is wrong. A p-value of less than 0.01 allows one to reject the null hypothesis at "a significance level of 1%". Similarly, you should interpret a p-value of 0.01 as meaning that you would see evidence this unusual in about one experiment in a hundred if the null hypothesis was true. We now have a general recipe for testing whether a population has a particular mean, which belongs in a box:

Procedure: 9.1 *Testing whether a population has a given mean, based on a sample*

The initial hypothesis is that the population has a known mean, which we write μ. Write $\{x\}$ for the sample, and k for the sample size.

- Compute the sample mean, which we write $\mathsf{mean}\,(\{x\})$.
- Estimate the standard error s_e using

$$s_e = \frac{\mathsf{std}\,(x)}{\sqrt{k}}.$$

- Compute the **test statistic** using

$$s = \frac{(\mu - \mathsf{mean}\,(\{x\}))}{s_e}.$$

- Compute the p-value, using

$$p = (1 - f) = (1 - \frac{1}{\sqrt{2\pi}} \int_{-|s|}^{|s|} \exp\left(\frac{-u^2}{2}\right) du)$$

- The p-value summarizes the extent to which the data contradicts the hypothesis. A small p-value implies that, *if* the hypothesis is true, the sample is very unusual. The smaller the p-value, the more strongly the evidence contradicts the hypothesis.

It is common to think that a hypothesis can be rejected only if the p-value is less than 5% (or some number). You should not think this way; the p-value summarizes the extent to which the data contradicts the hypothesis, and your particular application logic affects how you interpret it.

Worked example 9.8 *Average Human Body Weight*

Assess the null hypothesis that the average human body weight is 175 lb, using the height and weight data set of http://www2.stetson.edu/~jrasp/data.htm (called bodyfat.xls).

Solution: The dataset contains 252 samples; the average weight is 178.9lb. We know this average is an estimate of the mean of the original large set of all people. This estimate is a normal random variable whose mean is the mean of the population, and whose standard deviation is given by the standard error. Our sample is large (which usually means over 30 elements) and so we can estimate the standard error as

$$\frac{\text{standard deviation of sample}}{\sqrt{\text{number in sample}}} = \frac{29.4}{15.9} = 1.9,$$

where the units are lb. The test statistic is

$$\frac{\text{sample mean} - \text{hypothesized population mean}}{\text{standard error}} = \frac{178.9 - 175}{1.9} = 2.05.$$

This is the value of a standard normal random variable. We must compute the fraction of outcomes where the test statistic would be larger than 2.05, or smaller than -2.05. This is

$$\left(1 - \frac{1}{\sqrt{2\pi}} \int_{-2.05}^{2.05} \exp\left(\frac{-x^2}{2}\right) dx\right) = 0.02$$

so the p-value is 0.02. We can interpret this as quite strong evidence that the average human body weight is not, in fact, 175lb. This p-value says that, if (a) the average human body weight is 175lb and (b) we repeat the experiment (weigh 252 people and average their weights) 50 times, we would see a number as big as the one we see about once.

Worked example 9.9 *Average BMI*

The BMI (body mass index) is a number intended to capture how much a person's weight deviates from the usual for their height. Assess the null hypothesis that the average human BMI is 27, using the height and weight data set of http://www2.stetson.edu/~jrasp/data.htm; bodyfat.xls.

Solution: BMI is computed using

$$BMI = \frac{\text{weight in lb}}{(\text{height in in})^2} \times 703.$$

I excluded two possible outliers in this dataset (entry 39 has weight 363 and height 72.25; entry 42 has weight 205 and height 29.5). I found a mean BMI of 25.3, and a standard deviation of 3.35. There are 250 items in the dataset, so the standard error is 0.21. The test statistic is

$$\frac{\text{sample mean} - \text{hypothesized population mean}}{\text{standard error}} = \frac{25.3 - 27}{2.1} = -8.1$$

so the p-value is the probability that a standard normal random variable would take a value larger than -8.1 or smaller than 8.1. This number is very small — there is no particular reason to care about the difference between this number and zero. In turn, the evidence is very strongly against the hypothesis.
If one keeps the outliers, one gets a mean BMI of 25.9, and a standard deviation of 9.56. There are 252 items in the dataset, so the standard error is 0.60. Now the p-value is 0.08, suggesting that the evidence is against the hypothesis, but not overwhelmingly. Notice the significant effect of just two datapoints. We might need more data to be sure we could reject the hypothesis.

You should notice an important relationship between our test and the material of section 9.1. When we constructed a confidence interval, we knew the distribution that the sample mean would take as we randomly chose different samples from the population. In turn, this meant we could plot (for example) the range of possible values the population mean would take for (say) 95% of possible samples. When we test a hypothesis about the mean, we are asking what kind of confidence interval we would have to draw around the hypothesized mean to encompass the observed sample mean.

9.2.3 Do Two Populations have the same Mean?

We have two samples, and we need to know whether they come from the same, or different, populations. For example, we might observe people using two different interfaces, measure how quickly they perform a task, then ask are their performances different? As another example, we might run an application with no other applications running, and test how long it takes to run some standard tasks. Because we don't know what the operating system, cache, etc. are up to, this number behaves a bit like a random variable, so it is worthwhile to do several experiments,

yielding one set of samples. We now do this with other applications running as well, yielding another set of samples — is this set different from the first set? For realistic datasets, the answer is always yes, because they're random samples. A better question is could the differences be the result of chance, or are these datasets really samples of two different populations?

Worked example 9.10 *Male and female chow eating mice*

Give a centered 95% confidence interval for the weight of a female mouse who ate chow and for the weight of a male mouse who ate chow, based on the dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml. Compare these intervals.

Solution: We know from worked example 9.1 that the interval we want runs from 26.38 to 29.18 gr. For male mice, the same procedure yields the interval 34.75 to 38.06 gr. Now these two ranges are quite distinct. This means that, to believe the two populations are the same, we'd have to believe we have a really strange sample of at least one. Here is rather compelling evidence that male and female chow-eating mice do not have the same mean weight.

As worked example 9.10 shows, we can reason about populations with confidence intervals on the means. There's a simpler way, using our methods for testing a hypothesis about a population mean, and some tricks about normal random variables. Assume that the samples are large enough (30 seems to be the magic number here). Write $\{x\}$ for the first dataset, which has size k_x, and $\{y\}$ for the second, which has size k_y. These datasets need not be of the same size Each dataset is a random sample, drawn with replacement, from a population. We should like to assess the evidence that these two populations have the same mean. To do so, we need some simple facts about normal random variables.

Useful Facts: 9.2 *Sums and differences of normal random variables*

Let X_1 be a normal random variable with mean μ_1 and standard deviation σ_1. Let X_2 be a normal random variable with mean μ_2 and standard deviation σ_2. Let X_1 and X_2 be independent. Then we have that:

- for any constant $c_1 \neq 0$, $c_1 X_1$ is a normal random variable with mean $c_1\mu_1$ and standard deviation $c_1\sigma_1$;
- for any constant c_2, $X_1 + c_2$ is a normal random variable with mean $\mu_1 + c_2$ and standard deviation σ_1;
- $X_1 + X_2$ is a normal random variable with mean $\mu_1 + \mu_2$ and standard deviation $\sqrt{\sigma_1^2 + \sigma_2^2}$.

I will not prove these facts; we already know the expressions for means and standard deviations from our results on expectations. The only open question is to show that the distributions are normal. This is easy for the first two results. The third requires a bit of integration that isn't worth our trouble; you could reconstruct the proof from section 13.1's notes on sums of random variables and some work with tables of integrals.

Write $X^{(k_x)}$ for the random variable obtained by: drawing a random sample with replacement of k_x elements from the first population, then averaging this sample. Write $Y^{(k_y)}$ for the random variable obtained by: drawing a random sample with replacement of k_y elements from the first population, then averaging this sample. From section 8.3, we know that each random variable is normal, and we know the means and standard deviations of these random variables. In turn, this means that $X^{(k_x)} - Y^{(k_y)}$ is a normal random variable.

Now write D for $X^{(k_x)} - Y^{(k_y)}$. If the two populations have the same mean, then

$$\mathbb{E}[D] = 0.$$

Furthermore,

$$\mathsf{std}\,(D) = \sqrt{\mathsf{std}\left(X^{(k_x)}\right)^2 + \mathsf{std}\left(X^{(k_y)}\right)^2}.$$

We know how to estimate $\mathsf{std}\left(X^{(k_x)}\right)$ — it is the standard error of the sample, and we can estimate it as $\mathsf{std}\,(x)/\sqrt{k_x}$. So we can estimate $\mathsf{std}\,(D)$ as

$$\mathsf{std}\,(D) \approx \sqrt{\left(\frac{\mathsf{std}\,(x)}{\sqrt{k_x}}\right)^2 + \left(\frac{\mathsf{std}\,(y)}{\sqrt{k_y}}\right)^2}.$$

We can now use the same reasoning we used to test the hypothesis that a population had a particular, known mean. We have identified a number we can compute from

the two samples. We know how this number would vary under random choices of sample. If the value we observe is too many standard deviations away from the mean, the evidence is against our hypothesis. If we wanted to believe the hypothesis, we would be forced to believe that the sample is extremely strange. I have summarized this reasoning in box 9.2.

Procedure: 9.2 *Testing whether two populations have a given mean, based on a sample of each*

The initial hypothesis is that the populations have the same, unknown, mean. Write $\{x\}$ for the sample of the first population, $\{y\}$ for the sample of the second population, and k_x, k_y for the sample sizes.

- Compute the sample means for each population, $\mathsf{mean}\,(\{x\})$ and $\mathsf{mean}\,(\{y\})$.
- Estimate the standard error s_e for each population, using

$$s_{ex} = \frac{\mathsf{std}\,(x)}{\sqrt{k}}, \qquad s_{ey} = \frac{\mathsf{std}\,(y)}{\sqrt{k}}.$$

- Compute the standard error for the difference between the means,

$$s_{ed} = \sqrt{s_{ex}^2 + s_{ey}^2}.$$

- Compute the **test statistic** using

$$s = \frac{(\mathsf{mean}\,(\{x\}) - \mathsf{mean}\,(\{y\}))}{s_{ed}}.$$

- Compute the p-value, using

$$p = (1 - f) = (1 - \int_{-|s|}^{|s|} \exp\left(\frac{-u^2}{2}\right) du)$$

- The p-value summarizes the extent to which the data contradicts the hypothesis. A small p-value implies that, *if* the hypothesis is true, the sample is very unusual. The smaller the p-value, the more strongly the evidence contradicts the hypothesis.

It is common to think that a hypothesis can be rejected only if the p-value is less than 5% (or some number). You should not think this way; the p-value summarizes the extent to which the data contradicts the hypothesis, and your particular application logic affects how you interpret it.

Worked example 9.11 *Are US and Japanese cars different*

At http://www.itl.nist.gov/div898/handbook/eda/section3/eda3531.htm you can find a dataset, published by NIST, giving measurements of miles per gallon for Japanese and US cars. Assess the evidence these two populations have the same mean MPG.

Solution: There are 249 measurements for Japanese cars, and 79 for US cars. The mean for Japanese cars is 20.1446, and for US cars is 30.4810. The standard error for Japanese cars is 0.4065, and for US cars is 0.6872. The value of the test statistic is

$$\frac{(\mathsf{mean}\,(\{x\}) - \mathsf{mean}\,(\{y\}))}{s_{ed}} = 10.33/0.7984 = 12.94$$

and the p-value is the probability of encountering a standard normal random variable of this value or greater. This is so close to zero I had trouble getting sensible numbers out of MATLAB; the evidence very strongly rejects this hypothesis. A version of this example, using the more complex two-sample t-test, is worked in the NIST/SEMATECH e-Handbook of Statistical Methods, at http://www.itl.nist.gov/div898/handbook/, as of 2013.

9.2.4 Variants on the Basic Test

The basic recipe we described is: find a statistic that summarizes the relationship between the sample and the hypothesis; figure out the probability distribution of that statistic under random variation caused by sampling; and then determine how rare the sample would have to be to observe the value of the statistic that was actually observed. There are a wide range of different instances of the recipe. I will sketch some variants here, but be aware that this subject is very big indeed.

One-sided Tests

The test of procedure 9.1 is usually referred to as a **two-sided** test. This is because it computes

$$P(\{X > |s|\}) \cup P(\{X < -|s|\}).$$

This is the fraction of samples that would produce a value that is larger than $|s|$ or smaller than $-|s|$. In some cases, we can expect that only the larger (or smaller) value is of interest. This yields a **one-sided** test; the relevant expression is either

$$P(\{X > s\})$$

or

$$P(\{X < s\}).$$

Generally, it is more conservative to use a two-sided test, and one should do so unless there is a good reason not to. Very often, authors use one-sided tests because they result in smaller p-values, and small p-values are often a requirement for publication.

Correcting the Standard Deviation

When the sample is small, the sample standard deviation is a poor estimate of the population standard deviation. The value of the sample mean that we compute minimizes the sample standard deviation, which means that the estimate tends to be a little too small. In turn, the standard error is a little too small, and there is slightly more probability that the sample mean is far from the population mean than the normal model allows for. This can be corrected for. Instead of using the standard deviation of the sample to form the standard error, we use

$$\sqrt{\frac{\sum_i (x_i - \mathsf{mean}\,(\{x_i\}))^2}{k-1}}.$$

Z-Tests and T-Tests

When we have a large sample, the sample mean is a normal random variable whose mean is the population mean and whose standard deviation is the standard error. In this case, the test is known as a **z-test**. An important difficulty with this procedure is we assumed we know the standard error. Estimating standard error with $\mathsf{std}\,(x)/\sqrt{k}$ works fine for large k, but is less successful for small k. Usually, practical advice suggests that one should use a Z-test only if the sample has at least 30 elements.

If the sample is smaller, the distribution of the sample mean is not really normal. It turns out to have a form known as **Student's t-distribution**. This is a family of probability distributions. There are two parameters; the observed value s, and the number of degrees of freedom. The number of degrees of freedom is $k-1$ for our purposes. When the number of degrees of freedom is small, the t-distribution has rather heavier tails than the normal distribution, so the test takes into account that the standard error may be larger than we think (because the population standard deviation is larger than we estimated). When the number of degrees of freedom is large, the t-distribution is very similar to the normal distribution. One can get p-values from tables, or from pretty much any programming environment.

Dangerous Behavior

It is important to follow the logic of the method carefully. If you don't, you can fairly easily come to false conclusions. Removing data points and recomputing p-values is one way to have a serious problem. One context where this occurs in evaluating medical procedures. We would test the hypothesis that some sample mean of a treated population is the same as that of an untreated population. If this hypothesis fails, then the procedure did something. However, we might see outliers in the dataset. If we remove these outliers, then (of course) the p-value changes. This presents an temptation to remove outliers that shift the p-value in some desired direction. Of course, doing this consciously is fraud; but it's quite easy to simply fool yourself into removing data points whose absence is helpful.

Another way to fool yourself is to change hypotheses halfway through an experiment. Imagine we want to test the effect of a medical procedure. We decide to look at one particular sample mean, but halfway through collecting data it looks as though there won't be anything interesting to say about that mean. It's tempting to change measurements and focus on another statistic instead. If you do, the logic underlying the procedures we describe here fails. Essentially, you will bias the test to reject a hypothesis to a degree we can't go into. One solution is really strict protocols, where you describe everything you will do and everything you will test before doing the experiment and then don't vary from that plan. But such protocols can be expensive and clumsy in practice.

Yet another way to fool yourself is to test multiple hypotheses at the same time, and reject the one with the smallest p-value. The problem here is that the test isn't taking into account that you're testing multiple hypotheses. If you repeatedly test different hypotheses using the same dataset, the chances go up that the data you observed are inconsistent with a hypothesis that is true, purely as a result of sampling. Special procedures are required for this case.

Yet another way to fool yourself is to get confused about what a test means. The tests I have described are often referred to as tests of statistical significance. I don't find this label helpful, because there is an insidious suggestion that a statistically significant difference actually matters – i.e. is significant. What the procedures tell you is what fraction of random samples of data have the mean that you observe, if your hypothesis is true, and if you have collected the data correctly, tested correctly, and so on. The procedures don't tell you that you've done important, or even interesting, science.

9.3 χ^2 TESTS: IS DATA CONSISTENT WITH A MODEL?

Sometimes we have a model, and we would like to know whether the data is consistent with that model. For example, imagine we have a six-sided die. We throw it many times, and record which number comes up each time. We would like to know if the die is fair (i.e. is the data consistent with the model that the die is fair). It is highly unlikely that each face comes up the same number of times, even if the die is fair. Instead, there will be some variation in the frequencies observed; with what probability is that variation, or bigger, the result of chance effects?

As another example, we decide that the number of calls by a telemarketer in each hour is distributed with a Poisson distribution. We don't know the intensity. We could collect call data, and use maximum likelihood to determine the intensity. Once we have the best estimate of the intensity, we still want to know whether the model is consistent with the data.

In each case, the model predicts the frequencies of events. For the six-sided die case, the model tells us how often we expect to see each side. For the call case, the model predicts how often we would see no calls, one call, two calls, three calls, etc. in each hour. To tell whether the model fits the data, we need to compare the frequencies we observed with theoretical frequencies.

The appropriate test is a χ^2 (say "khi-squared") test. Assume we have a set of k disjoint events $\mathcal{E}_1, \ldots, \mathcal{E}_k$ which cover the space of outcomes (i.e. any outcome lies in one of these events). Assume we perform N experiments, and record the

number of times each event occurs. We have a hypothesis regarding the probability of events. We can take the probability of each event and multiply by k to get a frequency under that hypothesis. Now write $f_o(\mathcal{E}_i)$ for the observed frequency of event i; $f_t(\mathcal{E}_i)$ for the theoretical frequency of the event under the null hypothesis. We form the statistic

$$\sum_i \frac{(f_o(\mathcal{E}_i) - f_t(\mathcal{E}_i))^2}{f_t(\mathcal{E}_i)}$$

which compares the observed and actual frequency of events. It turns out that this statistic has a distribution very close to a known form, called the χ^2 distribution, as long as each count is 5 or more. The distribution has two parameters; the statistic, and the number of degrees of freedom.

The degrees of freedom refers to the dimension of the space of measurement values that you could have. We will need to fix some values. The number of values to fix has to do with the type of test you are doing. In the most common case, you want to inspect the counts in each of k bins to tell whether they are consistent with some distribution. We know the sum of counts is N. It turns out that we should compare what we observe with what we could have observed with the same N. In this case, the dimension of the space of measurement value is $k - 1$, because you have k numbers but they must add up to N.

Now assume we have to estimate p parameters for the null hypothesis. For example, rather than asking whether the data comes from a standard normal distribution, we might use the data to estimate the mean. We then test whether the data comes from a normal distribution with the estimated mean, but unit standard deviation. As another example, we could estimate both mean and standard deviation from the data. If we estimate p parameters from the data, then the number of degrees of freedom becomes $k - p - 1$ (because there are k counts, they must lead to p parameter values, and they must add to 1).

After this, things follow the usual recipe. We compute the statistic; we then look at tables, or use our programming environment, to find the probability that the statistic takes this value or greater under the null hypothesis. If this is small, then we reject the null hypothesis.

Worked example 9.12 χ^2 *test for dice*

I throw a die 100 times. I record the outcomes, in the table below. Is this a fair die?

face	count
1	46
2	13
3	12
4	11
5	9
6	9

Solution: The expected frequency is 100/6 for each face. The χ^2 statistic has the value 62.7, and there are 5 degrees of freedom. We get a p-value of about 3e-12. You would have to run this experiment 3e11 times to see a table as skewed as this once, by chance. It's quite unreasonable to believe the die is fair – or, at least, if you wanted to do so, you would have to believe you did a quite extraordinary unusual experiment.

Worked example 9.13 *Are mouse weights normally distributed?*

Assess the evidence against the hypothesis that the weights of all mice who ate chow are normally distributed, based on the dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml.

Solution: This example takes a little thought. The way to check whether a set of data is (roughly) normally distributed, is to break the values into intervals, and count how many data items fall into each interval. This gives the observed frequencies. You can then also compute theoretical frequencies for those intervals with a normal distribution (or simulate). Then you use a χ^2 test to tell whether the two are consistent. The choice of intervals matters. It is natural to have intervals that are some fraction of a standard deviation wide, with the mean at the center. You should have one running to infinity, and one to minus infinity. You'd like to have enough intervals so that you can tell any big difference from normal, but it's important to have at least five data items in each interval. There are 92 mice who make it to whenever Weight2 is evaluated (sacrifice, I think). The mean of Weight2 is 31.91 and the standard deviation is 6.72. I broke the data into 10 intervals at breakpoints $[-\infty, -1.2, -0.9, -0.6, -0.3, 0, 0.3, 0.6, 0.9, 1.2, \infty] * 6.72 + 31.91$. This gave me a count vector $[10, 9, 12, 9, 7, 11, 7, 9, 8, 10]$. I simulated 2000 draws from a normal distribution with the given mean and standard deviation and sorted them into these intervals, getting $[250, 129, 193, 191, 255, 240, 192, 192, 137, 221]$ (if you are less idle, you'll evaluate the integrals, but this shouldn't make much difference). I found a statistic with value 5.6338. Using 7 degrees of freedom (10 counts, but there are two parameters estimated), I found a p-value of 0.5830979, meaning there is no reason to reject the idea that the weights are normally distributed.

Worked example 9.14 *Is swearing Poisson?*

A famously sweary politician gives a talk. You listen to the talk, and for each of 30 intervals 1 minute long, you record the number of swearwords. You record this as a histogram (i.e. you count the number of intervals with zero swear words, with one, etc.), obtaining the table below.

no. of swear words	no. of intervals
0	13
1	9
2	8
3	5
4	5

The null hypothesis is that the politician's swearing is Poisson distributed, with intensity (λ) one. Can you reject this null hypothesis?

Solution: If the null hypothesis is true, then the probability of getting n swear words in a fixed length interval would be $\frac{\lambda^n e^{-\lambda}}{n!}$. There are 10 intervals, so the theoretical frequencies are 10 times the following probabilities

number of swear words	probability
0	0.368
1	0.368
2	0.184
3	0.061
4	0.015

so the χ^2 statistic takes the value 243.1 and there are 4 degrees of freedom. The significance is indistinguishable from zero by my programming environment, so you can firmly reject the null hypothesis. Of course, it may just be that the intensity is wrong (exercises).

Worked example 9.15 *Are goals independent of gender?*

Assess the evidence that student goals are independent of student gender in the dataset of Chase and Dunner, which you can find at http://lib.stat.cmu.edu/DASL/Datafiles/PopularKids.html.

Solution: This is an example of a common use of the χ^2 test. The table below shows the count of students in the study by gender and goal. I have inserted row and column totals for convenience. In total, there were 478 students.

	boy	girl	total
Grades	117	130	247
Popular	50	91	141
Sports	60	30	90
total	227	251	478

We will test whether the counts observed are different from those predicted if gender and goal are independent. If they are independent, then $P(\text{boy}) = 227/478 = 0.47$, and $P(\text{Grades}) = 247/478 = 0.52$ (and so on). This means that we can produce a table of theoretical counts under the model (below).

	boy	girl
Grades	117.29916	129.70084
Popular	66.96025	74.03975
Sports	42.74059	47.25941

There are 6 cells in our table. One degree of freedom is used up by requiring that there are 478 students. Two further degrees of freedom are used up by insisting that the Grades/Popular/Sports counts yield the distribution we observed. One further degree of freedom is used up by insisting that the gender counts yield the distribution we observe. This means that there are a total of two degrees of freedom. We compute a χ^2 value of 21.46. The p-value is 2e-5. In turn, if the two factors were independent, you'd see counts like these by chance about twice in a hundred thousand experiments. It's very hard to believe they are independent.

9.4 USING SIMULATION TO CONSTRUCT INTERVALS

The analysis of some problems is difficult. But the idea of a confidence interval is naturally associated with repeated experiments. We can use simulations to estimate confidence intervals when analysis is difficult. Generally, we see the dataset as one of many possible dataset we could have seen. We must then assess what our estimate would be like for the other possible datasets. But what are those possible datasets to be?

When the data is explained by a parametric probability model, we can use that model to produce other possible datasets. If we compute a maximum likelihood estimate $\hat{\theta}$ of the parameters of the model, we can draw IID samples *from the model*. We then look at the estimate from that new dataset; the spread of the estimates yields our confidence interval.

When we have no applicable probability model of the dataset, we can resample the dataset to simulate the effects of sampling error. This strategy allows us to build confidence intervals for properties like medians.

9.4.1 Constructing Confidence Intervals for Parametric Models

Assume we have a dataset $\mathcal{D} = \{\mathbf{x}\}$, and a probability model we believe applies to that dataset. We know how to estimate appropriate parameter values by maximizing the likelihood function $L(\theta)$. But we do not yet have a way to think about how accurately the data determines the best choice of parameter value. For example, if we change the dataset slightly, will the estimated parameter change a lot? or a little?

A $(1 - 2\alpha)$ confidence interval for a parameter is an interval $[c_\alpha, c_{(1-\alpha)}]$. We construct the interval such that, if we were to perform a very large number of repetitions of our original experiment then estimate a parameter value for each, c_α would be the α quantile and $c_{(1-\alpha)}$ would be the $(1-\alpha)$ quantile of those parameter values. We interpret this to mean that, with confidence $(1 - 2\alpha)$, the correct value of our parameter lies in this interval. This definition isn't really watertight. How do we perform a very large number of repetitions? If we don't, how can we tell how good the confidence interval is? Nonetheless, we can construct intervals that illustrate how sensitive our original inference is to the data that we have.

There are a variety of methods to construct confidence intervals. We will focus on simulation. The algorithm should feel so natural to you that you may already have guessed what to do. First, we compute the maximum likelihood estimate of the parameters, $\hat{\theta}$. We assume this estimate is right, but need to see how our estimates of $\hat{\theta}$ would vary with different collections of data from our model with that parameter value. So we compute a collection of simulated datasets, $\mathcal{D}_i$, each the same size as the original dataset. We obtain these by simulating $P(d|\hat{\theta})$. Next, we compute a maximum likelihood estimate for each of these simulated datasets, $\hat{\theta}_i$. Finally, we compute the relevant percentiles of these datasets. The result is our interval.

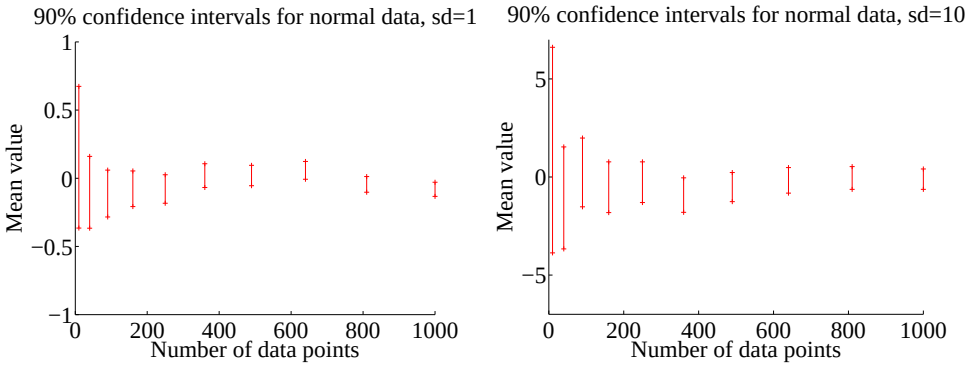

FIGURE 9.2: *Confidence intervals computed for simulated normal data; details in the text.*

Procedure: 9.3 *Estimating Confidence Intervals for Maximum Likelihood Estimates using Simulation*

Assume we have a dataset $\mathcal{D} = \{x\}$ of N items. We have a parametric model of this data $p(x|\theta)$, and write the likelihood $\mathcal{L}(\theta; \mathcal{D}) = P(\mathcal{D}|\theta)$. We construct $(1 - 2\alpha)$ confidence intervals using the following steps.

1. Compute the maximum likelihood estimate of the parameter, $\hat{\theta} = \underset{\theta}{\text{argmax}}\ \mathcal{L}(\theta; \mathcal{D})$.
2. Construct R simulated datasets $\mathcal{D}_i$, each consisting of N IID samples drawn from $p(x|\hat{\theta})$.
3. For each such dataset, compute $\hat{\theta}_i = \underset{\theta}{\text{argmax}}\ \mathcal{L}(\theta; \mathcal{D})$.
4. Obtain $c_\alpha(\hat{\theta}_i)$, the α'th quantile of the collection $\hat{\theta}_i$ and $c_{(1-\alpha)}(\hat{\theta}_i)$, the $1 - \alpha$'th quantile of the collection $\hat{\theta}_i$.

The confidence interval is $[c_\alpha, c_{(1-\alpha)}]$.

Figure 9.2 shows an example. In this case, I worked with simulated data from a normal distribution. In each case, the normal distribution had mean 0, but there are two different standard deviations (1 and 10). I simulated 10 different datasets from each of these distributions, containing $10, 40, 90, \ldots, 810, 1000$ data items. For each, I computed the maximum likelihood estimate of the mean. This isn't zero, *even though the data was drawn from a zero-mean distribution*, because the dataset is finite. I then estimated the confidence intervals for each using 10, 000 simulated datasets of the same size. I show 95% confidence intervals for the two cases, plotted against the size of the dataset used for the estimate. Notice that these intervals

aren't symmetric about zero, because the maximum likelihood estimate isn't zero. They shrink as the dataset grows, but slowly. They are bigger when the standard deviation is bigger. It should seem reasonable that you can't expect an accurate estimate of the mean of a normal distribution with large standard deviation using only a few data points. One can demonstrate using statistical theory that would take us rather out of our way that the size of these intervals should behave like

$$\frac{\sigma}{\sqrt{N}}.$$

Worked example 9.16 *Confidence Intervals by Simulation - II*

Construct a 90% confidence interval for the intensity estimate for the data of example 8.5 for the cases of 10 observations, 20 observations, and all 30 observations.

Solution: Recall from that example the maximum likelihood estimates of the intensity are 7/10, 22/20, and 29/30 in the three cases. I used the Matlab function `poissrnd` to get 10000 replicates of a dataset of 10 (resp. 20, 30) items from a Poisson distribution with the relevant intensities. I then used `prctile` to get the 5% and 95% percentiles, yielding the intervals

$[0.3, 1.2]$	for 10 observations
$[0.75, 1.5]$	for 20 observations
$[0.6667, 1.2667]$	for 30 observations

Notice how having more observations makes the confidence interval smaller.

9.4.2 Standard Error Estimates from Simulation

??

We were able to produce convenient and useful estimates of standard error for sample means. But what happens if we want to reason about, say, the median of a population? It turns out that estimating the standard error of a median is difficult mathematically, and estimating the standard error of other interesting statistics, can be difficult, too. This is an important problem, because our methods for building confidence intervals and for testing hypotheses rely on being able to construct standard error estimates. It turns out that quite simple simulation methods give very good estimates of standard error.

The distribution of median values for different samples of a population looks normal by simple tests. For Figure 9.3, I assumed that all 253 weight measurements represented the entire population, then simulated what would happen for different random samples (with replacement) of different sizes. Figure 9.3 suggests that the sample median behaves quite like the sample mean as the random sample changes. Different samples have different medians, but the distribution of values looks fairly normal. When there are more elements in the sample, the standard deviation of median values is smaller. But we have no method to compute this standard

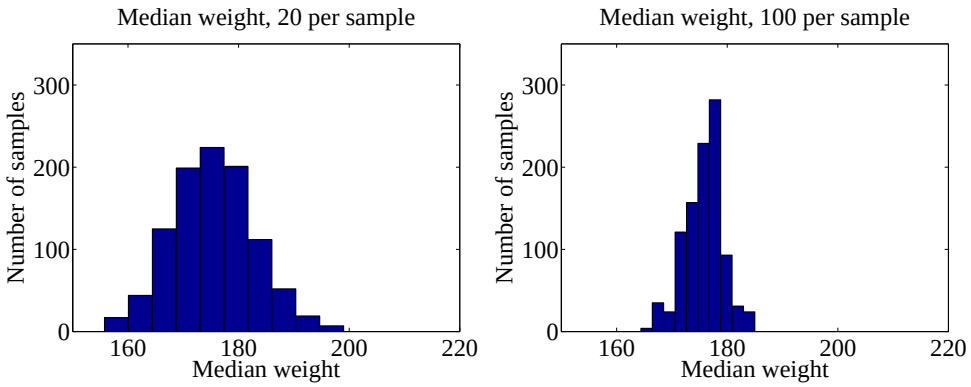

FIGURE 9.3: *I took the weights dataset used all 253 measurements to represent a population. Rather than compute the median of the whole population, I chose to compute the median of a randomly chosen sample. The figures show a histogram of 1000 different values of the median, computed for 1000 different samples (of size 20 on the* **left***, and of size 100 on the* **right***). Notice that (a) there is a moderate amount of variation in the median of the sample; (b) these histograms look normal, and appear to have about the same mean; (c) increasing the size of the sample has reduced the spread of the histogram.*

deviation. The method I used here doesn't apply in practice, because we don't usually have the whole population.

There is a method, known as the **bootstrap**, which gives a very good estimate of the standard error of any statistic. Assume we wish to estimate the standard error of a statistic $S(\{\mathbf{x}\})$, which is a function of our dataset $\{\mathbf{x}\}$ of N data items. We compute r **bootstrap replicates** of this sample. Each replicate is obtained by sampling the dataset uniformly, and with replacement. One helpful way to think of this is that we are modelling our dataset as a sample of a probability distribution. This distribution, sometimes known as the **empirical distribution**, has probability $1/N$ at each of the data items we see, and zero elsewhere. Now to obtain replicates, we simply draw new sets of IID samples from this probability distribution.

Notice that the bootstrap replicates are *not* a random permutation of the dataset; instead, we select one data item fairly and at random from the whole dataset N times. This means we expect a particular bootstrap replicate will have multiple copies of some data items, and no copies of others.

We write $\{\mathbf{x}\}_i$ for the i'th bootstrap replicate of the dataset. We now compute

$$\overline{S} = \frac{\sum_i S(\{\mathbf{x}\}_i)}{r}$$

and the standard error estimate for S is given by:

$$\mathsf{stderr}\left(\{S\}\right) = \sqrt{\frac{\sum_i \left[S(\{\mathbf{x}\}_i) - \overline{S}\right]}{r-1}}$$

Procedure: 9.4 *The bootstrap*

Estimate the standard error for a statistic S evaluated on a dataset of N items $\{\mathbf{x}\}$.

1. Compute r bootstrap replicates of the dataset. Write the i'th replicate $\{\mathbf{x}\}_i$ Obtain each by:
   - **(a)** Building a uniform probability distribution on the numbers $1 \ldots N$.
   - **(b)** Drawing N independent samples from this distribution. Write $s(i)$ for the i'th such sample.
   - **(c)** Building a new dataset $\left\{x_{s(1)}, \ldots, x_{s(N)}\right\}$.
2. For each replicate, compute $\sum_i S(\{\mathbf{x}\}_i)$.
3. Compute
$$\overline{S} = \frac{\sum_i S(\{\mathbf{x}\}_i)}{r}$$
4. The standard error estimate for S is given by:
$$\mathsf{stderr}\left(\{S\}\right) = \sqrt{\frac{\sum_i \left[S(\{\mathbf{x}\}_i) - \overline{S}\right]}{r-1}}$$

Worked example 9.17 *The bootstrap standard error of the median*

You can find a dataset giving the salaries of CEO's at small firms in 1993 at http://lib.stat.cmu.edu/DASL/Datafiles/ceodat.html. Construct a 90% confidence interval for the median salary.

Solution: Salaries are in thousands of dollars, and one salary isn't given (we omit this value in what follows). Figure 9.4 shows a histogram of the salaries; notice there are some values that look like outliers. This justifies using a median. The median of the dataset is 350 (i.e. \$ 350, 000 — this is 1993 data!). I constructed 10, 000 bootstrap replicates. Figure 9.4 shows a histogram of the medians of the replicates. I used the matlab `prctile` function to extract the 5% and 95% percentiles of these medians, yielding the interval between 298 and 390. This means that we can expect that, for 90% of samples of CEO salaries for small companies, the median salar will be in the given range.

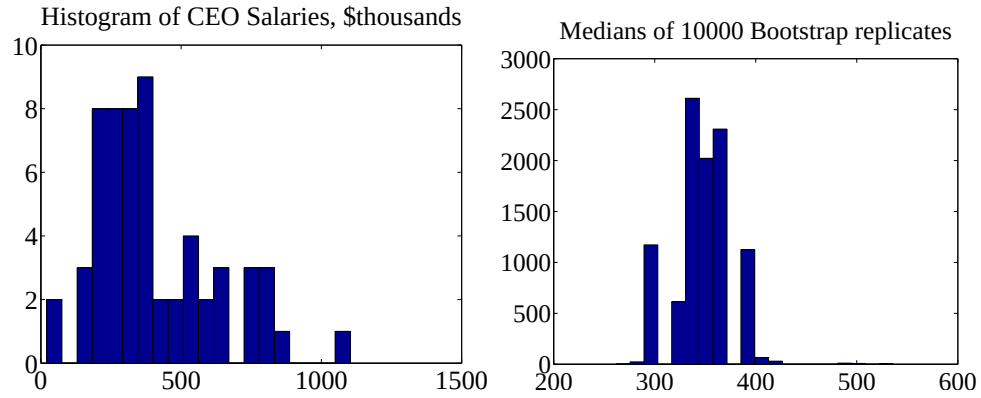

FIGURE 9.4: *On the* **left**, *a histogram of salaries for CEO's of small companies in 1993, from the dataset of* http://lib.stat.cmu.edu/DASL/Datafiles/ceodat.html. *On the* **right**, *a histogram of the medians of 10, 000 bootstrap replicates of this data. This simulates the effect of sampling variation on the median; see worked example 9.17.*

9.5 YOU SHOULD

9.5.1 remember these definitions:

9.5.2 remember these terms:

9.5.3 remember these facts:

9.5.4 remember these procedures:

9.5.5 be able to:

- compute a confidence interval for an estimate of a sample mean;
- compute a confidence interval from a posterior;
- assess the evidence against a hypothesis of a sample mean;
- assess the evidence that two populations have different sample means;
- test a hypothesis using a χ^2 test.

PROBLEMS

Confidence Intervals for Population Means

9.1. The Weight of Mice You wish to estimate the average weight of a mouse. You obtain 10 mice, sampled uniformly at random and with replacement from the mouse population. Their weights are $21, 23, 27, 19, 17, 18, 20, 15, 17, 22$ grams respectively.
(a) Give a 68% confidence interval for the weight of a mouse, from this data.
(b) Give a 99% confidence interval for the weight of a mouse, from this data.

9.2. The Weight of Rats You wish to estimate the average weight of a pet rat. You obtain 40 rats (easily and cheaply done; keep them, because they're excellent pets), sampled uniformly at random and with replacement from the pet rat population. The mean weight is 340 grams, with a standard deviation of 75 grams.
(a) Give a 68% confidence interval for the weight of a pet rat, from this data.
(b) Give a 99% confidence interval for the weight of a pet rat, from this data.

Bayesian Confidence Intervals

9.3. In worked example 9.13, we found no reason to reject the idea that mouse weights are normally distributed, using the dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml.
(a) Construct a 75% confidence interval for the weight of a mouse that eats chow, using this data. You should get this interval using reasoning about standard errors – this isn't Bayesian.
(b) Now construct a Bayesian 75% confidence interval for the weight of a mouse that eats chow, using this data. You should assume that the prior on the weight of such a mouse is normal, with mean 32 and standard deviation 10. Recall from section 8.2.2 that a normal prior and a normal likelihood lead to normal posterior.
(c) Compare the intervals constructed above with a Bayesian 75% confidence interval for the weight of a mouse that eats chow, using this data and assuming the prior on the weight of such a mouse is normal, with mean 32 and standard deviation 1. What does this tell you about the importance of the prior?

Hypothesis Testing

9.4. Yet more Mouse-weighing I claim the average weight of a mouse is 25 grams. You decide to evaluate the evidence in support of this claim. You obtain 10 mice, sampled uniformly at random and with replacement from the mouse population. Their weights are $21, 23, 27, 19, 17, 18, 20, 15, 17, 22$ grams respectively. Does the evidence support my claim? to what extent? Why?

9.5. How big are Parktown Prawns? The Parktown prawn is an impressively repellent large insect, common in Johannesburg (look them up on the Web). I claim that their average length is 10cm. You collect 100 Parktown prawns (this will take about 10 mins, in the right places in Johannesburg; more difficult from here). The mean length of these prawns is 7cm. The standard deviation is 1cm. Assess the evidence against my claim.

9.6. Two Populations of Rats Zucker rats are specially bred to have curious weight properties, related to their genetics (look them up on the Web). You

measure 30 lean Zucker rats, obtaining an average weight of 500 grams with a standard deviation of 50 grams. You measure 20 fatty Zucker rats, obtaining an average weight of 1000 grams with a standard deviation of 100 grams. Assess the evidence against the claim that these populations have the same weight.

9.7. Male and Female pet Rats You measure 35 female pet rats, obtaining an average weight of 300 grams with a standard deviation of 30 grams. You measure 30 male pet rats, obtaining an average weight of 400 grams with a standard deviation of 100 grams. Assess the evidence against the claim that these populations have the same weight.

9.8. Lean and Fatty Zucker Rats Zucker rats are specially bred to have curious weight properties, related to their genetics (look them up on the Web). You measure 30 lean Zucker rats, obtaining an average weight of 500 grams with a standard deviation of 50 grams. You measure 35 fatty Zucker rats, obtaining an average weight of 1000 grams with a standard deviation of 100 grams. In steps, you will assess the evidence against the claim that a fatty Zucker rat has exactly twice the weight of a lean Zucker rat. You know that the product of a normal random variable and a constant is a normal random variable. You should assume (and accept, because I won't prove it) that the sum of two normal random variables is a normal random variable.

(a) Write $L^{(k)}$ for the random variable obtained by drawing a uniform sample of k lean rats and averaging their weights. You can assume that k is large enough that this is normal.

- What is $\mathbb{E}\left[L^{(k)}\right]$? (write an expression, no need to prove anything)
- What is $\mathsf{std}\left(L^{(k)}\right)$? (write an expression, no need to prove anything)

(b) Now write $F^{(s)}$ for the random variable obtained by drawing a uniform sample of s fatty rats and averaging their weights. You can assume that s is large enough that this is normal.

- What is $\mathbb{E}\left[F^{(s)}\right]$? (write an expression, no need to prove anything)
- What is $\mathsf{std}\left(F^{(s)}\right)$? (write an expression, no need to prove anything)

(c) Write $\mathsf{popmean}\left(\{L\}\right)$ for the population mean weight of lean rats, and $\mathsf{popmean}\left(\{F\}\right)$ for the population mean weight of fatty rats. Assume that $2\mathsf{popmean}\left(\{L\}\right) = \mathsf{popmean}\left(\{F\}\right)$.

- In this case, what is $\mathbb{E}\left[F^{(s)} - 2L^{(k)}\right]$?
- In this case, what is $\mathsf{std}\left(F^{(s)} - 2L^{(k)}\right)$?
- Your expression for $\mathsf{std}\left(F^{(s)} - 2L^{(k)}\right)$ will have contained terms in the population standard deviation of F and L. What is the standard error of $F^{(s)} - 2L^{(k)}$?

(d) Now assess the evidence against the hypothesis that a fatty Zucker rat weighs exactly twice as much as a lean Zucker rat.

9.9. Are boys and girls equiprobable? In Carcelle-le-Grignon at the end of the 18'th century, there were 2,009 births. There were 983 boys and 1026 girls. You can regard this as a fair random sample (with replacement, though try

not to think too hard about what that means) of births. Assess the evidence against the hypothesis that a boy is born with probability exactly 0.5.

9.10. In 1998, the average height of an adult male in South Africa was estimated to be 169cm. Assume that this estimate is exact; assume also that the population standard deviation is 10cm. What fraction of samples consisting of 50 adult males from South Africa (selected uniformly at random, and with replacement) will have average height greater than 200cm?

9.11. Assume the average weight of an adult male short-hair house cat is 5 kg, and the standard deviation is 0.7 kg (these numbers are reasonable, but there's quite a lively fight between cat fanciers about the true numbers).

(a) What fraction of samples consisting of 30 adult male short-hair house cats (selected uniformly at random, and with replacement) will have average weight less than 4kg?

(b) What fraction of samples consisting of 300 adult male short-hair house cats (selected uniformly at random, and with replacement) will have average weight less than 4kg?

(c) Why are these numbers different?

Chi-squared Tests

9.12. You can find a dataset of the passenger list for the Titanic disaster at http://www.statsci.org/data/general/titanic.html.

(a) Assess the evidence that survival is independent of passenger ticket class.

(b) Assess the evidence that survival is independent of passenger gender.

9.13. You can find a dataset giving income data for US citizens at the UC Irvine Machine Learning data archive, at http://archive.ics.uci.edu/ml/datasets/Adult. Each item consists of a set of numeric and categorical features describing a person, together with whether their annual income is larger than or smaller than 50K$.

(a) Assess the evidence that income category is independent of gender.

(b) Assess the evidence that income category is independent of education level.

9.14. Assess the evidence that the swearing behavior of the politician of worked example 9.14 follows a Poisson distribution. *Hint:* Once you've estimated the intensity, the rest is like that example; be careful about the number of degrees of freedom.

PROGRAMMING EXERCISES

Simulation Based Confidence Intervals

9.15. In the mouse dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml, there are 92 mice that ate a chow diet ("chow" in the relevant column) and where the weight at sacrifice is known (the value of Weight2).

(a) Draw a sample of 30 chow eating mice uniformly at random, and compute the mean weight of these mice. Now use the simulation method of section 9.4.1 to estimate a centered 75% confidence interval for the mean weight of a mouse eating a chow diet, estimated from a sample of 30 mice.

(b) Draw 1000 samples of 30 chow eating mice uniformly at random, and use these samples to estimate a centered 75% confidence interval for the mean

weight of a mouse eating a chow diet, estimated from a sample of 30 mice. How does this estimate compare to the previous estimate?

9.16. In the mouse dataset at http://cgd.jax.org/datasets/phenotype/SvensonDO.shtml, of the mice where the weight at sacrifice is known (the value of Weight2), 92 ate a chow diet ("chow" in the relevant column) and 48 at a high fat diet ("HF" in the relevant column). Use the bootstrap algorithm of section **??** to estimate a 95% confidence interval for the median weight of each population. Use this information to assess the hypothesis that the median weight of these populations is different.

Bayesian Confidence Intervals

9.17. Example 8.5 gives data on swearing by a politician (which I've reproduced below, for your convenience).

no. of swear words	no. of intervals
0	5
1	2
2	2
3	1
4	0

and for the following 20 intervals, you see

no. of swear words	no. of intervals
0	9
1	5
2	3
3	2
4	1

Worked example 8.12 shows how to use a conjugate prior gamma distribution to obtain a gamma posterior. Write a program to identify a centered, 90% bayesian confidence interval for the intensity of the politicians swearing for different values of the prior parameters. How do different choices of these parameters affect your interval?

PART FOUR

TOOLS

CHAPTER 10

Extracting Important Relationships in High Dimensions

Chapter 3 described methods to explore the relationship between two elements in a dataset. We could extract a pair of elements and construct various plots. For vector data, we could also compute the correlation between different pairs of elements. But if each data item is d-dimensional, there could be a lot of pairs to deal with.

We will think of our dataset as a collection of d dimensional vectors. It turns out that there are easy generalizations of our summaries. However, is hard to plot d-dimensional vectors. We need to find some way to make them fit on a 2-dimensional plot. Some simple methods can offer insights, but to really get what is going on we need methods that can represent all relationships in a dataset in one go.

These methods visualize the dataset as a "blob" in a d-dimensional space. Many such blobs are flattened in some directions, because components of the data are strongly correlated. Finding the directions in which the blobs are flat yields methods to compute lower dimensional representations of the dataset.

10.1 SUMMARIES AND SIMPLE PLOTS

In this chapter, we assume that our data items are vectors. This means that we can add and subtract values and multiply values by a scalar without any distress. This is an important assumption, but it doesn't necessarily mean that data is continuous (for example, you can meaningfully add the number of children in one family to the number of children in another family). It does rule out a lot of discrete data. For example, you can't add "sports" to "grades" and expect a sensible answer.

When we plotted histograms, we saw that mean and variance were a very helpful description of data that had a unimodal histogram. If the histogram had more than one mode, one needed to be somewhat careful to interpret the mean and variance; in the pizza example, we plotted diameters for different manufacturers to try and see the data as a collection of unimodal histograms. In higher dimensions, the analogue of a unimodal histogram is a "blob" — a group of data points that clusters nicely together and should be understood together.

You might not believe that "blob" is a technical term, but it's quite widely used. This is because it is relatively easy to understand a single blob of data. There are good summary representations (mean and covariance, which I describe below). If a dataset forms multiple blobs, we can usually coerce it into a representation as a collection of blobs (using the methods of chapter 12). But many datasets really are single blobs, and we concentrate on such data here. There are quite useful tricks for understanding blobs of low dimension by plotting them, which I describe below.

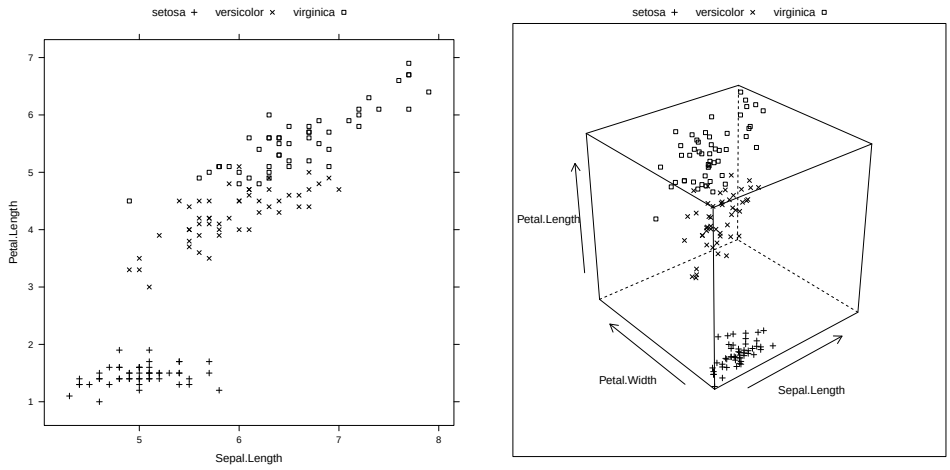

FIGURE 10.1: **Left:** *a 2D scatterplot for the famous Iris data. I have chosen two variables from the four, and have plotted each species with a different marker.* **Right:** *a 3D scatterplot for the same data. You can see from the plots that the species cluster quite tightly, and are different from one another. If you compare the two plots, you can see how suppressing a variable leads to a loss of structure. Notice that, on the left, some 'x's lie on top of boxes; you can see that this is an effect of projection by looking at the 3D picture (for each of these data points, the petal widths are quite different). You should worry that leaving out the last variable might have suppressed something important like this.*

To understand a high dimensional blob, we will need to think about the coordinate transformations that places it into a particularly convenient form.

Notation: Our data items are vectors, and we write a vector as $\mathbf{x}$. The data items are d-dimensional, and there are N of them. The entire data set is $\{\mathbf{x}\}$. When we need to refer to the i'th data item, we write $\mathbf{x}_i$. We write $\{\mathbf{x}_i\}$ for a new dataset made up of N items, where the i'th item is $\mathbf{x}_i$. If we need to refer to the j'th component of a vector $\mathbf{x}_i$, we will write $x_i^{(j)}$ (notice this isn't in bold, because it is a component not a vector, and the j is in parentheses because it isn't a power). Vectors are always column vectors.

10.1.1 The Mean

For one-dimensional data, we wrote

$$\text{mean}\left(\{x\}\right) = \frac{\sum_i x_i}{N}.$$

This expression is meaningful for vectors, too, because we can add vectors and divide by scalars. We write

$$\text{mean}\left(\{\mathbf{x}\}\right) = \frac{\sum_i \mathbf{x}_i}{N}$$

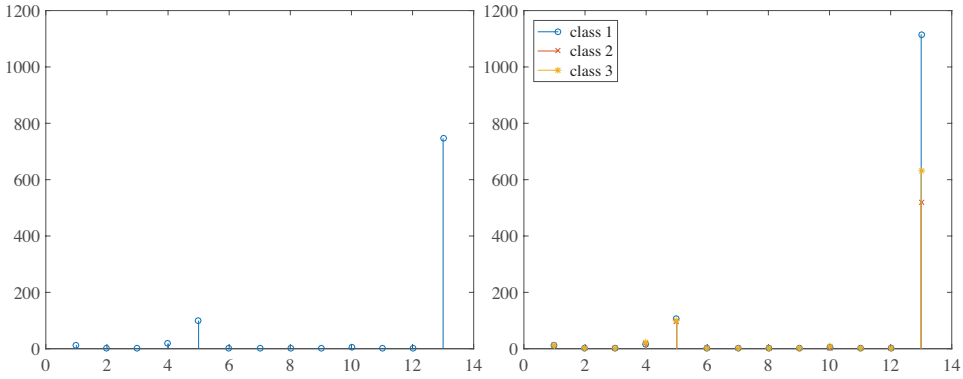

FIGURE 10.2: *On the* **left**, *a stem plot of the mean of all data items in the wine dataset, from* http://archive.ics.uci.edu/ml/datasets/Wine. *On the* **right**, *I have overlaid stem plots of each class mean from the wine dataset, from* http://archive.ics.uci.edu/ml/datasets/Wine, *so that you can see the differences between class means.*

and call this the mean of the data. Notice that each component of $\text{mean}(\{\mathbf{x}\})$ is the mean of that component of the data. There is not an easy analogue of the median, however (how do you order high dimensional data?) and this is a nuisance. Notice that, just as for the one-dimensional mean, we have

$$\text{mean}(\{\mathbf{x} - \text{mean}(\{\mathbf{x}\})\}) = 0$$

(i.e. if you subtract the mean from a data set, the resulting data set has zero mean).

10.1.2 Stem Plots and Scatterplot Matrices

Plotting high dimensional data is tricky. If there are relatively few dimensions, you could just choose two (or three) of them and produce a 2D (or 3D) scatterplot. Figure 10.1 shows such a scatterplot, for data that was originally four dimensional. This dataset is a famous dataset to do with the botanical classification of irises. I found a copy at the UC Irvine repository of datasets that are important in machine learning. You can find the repository at http://archive.ics.uci.edu/ml/index.html.

Another simple but useful plotting mechanism is the stem plot. This is can be a useful way to plot a few high dimensional data points. One plots each component of the vector as a vertical line, typically with a circle on the end (easier seen than said; look at figure 10.2). The dataset I used for this is the wine dataset, from the UC Irvine machine learning data repository. You can find this dataset at http://archive.ics.uci.edu/ml/datasets/Wine. For each of three types of wine, the data records the values of 13 different attributes. In the figure, I show the overall mean of the dataset, and also the mean of each type of wine (also known as the class means, or class conditional means). A natural way to compare class means is to plot them on top of one another in a stem plot (figure 10.2).

Another strategy that is very useful when there aren't too many dimensions is to use a scatterplot matrix. To build one, you lay out scatterplots for each pair

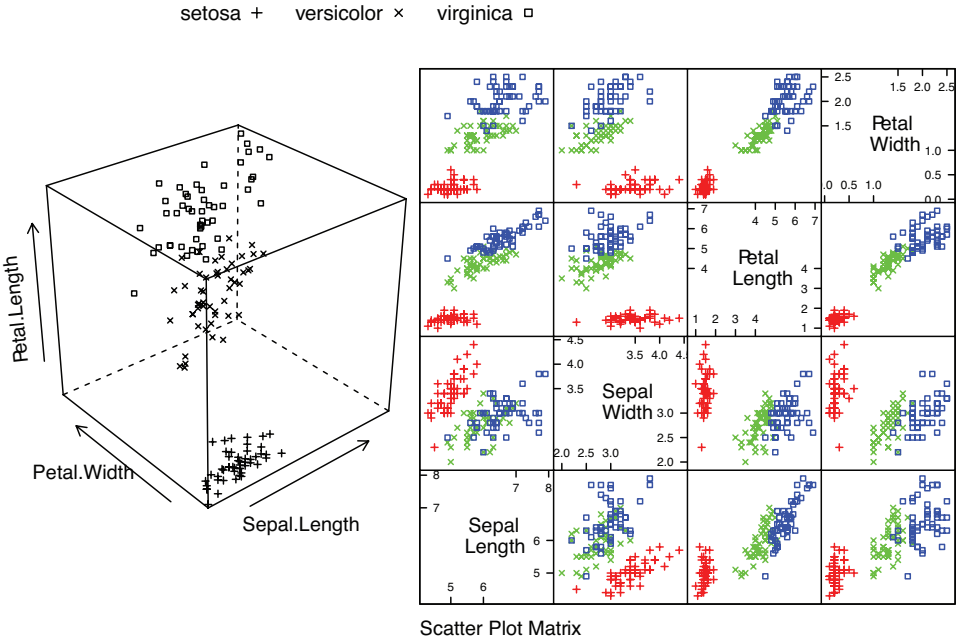

FIGURE 10.3: **Left:** *the 3D scatterplot of the iris data of Figure 10.1, for comparison.* **Right:** *a scatterplot matrix for the Iris data. There are four variables, measured for each of three species of iris. I have plotted each species with a different marker. You can see from the plot that the species cluster quite tightly, and are different from one another.*

of variables in a matrix. On the diagonal, you name the variable that is the vertical axis for each plot in the row, and the horizontal axis in the column. This sounds more complicated than it is; look at the example of figure 10.3, which shows both a 3D scatter plot and a scatterplot matrix for the same dataset. This is the famous Iris dataset, collected by Edgar Anderson in 1936, and made popular amongst statisticians by Ronald Fisher in that year.

Figure 10.4 shows a scatter plot matrix for four of the variables in the height weight dataset of http://www2.stetson.edu/~jrasp/data.htm; look for bodyfat.xls at that URL). This is originally a 16-dimensional dataset, but a 16 by 16 scatterplot matrix is squashed and hard to interpret. For figure 10.4, you can see that weight and adiposity appear to show quite strong correlations, but weight and age are pretty weakly correlated. Height and age seem to have a low correlation. It is also easy to visualize unusual data points. Usually one has an interactive process to do so — you can move a "brush" over the plot to change the color of data points under the brush.

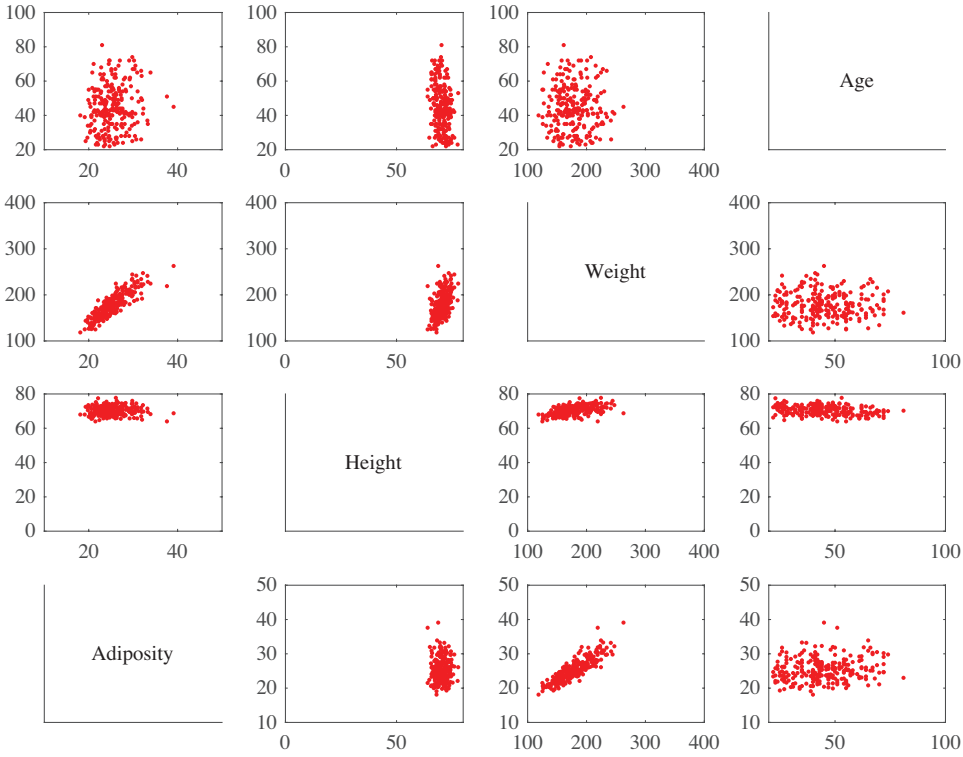

FIGURE 10.4: *This is a scatterplot matrix for four of the variables in the height weight dataset of http://www2.stetson.edu/~jrasp/data.htm. Each plot is a scatterplot of a pair of variables. The name of the variable for the horizontal axis is obtained by running your eye down the column; for the vertical axis, along the row. Although this plot is redundant (half of the plots are just flipped versions of the other half), that redundancy makes it easier to follow points by eye. You can look at a column, move down to a row, move across to a column, etc. Notice how you can spot correlations between variables and outliers (the arrows).*

10.1.3 Covariance

Variance, standard deviation and correlation can each be seen as an instance of a more general operation on data. Recall that I described correlation by extracting two components from each vector of a dataset of vectors. This gave me two datasets of N items; write $\{x\}$ for one and $\{y\}$ for the other. The i'th element of $\{x\}$ corresponds to the i'th element of $\{y\}$. This is because the i'th element of $\{x\}$ is one component of some bigger vector $\mathbf{x}_i$ and the i'th element of $\{y\}$ is another component of this vector. We can define the covariance of $\{x\}$ and $\{y\}$.

Definition: 10.1 *Covariance*

Assume we have two sets of N data items, $\{x\}$ and $\{y\}$. We compute the covariance by

$$\mathsf{cov}\left(\{x\},\{y\}\right) = \frac{\sum_i (x_i - \mathsf{mean}\left(\{x\}\right))(y_i - \mathsf{mean}\left(\{y\}\right))}{N}$$

Covariance measures the tendency of corresponding elements of $\{x\}$ and of $\{y\}$ to be larger than (resp. smaller than) the mean. Just like mean, standard deviation and variance, covariance can refer either to a property of a dataset (as in the definition here) or a particular expectation (as in chapter 5). The correspondence is defined by the order of elements in the data set, so that x_1 corresponds to y_1, x_2 corresponds to y_2, and so on. If $\{x\}$ tends to be larger (resp. smaller) than its mean for data points where $\{y\}$ is also larger (resp. smaller) than its mean, then the covariance should be positive. If $\{x\}$ tends to be larger (resp. smaller) than its mean for data points where $\{y\}$ is smaller (resp. larger) than its mean, then the covariance should be negative.

From this description, it should be clear we have seen examples of covariance already. Notice that

$$\mathsf{std}\left(x\right)^2 = \mathsf{var}\left(\{x\}\right) = \mathsf{cov}\left(\{x\},\{x\}\right)$$

which you can prove by substituting the expressions. Recall that variance measures the tendency of a dataset to be different from the mean, so the covariance of a dataset with itself is a measure of its tendency not to be constant. More important is the relationship between covariance and correlation, in the box below.

Remember this:

$$\mathsf{corr}\left(\{(x,y)\}\right) = \frac{\mathsf{cov}\left(\{x\},\{y\}\right)}{\sqrt{\mathsf{cov}\left(\{x\},\{x\}\right)}\sqrt{\mathsf{cov}\left(\{y\},\{y\}\right)}}.$$

This is occasionally a useful way to think about correlation. It says that the correlation measures the tendency of $\{x\}$ and $\{y\}$ to be larger (resp. smaller) than their means for the same data points, *compared to* how much they change on their own.

10.1.4 The Covariance Matrix

Working with covariance (rather than correlation) allows us to unify some ideas. In particular, for data items which are d dimensional vectors, it is straightforward to compute a single matrix that captures all covariances between all pairs of components — this is the covariance matrix.

Definition: 10.2 *Covariance Matrix*

The covariance matrix is:

$$\mathsf{Covmat}\left(\{\mathbf{x}\}\right) = \frac{\sum_i (\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))^T}{N}$$

Notice that it is quite usual to write a covariance matrix as Σ, and we will follow this convention.

Covariance matrices are often written as Σ, whatever the dataset (you get to figure out precisely which dataset is intended, from context). Generally, when we want to refer to the j, k'th entry of a matrix $\mathcal{A}$, we will write $\mathcal{A}_{jk}$, so Σ_{jk} is the covariance between the j'th and k'th components of the data.

Useful Facts: 10.1 *Properties of the covariance matrix*

- The j, k'th entry of the covariance matrix is the covariance of the j'th and the k'th components of $\mathbf{x}$, which we write $\mathsf{cov}\left(\left\{x^{(j)}\right\}, \left\{x^{(k)}\right\}\right)$.
- The j, j'th entry of the covariance matrix is the variance of the j'th component of $\mathbf{x}$.
- The covariance matrix is symmetric.
- The covariance matrix is always positive semi-definite; it is positive definite, *unless* there is some vector $\mathbf{a}$ such that $\mathbf{a}^T(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}_i\}\right) = 0$ for all i.

Proposition:

$$\mathsf{Covmat}\left(\{\mathbf{x}\}\right)_{jk} = \mathsf{cov}\left(\left\{x^{(j)}\right\}, \left\{x^{(k)}\right\}\right)$$

Proof: Recall

$$\mathsf{Covmat}\left(\{\mathbf{x}\}\right) = \frac{\sum_i (\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))^T}{N}$$

and the j, k'th entry in this matrix will be

$$\frac{\sum_i (x_i^{(j)} - \mathsf{mean}\left(\left\{x^{(j)}\right\}\right))(x_i^{(k)} - \mathsf{mean}\left(\left\{x^{(k)}\right\}\right))^T}{N}$$

which is $\mathsf{cov}\left(\left\{x^{(j)}\right\}, \left\{x^{(k)}\right\}\right)$.

Proposition:

$$\mathsf{Covmat}\left(\{\mathbf{x}_i\}\right)_{jj} = \Sigma_{jj} = \mathsf{var}\left(\left\{x^{(j)}\right\}\right)$$

Proof:

$$\begin{aligned} \mathsf{Covmat}\left(\{\mathbf{x}\}\right)_{jj} &= \mathsf{cov}\left(\left\{x^{(j)}\right\}, \left\{x^{(j)}\right\}\right) \\ &= \mathsf{var}\left(\left\{x^{(j)}\right\}\right) \end{aligned}$$

Proposition:

$$\mathsf{Covmat}\left(\{\mathbf{x}\}\right) = \mathsf{Covmat}\left(\{\mathbf{x}\}\right)^T$$

Proof: We have

$$\begin{aligned}\mathsf{Covmat}\left(\{\mathbf{x}\}\right)_{jk} &= \mathsf{cov}\left(\left\{x^{(j)}\right\}, \left\{x^{(k)}\right\}\right) \\ &= \mathsf{cov}\left(\left\{x^{(k)}\right\}, \left\{x^{(j)}\right\}\right) \\ &= \mathsf{Covmat}\left(\{\mathbf{x}\}\right)_{kj}\end{aligned}$$

Proposition: *Write* $\Sigma = \mathsf{Covmat}\left(\{\mathbf{x}\}\right)$. *If there is no vector* $\mathbf{a}$ *such that* $\mathbf{a}^T(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right)) = 0$ *for all* i, *then for any vector* $\mathbf{u}$, *such that* $\|\mathbf{u}\| > 0$,

$$\mathbf{u}^T\Sigma\mathbf{u} > 0.$$

If there is such a vector $\mathbf{a}$, *then*

$$\mathbf{u}^T\Sigma\mathbf{u} \geq 0.$$

Proof: We have

$$\begin{aligned}\mathbf{u}^T\Sigma\mathbf{u} &= \frac{1}{N}\sum_i \left[\mathbf{u}^T(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))\right]\left[(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))^T\mathbf{u}\right] \\ &= \frac{1}{N}\sum_i \left[\mathbf{u}^T(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right))\right]^2.\end{aligned}$$

Now this is a sum of squares. If there is some $\mathbf{a}$ such that $\mathbf{a}^T(\mathbf{x}_i - \mathsf{mean}\left(\{\mathbf{x}\}\right)) = 0$ for every i, then the covariance matrix must be positive semidefinite (because the sum of squares could be zero in this case). Otherwise, it is positive definite, because the sum of squares will always be positive.

10.2 USING MEAN AND COVARIANCE TO UNDERSTAND HIGH DIMENSIONAL DATA

The trick to interpreting high dimensional data is to use the mean and covariance to understand the blob. Figure 10.5 shows a two-dimensional data set. Notice that there is obviously some correlation between the x and y coordinates (it's a diagonal

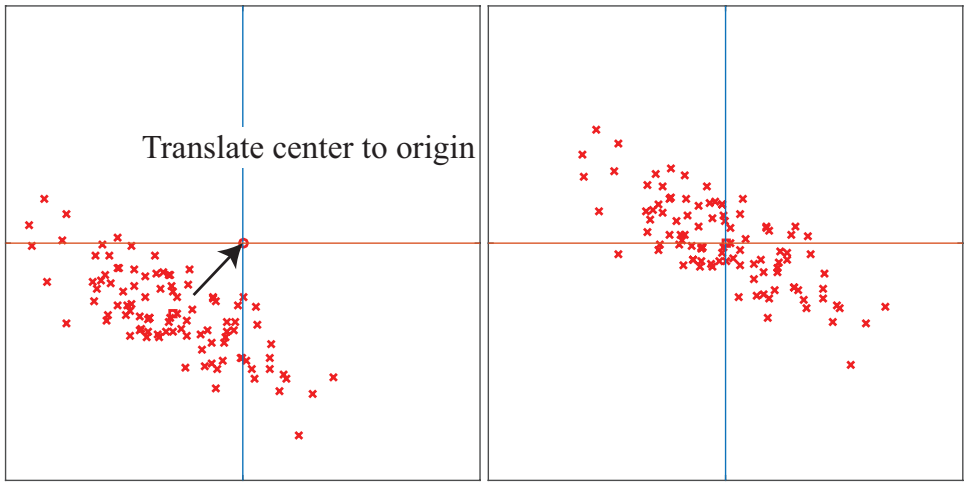

FIGURE 10.5: *On the* **left**, *a "blob" in two dimensions. This is a set of data points that lie somewhat clustered around a single center, given by the mean. I have plotted the mean of these data points with a hollow square (it's easier to see when there is a lot of data). To translate the blob to the origin, we just subtract the mean from each datapoint, yielding the blob on the* **right**.

blob), and that neither x nor y has zero mean. We can easily compute the mean and subtract it from the data points, and this translates the blob so that the origin is at the mean (Figure 10.5). The mean of the new, translated dataset is zero.

Notice this blob is diagonal. We know what that means from our study of correlation – the two measurements are correlated. Now consider *rotating* the blob of data about the origin. This doesn't change the distance between any pair of points, but it does change the overall appearance of the blob of data. We can choose a rotation that means the blob looks (roughly!) like an axis aligned ellipse. *In these coordinates* there is no correlation between the horizontal and vertical components. But one direction has more variance than the other.

It turns out we can extend this approach to high dimensional blobs. We will translate their mean to the origin, then rotate the blob so that there is no correlation between any pair of distinct components (this turns out to be straightforward, which may not be obvious to you). Now the blob looks like an axis-aligned ellipsoid, and we can reason about (a) what axes are "big" and (b) what that means about the original dataset.

10.2.1 Mean and Covariance under Affine Transformations

We have a d dimensional dataset $\{\mathbf{x}\}$. An affine transformation of this data is obtained by choosing some matrix $\mathcal{A}$ and vector $\mathbf{b}$, then forming a new dataset $\{\mathbf{m}\}$, where $\mathbf{m}_i = \mathcal{A}\mathbf{x}_i + \mathbf{b}$. Here $\mathcal{A}$ doesn't have to be square, or symmetric, or anything else; it just has to have second dimension d.

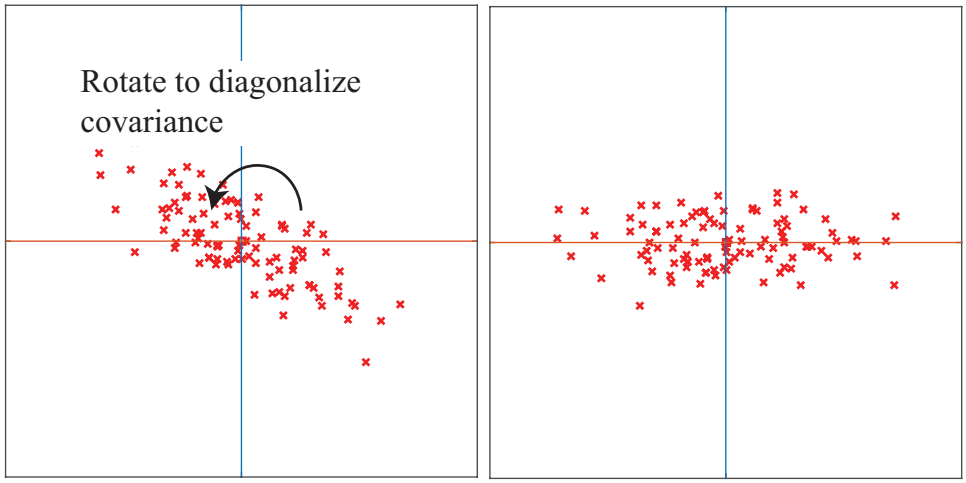

FIGURE 10.6: *On the* **left**, *the translated blob of figure 10.5. This blob lies somewhat diagonally, because the vertical and horizontal components are correlated. On the* **right**, *that blob of data rotated so that there is no correlation between these components. We can now describe the blob by the vertical and horizontal variances alone, as long as we do so in the new coordinate system. In this coordinate system, the vertical variance is significantly larger than the horizontal variance — the blob is short and wide.*

It is easy to compute the mean and covariance of $\{\mathbf{m}\}$. We have

$$\begin{aligned}\text{mean}\left(\{\mathbf{m}\}\right) &= \text{mean}\left(\{\mathcal{A}\mathbf{x}+\mathbf{b}\}\right)\\ &= \mathcal{A}\text{mean}\left(\{\mathbf{x}\}\right)+\mathbf{b},\end{aligned}$$

so you get the new mean by multiplying the original mean by $\mathcal{A}$ and adding $\mathbf{b}$.

The new covariance matrix is easy to compute as well. We have:

$$\begin{aligned}\text{Covmat}\left(\{\mathbf{m}\}\right) &= \text{Covmat}\left(\{\mathcal{A}\mathbf{x}+\mathbf{b}\}\right)\\ &= \frac{\sum_i(\mathbf{m}_i-\text{mean}\left(\{\mathbf{m}\}\right))(\mathbf{m}_i-\text{mean}\left(\{\mathbf{m}\}\right))^T}{N}\\ &= \frac{\sum_i(\mathcal{A}\mathbf{x}_i+\mathbf{b}-\mathcal{A}\text{mean}\left(\{\mathbf{x}\}\right)-\mathbf{b})(\mathcal{A}\mathbf{x}_i+\mathbf{b}-\mathcal{A}\text{mean}\left(\{\mathbf{x}\}\right)-\mathbf{b})^T}{N}\\ &= \frac{\mathcal{A}\left[\sum_i(\mathbf{x}_i-\text{mean}\left(\{\mathbf{x}\}\right))(\mathbf{x}_i-\text{mean}\left(\{\mathbf{x}\}\right))^T\right]\mathcal{A}^T}{N}\\ &= \mathcal{A}\text{Covmat}\left(\{\mathbf{x}\}\right)\mathcal{A}^T.\end{aligned}$$

All this means that we can try and choose affine transformations that yield "good" means and covariance matrices. It is natural to choose $\mathbf{b}$ so that the mean of the new dataset is zero. An appropriate choice of $\mathcal{A}$ can reveal a lot of information about the dataset.

10.2.2 Eigenvectors and Diagonalization

Recall a matrix $\mathcal{M}$ is **symmetric** if $\mathcal{M} = \mathcal{M}^T$. A symmetric matrix is necessarily square. Assume $\mathcal{S}$ is a $d \times d$ symmetric matrix, $\mathbf{u}$ is a $d \times 1$ vector, and λ is a scalar. If we have

$$\mathcal{S}\mathbf{u} = \lambda\mathbf{u}$$

then $\mathbf{u}$ is referred to as an **eigenvector** of $\mathcal{S}$ and λ is the corresponding **eigenvalue**. Matrices don't have to be symmetric to have eigenvectors and eigenvalues, but the symmetric case is the only one of interest to us.

In the case of a symmetric matrix, the eigenvalues are real numbers, and there are d distinct eigenvectors that are normal to one another, and can be scaled to have unit length. They can be stacked into a matrix $\mathcal{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_d]$. This matrix is orthonormal, meaning that $\mathcal{U}^T\mathcal{U} = \mathcal{I}$.

This means that there is a diagonal matrix Λ and an orthonormal matrix $\mathcal{U}$ such that

$$\mathcal{S}\mathcal{U} = \mathcal{U}\Lambda.$$

In fact, there is a large number of such matrices, because we can reorder the eigenvectors in the matrix $\mathcal{U}$, and the equation still holds with a new Λ, obtained by reordering the diagonal elements of the original Λ. There is no reason to keep track of this complexity. Instead, we adopt the convention that the elements of $\mathcal{U}$ are always ordered so that the elements of Λ are sorted along the diagonal, with the largest value coming first. This gives us a particularly important procedure.

Procedure: 10.1 *Diagonalizing a symmetric matrix*

We can convert any symmetric matrix $\mathcal{S}$ to a diagonal form by computing

$$\mathcal{U}^T\mathcal{S}\mathcal{U} = \Lambda.$$

Numerical and statistical programming environments have procedures to compute $\mathcal{U}$ and Λ for you. We assume that the elements of $\mathcal{U}$ are always ordered so that the elements of Λ are sorted along the diagonal, with the largest value coming first.

Useful Facts: 10.2 *Orthonormal matrices are rotations*

You should think of orthonormal matrices as rotations, because they do not change lengths or angles. For $\mathbf{x}$ a vector, $\mathcal{R}$ an orthonormal matrix, and $\mathbf{m} = \mathcal{R}\mathbf{x}$, we have

$$\mathbf{u}^T\mathbf{u} = \mathbf{x}^T\mathcal{R}^T\mathcal{R}\mathbf{x} = \mathbf{x}^T\mathcal{I}\mathbf{x} = \mathbf{x}^T\mathbf{x}.$$

This means that $\mathcal{R}$ doesn't change lengths. For $\mathbf{y}$, $\mathbf{z}$ both unit vectors, we have that the cosine of the angle between them is

$$\mathbf{y}^T\mathbf{x}.$$

By the argument above, the inner product of $\mathcal{R}\mathbf{y}$ and $\mathcal{R}\mathbf{x}$ is the same as $\mathbf{y}^T\mathbf{x}$. This means that $\mathcal{R}$ doesn't change angles, either.

10.2.3 Diagonalizing Covariance by Rotating Blobs

We start with a dataset of N d-dimensional vectors $\{\mathbf{x}\}$. We can translate this dataset to have zero mean, forming a new dataset $\{\mathbf{m}\}$ where $\mathbf{m}_i = \mathbf{x}_i - \mathsf{mean}(\{\mathbf{x}\})$. Now recall that, if we were to form a new dataset $\{\mathbf{a}\}$ where

$$\mathbf{a}_i = \mathcal{A}\mathbf{m}_i$$

the covariance matrix of $\{\mathbf{a}\}$ would be

$$\mathsf{Covmat}(\{\mathbf{a}\}) = \mathcal{A}\mathsf{Covmat}(\{\mathbf{m}\})\mathcal{A}^T = \mathcal{A}\mathsf{Covmat}(\{\mathbf{x}\})\mathcal{A}^T.$$

Recall also we can diagonalize $\mathsf{Covmat}(\{\mathbf{m}\}) = \mathsf{Covmat}(\{\mathbf{x}\})$ to get

$$\mathcal{U}^T\mathsf{Covmat}(\{\mathbf{x}\})\mathcal{U} = \Lambda.$$

But this means we could form the dataset $\{\mathbf{r}\}$, using the rule

$$\mathbf{r}_i = \mathcal{U}^T\mathbf{m}_i = \mathcal{U}^T(\mathbf{x}_i - \mathsf{mean}(\{\mathbf{x}\})).$$

The mean of this new dataset is clearly $\mathbf{0}$. The covariance of this dataset is

$$\begin{aligned}\mathsf{Covmat}(\{\mathbf{r}\}) &= \mathsf{Covmat}\left(\{\mathcal{U}^T\mathbf{x}\}\right) \\ &= \mathcal{U}^T\mathsf{Covmat}(\{\mathbf{x}\})\mathcal{U} \\ &= \Lambda,\end{aligned}$$

where Λ is a diagonal matrix of eigenvalues of $\mathsf{Covmat}(\{\mathbf{x}\})$ that we obtained by diagonalization. We now have a very useful fact about $\{\mathbf{r}\}$: its covariance matrix is diagonal. This means that every pair of distinct components has covariance zero, and so has correlation zero. Remember that, in describing diagonalization, we

adopted the convention that the eigenvectors of the matrix being diagonalized were ordered so that the eigenvalues are sorted in descending order along the diagonal of Λ. Our choice of ordering means that the first component of $\mathbf{r}$ has the highest variance, the second component has the second highest variance, and so on.

The transformation from $\{\mathbf{x}\}$ to $\{\mathbf{r}\}$ is a translation followed by a rotation (remember $\mathcal{U}$ is orthonormal, and so a rotation). So this transformation is a high dimensional version of what I showed in Figures 10.5 and 10.6.

Useful Fact: 10.3 *You can transform data to zero mean and diagonal covariance*

We can translate and rotate *any* blob of data into a coordinate system where it has (a) zero mean and (b) diagonal covariance matrix.

10.2.4 Approximating Blobs

The covariance matrix of $\{\mathbf{r}\}$ is diagonal, and the values on the diagonal are interesting. It is quite usual for high dimensional datasets to have a small number of large values on the diagonal, and a lot of small values. This means that the blob of data is really a low dimensional blob in a high dimensional space. For example, think about a line segment (a 1D blob) in 3D. As another example, look at Figure 10.3; the scatterplot matrix strongly suggests that the blob of data is flattened (eg look at the petal width vs petal length plot).

The blob represented by $\{\mathbf{r}\}$ is low dimensional in a very strong sense. We need some notation to see this. The data set $\{\mathbf{r}\}$ is d-dimensional. We will try to represent it with an s dimensional dataset, and see what error we incur. Choose some $s < d$. Now take each data point $\mathbf{r}_i$ and replace the last $d - s$ components with 0. Call the resulting data item $\mathbf{p}_i$. We should like to know the average error in representing $\mathbf{r}_i$ with $\mathbf{p}_i$.

This error is

$$\frac{1}{N}\sum_i \left[(\mathbf{r}_i - \mathbf{p}_i)^T (\mathbf{r}_i - \mathbf{p}_i)^T\right].$$

Write $r_i^{(j)}$ for the j' component of $\mathbf{r}_i$, and so on. Remember that $\mathbf{p}_i$ is zero in the last $d - s$ components. The error is then

$$\frac{1}{N}\sum_i \left[\sum_{j>s}^{j=d} \left(r_i^{(j)}\right)^2\right].$$

But we know this number, because we know that $\{\mathbf{r}\}$ has zero mean. The error is

$$\sum_{j>s}^{j=d} \mathsf{var}\left(\left\{r^{(j)}\right\}\right)$$

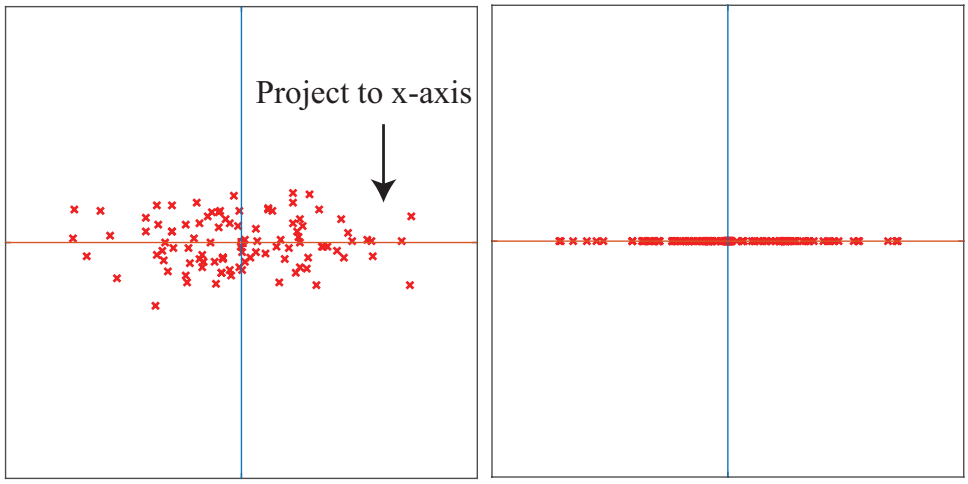

FIGURE 10.7: *On the* **left**, *the translated and rotated blob of figure 10.6. This blob is stretched — one direction has more variance than another. Setting the y coordinate to zero for each of these datapoints results in a representation that has relatively low error, because there isn't much variance in these values. This results in the blob on the* **right**. *The text shows how the error that results from this projection is computed.*

which is the sum of the diagonal elements of the covariance matrix from r, r to d, d. If this sum is small compared to the sum of the first r components, then dropping the last $d - r$ components results in a small error. In that case, we could think about the data as being r dimensional. Figure 10.7 shows the result of using this approach to represent the blob we've used as a running example as a 1D dataset.

This is an observation of great practical importance. As a matter of experimental fact, a great deal of high dimensional data produces relatively low dimensional blobs. We can identify the main directions of variation in these blobs, and use them to understand and to represent the dataset.

10.2.5 Example: Transforming the Height-Weight Blob

Translating a blob of data doesn't change the scatterplot matrix in any interesting way (the axes change, but the picture doesn't). Rotating a blob produces really interesting results, however. Figure 10.8 shows the dataset of figure 10.4, translated to the origin and rotated to diagonalize it. Now we do not have names for each component of the data (they're linear combinations of the original components), but each pair is now not correlated. This blob has some interesting shape features. Figure 10.8 shows the gross shape of the blob best. Each panel of this figure has the same scale in each direction. You can see the blob extends about 80 units in direction 1, but only about 15 units in direction 2, and much less in the other two directions. You should think of this blob as being rather cigar-shaped; it's long in one direction, but there isn't much in the others. The cigar metaphor isn't perfect because there aren't any 4 dimensional cigars, but it's helpful. You can think of

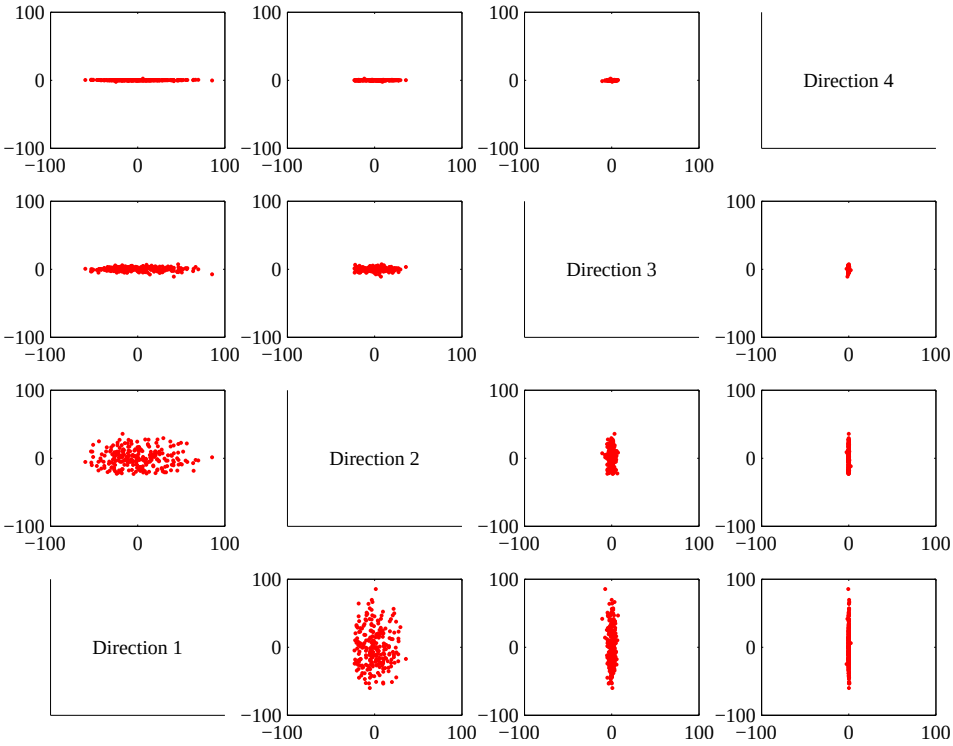

FIGURE 10.8: *A panel plot of the bodyfat dataset of figure 10.4, now rotated so that the covariance between all pairs of distinct dimensions is zero. Now we do not know names for the directions — they're linear combinations of the original variables. Each scatterplot is on the same set of axes, so you can see that the dataset extends more in some directions than in others.*

each panel of this figure as showing views down each of the four axes of the cigar.

Now look at figure 10.9. This shows the same rotation of the same blob of data, but now the scales on the axis have changed to get the best look at the detailed shape of the blob. First, you can see that blob is a little curved (look at the projection onto direction 2 and direction 4). There might be some effect here worth studying. Second, you can see that some points seem to lie away from the main blob. I have plotted each data point with a dot, and the interesting points with a number. These points are clearly special in some way.

The problem with these figures is that the axes are meaningless. The components are weighted combinations of components of the original data, so they don't have any units, etc. This is annoying, and often inconvenient. But I obtained Figure 10.8 by translating, rotating and projecting data. It's straightforward to undo the rotation and the translation – this takes the projected blob (which we know to be a good approximation of the rotated and translated blob) back to where the original blob was. Rotation and translation don't change distances, so the result is a good approximation of the original blob, but now in the original blob's coor-

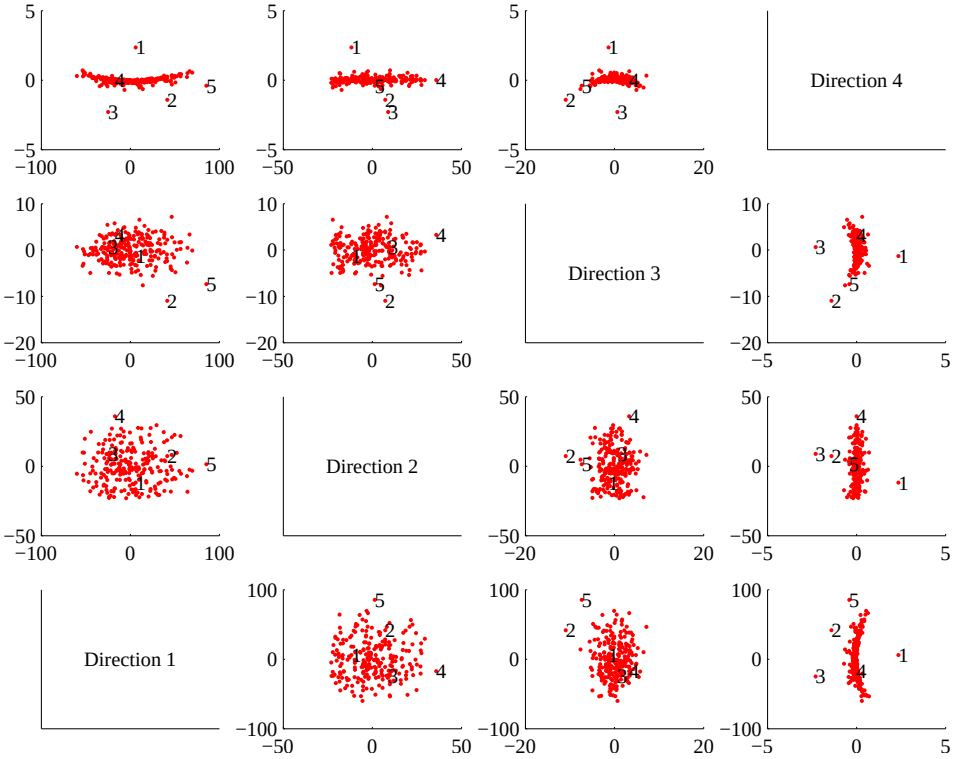

FIGURE 10.9: *A panel plot of the bodyfat dataset of figure 10.4, now rotated so that the covariance between all pairs of distinct dimensions is zero. Now we do not know names for the directions — they're linear combinations of the original variables. I have scaled the axes so you can see details; notice that the blob is a little curved, and there are several data points that seem to lie some way away from the blob, which I have numbered.*

dinates. Figure 10.10 shows what happens to the data of Figure 10.4. This is a two dimensional version of the original dataset, embedded like a thin pancake of data in a four dimensional space. Crucially, it represents the original dataset quite accurately.

10.3 PRINCIPAL COMPONENTS ANALYSIS

We have seen that a blob of data can be translated so that it has zero mean, then rotated so the covariance matrix is diagonal. In this coordinate system, we can set some components to zero, and get a representation of the data that is still accurate. The rotation and translation can be undone, yielding a dataset that is in the same coordinates as the original, but lower dimensional. The new dataset is a good approximation to the old dataset. All this yields a really powerful idea: we can represent the original dataset with a small number of appropriately chosen vectors.

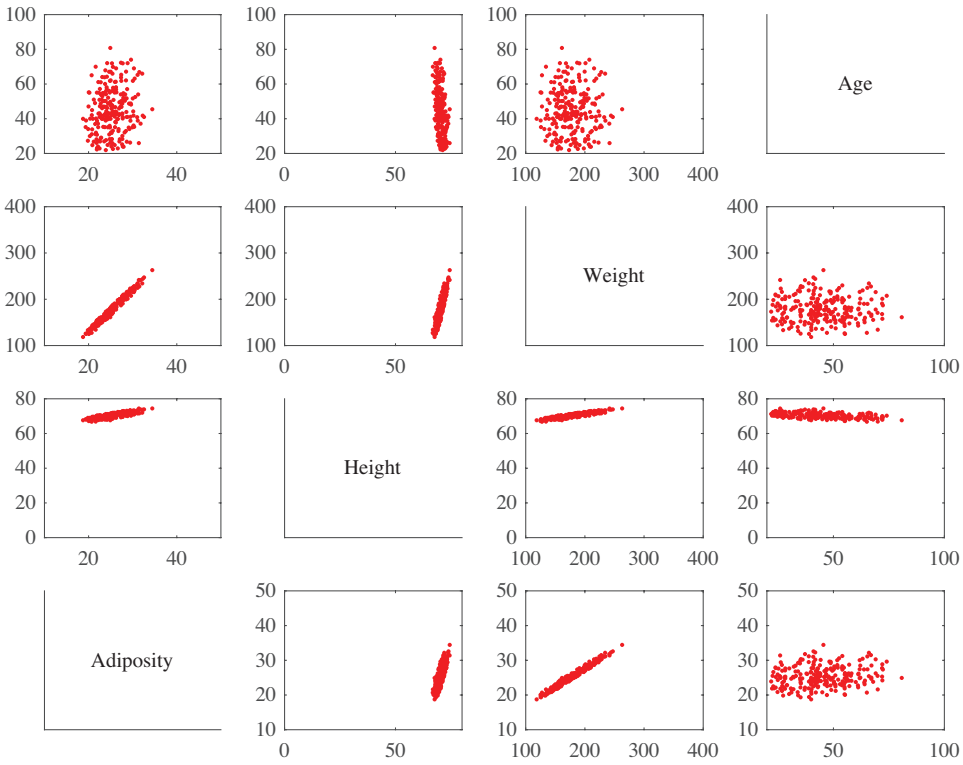

FIGURE 10.10: *The data of Figure 10.4, represented by translating and rotating so that the covariance is diagonal, projecting off the two smallest directions, then undoing the rotation and translation. This blob of data is two dimensional (because we projected off two dimensions), but is represented in a four dimensional space. You can think of it as a thin pancake of data in the four dimensional space (you should compare to Figure 10.4 on page 304). It is a good representation of the original data. Notice that it looks slightly thickened on edge, because it isn't aligned with the coordinate system – think of a view of a plate at a slight slant.*

We start with a dataset of N d-dimensional vectors $\{\mathbf{x}\}$. We translate this dataset to have zero mean, forming a new dataset $\{\mathbf{m}\}$ where $\mathbf{m}_i = \mathbf{x}_i - \mathsf{mean}\,(\{\mathbf{x}\})$. We diagonalize $\mathsf{Covmat}\,(\{\mathbf{m}\}) = \mathsf{Covmat}\,(\{\mathbf{x}\})$ to get

$$\mathcal{U}^T \mathsf{Covmat}\,(\{\mathbf{x}\})\mathcal{U} = \Lambda$$

and form the dataset $\{\mathbf{r}\}$, using the rule

$$\mathbf{r}_i = \mathcal{U}^T \mathbf{m}_i = \mathcal{U}^T(\mathbf{x}_i - \mathsf{mean}\,(\{\mathbf{x}\})).$$

We saw the mean of this dataset is zero, and the covariance is diagonal. We then represented the d-dimensional data set $\{\mathbf{r}\}$ with an s dimensional dataset, by choosing some $r < d$, then taking each data point $\mathbf{r}_i$ and replacing the last $d - s$ components with 0. We call the resulting data item $\mathbf{p}_i$.

Now consider undoing the rotation and translation. We would form a new dataset $\{\hat{\mathbf{x}}\}$, with the i'th element given by

$$\hat{\mathbf{x}}_i = \mathcal{U}\mathbf{p}_i + \mathsf{mean}\left(\{\mathbf{x}\}\right)$$

(you should check this expression). But this expression says that $\hat{\mathbf{x}}_i$ is constructed by forming a weighted sum of the first s columns of $\mathcal{U}$ (because all the other components of $\mathbf{p}_i$ are zero), then adding $\mathsf{mean}\left(\{\mathbf{x}\}\right)$. If we write $\mathbf{u}_j$ for the j'th column of $\mathcal{U}$, we have

$$\hat{\mathbf{x}}_i = \sum_{j=1}^{s} r_i^{(j)}\mathbf{u}_j + \mathsf{mean}\left(\{\mathbf{x}\}\right).$$

What is important about this sum is that s is usually a lot less than d. The $\mathbf{u}_j$ are known as **principal components** of the dataset.

Remember this: *Data items in a d dimensional data set can usually be represented with good accuracy as a weighted sum of a small number s of d dimensional vectors, together with the mean. This means that the dataset lies on an s-dimensional subspace of the d-dimensional space. The subspace is spanned by the principal components of the data.*

We can easily determine the error in approximating $\{\mathbf{x}\}$ with $\{\hat{\mathbf{x}}\}$. The error in representing $\{\mathbf{r}\}$ by $\{\mathbf{p}_s\}$ was easy to compute. We had

$$\frac{1}{N}\sum_i \left[\left(\mathbf{r}_i - \mathbf{p}_i\right)^T \left(\mathbf{r}_i - \mathbf{p}_i\right)^T\right] = \frac{1}{N}\sum_i \left[\sum_{j>s}^{j=d} \left(r_i^{(j)}\right)^2\right].$$

This was the sum of the diagonal elements of the covariance matrix of $\{\mathbf{r}\}$ from s, s to d, d. If this sum is small compared to the sum of the first s components, then dropping the last $d - s$ components results in a small error.

The error in representing $\{\mathbf{x}\}$ with $\{\hat{\mathbf{x}}\}$ is now easy to get. Rotations and translations do not change lengths. This means that

$$\left\|\mathbf{x}_i - \hat{\mathbf{x}}_i\right\|^2 = \left\|\mathbf{r}_i - \mathbf{p}_{r,i}\right\|^2 = \sum_{u=r+1}^{d} (\mathbf{r}_i^{(u)})^2$$

which is the sum of the diagonal elements of the covariance matrix of $\{\mathbf{r}\}$ from s, s to d, d which is easy to evaluate, because these are the values of the $d-s$ eigenvalues that we decided to ignore. Now we could choose s by identifying how much error we can tolerate. More usual is to plot the eigenvalues of the covariance matrix, and look for a "knee", like that in Figure 10.11. You can see that the sum of remaining eigenvalues is small.

Procedure: 10.2 *Principal Components Analysis*

Assume we have a general data set $\mathbf{x}_i$, consisting of N d-dimensional vectors. Now write $\Sigma = \mathsf{Covmat}(\{\mathbf{x}\})$ for the covariance matrix.
Form $\mathcal{U}$, Λ, such that

$$\Sigma\mathcal{U} = \mathcal{U}\Lambda$$

(these are the eigenvectors and eigenvalues of Σ). Ensure that the entries of Λ are sorted in decreasing order. Choose r, the number of dimensions you wish to represent. Typically, we do this by plotting the eigenvalues and looking for a "knee" (Figure 10.11). It is quite usual to do this by hand.
Constructing a low-dimensional representation: For $1 \leq j \leq r$, write $\mathbf{u}_i$ for the i'th column of $\mathcal{U}$. Represent the data point $\mathbf{x}_i$ as

$$\hat{\mathbf{x}}_i = \mathsf{mean}(\{\mathbf{x}\}) + \sum_{j=1}^{s} \left[\mathbf{u}_j^T (\mathbf{x}_i - \mathsf{mean}(\{\mathbf{x}\}))\right] \mathbf{u}_j$$

10.3.1 Example: Representing Colors with Principal Components

Diffuse surfaces reflect light uniformly in all directions. Examples of diffuse surfaces include matte paint, many styles of cloth, many rough materials (bark, cement, stone, etc.). One way to tell a diffuse surface is that it does not look brighter (or darker) when you look at it along different directions. Diffuse surfaces can be colored, because the surface reflects different fractions of the light falling on it at different wavelengths. This effect can be represented by measuring the spectral reflectance of a surface, which is the fraction of light the surface reflects as a function of wavelength. This is usually measured in the visual range of wavelengths (about 380nm to about 770 nm). Typical measurements are every few nm, depending on the measurement device. I obtained data for 1995 different surfaces from http://www.cs.sfu.ca/~colour/data/ (there are a variety of great datasets here, from Kobus Barnard).

Each spectrum has 101 measurements, which are spaced 4nm apart. This represents surface properties to far greater precision than is really useful. Physical properties of surfaces suggest that the reflectance can't change too fast from wavelength to wavelength. It turns out that very few principal components are sufficient to describe almost any spectral reflectance function. Figure 10.11 shows the mean spectral reflectance of this dataset, and Figure 10.11 shows the eigenvalues of the covariance matrix.

This is tremendously useful in practice. One should think of a spectral reflectance as a function, usually written $\rho(\lambda)$. What the principal components analysis tells us is that we can represent this function rather accurately on a (really small) finite dimensional basis. This basis is shown in figure 10.11. This means

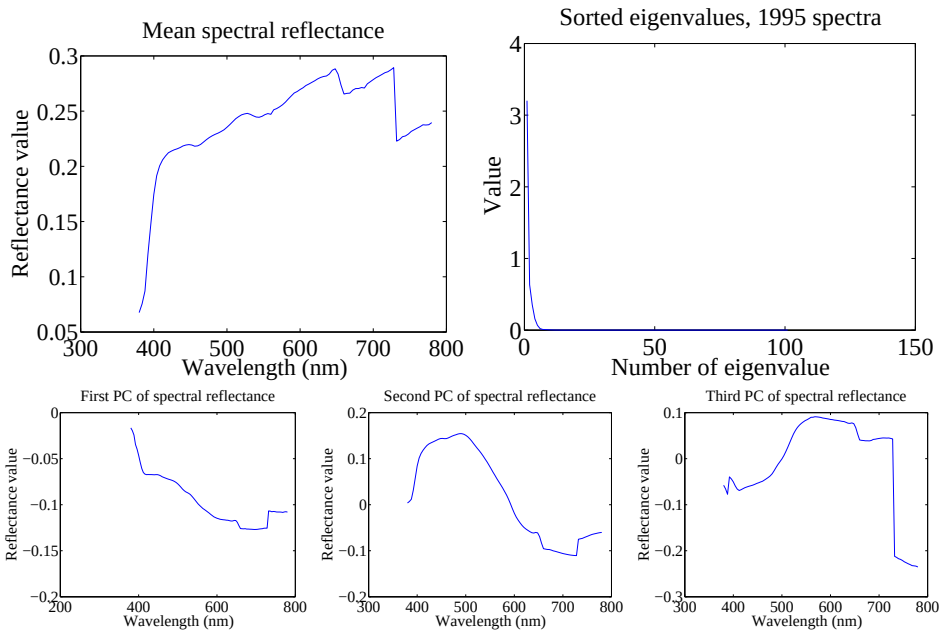

FIGURE 10.11: *On the* **top left**, *the mean spectral reflectance of a dataset of 1995 spectral reflectances, collected by Kobus Barnard (at http://www.cs.sfu.ca/~colour/data/). On the* **top right**, *eigenvalues of the covariance matrix of spectral reflectance data, from a dataset of 1995 spectral reflectances, collected by Kobus Barnard (at http://www.cs.sfu.ca/~colour/data/). Notice how the first few eigenvalues are large, but most are very small; this suggests that a good representation using few principal components is available. The* **bottom row** *shows the first three principal components. A linear combination of these, with appropriate weights, added to the mean of figure* **??**, *gives a good representation of the dataset.*

that there is a mean function $r(\lambda)$ and k functions $\phi_m(\lambda)$ such that, for any $\rho(\lambda)$,

$$\rho(\lambda) = r(\lambda) + \sum_{i=1}^{k} c_i \phi_i(\lambda) + e(\lambda)$$

where $e(\lambda)$ is the error of the representation, which we know is small (because it consists of all the other principal components, which have tiny variance). In the case of spectral reflectances, using a value of k around 3-5 works fine for most applications (Figure 10.12). This is useful, because when we want to predict what a particular object will look like under a particular light, we don't need to use a detailed spectral reflectance model; instead, it's enough to know the c_i for that object. This comes in useful in a variety of rendering applications in computer graphics. It is also the key step in an important computer vision problem, called **color constancy**. In this problem, we see a picture of a world of colored objects under unknown colored lights, and must determine what color the objects are. Modern color constancy systems are quite accurate, even though the problem

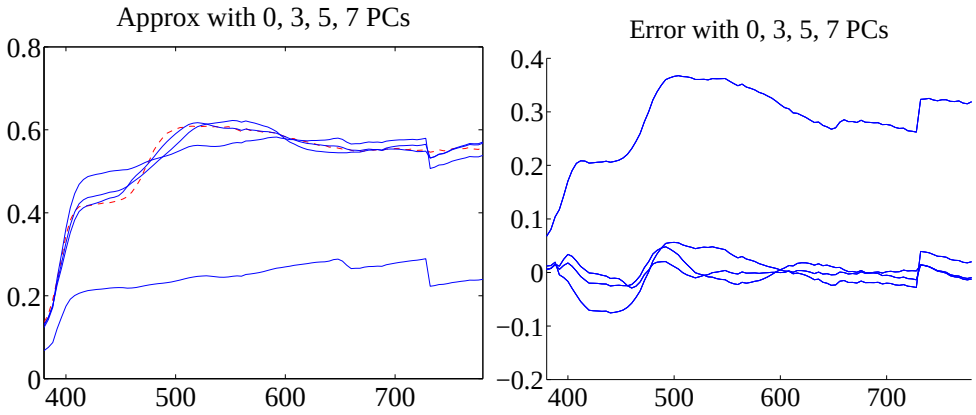

FIGURE 10.12: *On the* **left**, *a spectral reflectance curve (dashed) and approximations using the mean, the mean and 3 principal components, the mean and 5 principal components, and the mean and 7 principal components. Notice the mean is a relatively poor approximation, but as the number of principal components goes up, the error falls rather quickly. On the* **right** *is the error for these approximations. Figure plotted from a dataset of 1995 spectral reflectances, collected by Kobus Barnard (at http://www.cs.sfu.ca/~colour/data/).*

sounds underconstrained. This is because they are able to exploit the fact that relatively few c_i are enough to accurately describe a surface reflectance.

10.3.2 Example: Representing Faces with Principal Components

An image is usually represented as an array of values. We will consider intensity images, so there is a single intensity value in each cell. You can turn the image into a vector by rearranging it, for example stacking the columns onto one another. This means you can take the principal components of a set of images. Doing so was something of a fashionable pastime in computer vision for a while, though there are some reasons that this is not a great representation of pictures. However, the representation yields pictures that can give great intuition into a dataset.

Figure **??** shows the mean of a set of face images encoding facial expressions of Japanese women (available at http://www.kasrl.org/jaffe.html; there are tons of face datasets at http://www.face-rec.org/databases/). I reduced the images to 64x64, which gives a 4096 dimensional vector. The eigenvalues of the covariance of this dataset are shown in figure 10.13; there are 4096 of them, so it's hard to see a trend, but the zoomed figure suggests that the first couple of hundred contain most of the variance. Once we have constructed the principal components, they can be rearranged into images; these images are shown in figure 10.14. Principal components give quite good approximations to real images (figure 10.15).

The principal components sketch out the main kinds of variation in facial expression. Notice how the mean face in Figure 10.14 looks like a relaxed face, but with fuzzy boundaries. This is because the faces can't be precisely aligned, because each face has a slightly different shape. The way to interpret the components is to

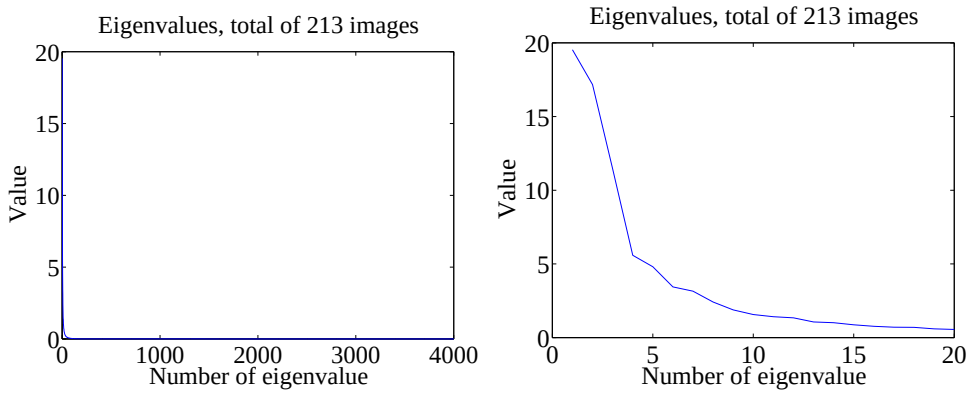

FIGURE 10.13: *On the* **left**,*the eigenvalues of the covariance of the Japanese facial expression dataset; there are 4096, so it's hard to see the curve (which is packed to the left). On the* **right**, *a zoomed version of the curve, showing how quickly the values of the eigenvalues get small.*

remember one adjusts the mean towards a data point by adding (or subtracting) some scale times the component. So the first few principal components have to do with the shape of the haircut; by the fourth, we are dealing with taller/shorter faces; then several components have to do with the height of the eyebrows, the shape of the chin, and the position of the mouth; and so on. These are all images of women who are not wearing spectacles. In face pictures taken from a wider set of models, moustaches, beards and spectacles all typically appear in the first couple of dozen principal components.

10.4 MULTI-DIMENSIONAL SCALING

One way to get insight into a dataset is to plot it. But choosing what to plot for a high dimensional dataset could be difficult. Assume we must plot the dataset in two dimensions (by far the most common choice). We wish to build a scatter plot in two dimensions — but where should we plot each data point? One natural requirement is that the points be laid out in two dimensions in a way that reflects how they sit in many dimensions. In particular, we would like points that are far apart in the high dimensional space to be far apart in the plot, and points that are close in the high dimensional space to be close in the plot.

10.4.1 Choosing Low D Points using High D Distances

We will plot the high dimensional point $\mathbf{x}_i$ at $\mathbf{v}_i$, which is a two-dimensional vector. Now the squared distance between points i and j in the high dimensional space is

$$D_{ij}^{(2)}(\mathbf{x}) = (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)$$

(where the superscript is to remind you that this is a squared distance). We could build an $N \times N$ matrix of squared distances, which we write $\mathcal{D}^{(2)}(\mathbf{x})$. The i, j'th

FIGURE 10.14: *The mean and first 16 principal components of the Japanese facial expression dataset.*

entry in this matrix is $D_{ij}^{(2)}(\mathbf{x})$, and the $\mathbf{x}$ argument means that the distances are between points in the high-dimensional space. Now we could choose the $\mathbf{v}_i$ to make

$$\sum_{ij} \left(D_{ij}^{(2)}(\mathbf{x}) - D_{ij}^{(2)}(\mathbf{v}) \right)^2$$

as small as possible. Doing so should mean that points that are far apart in the high dimensional space are far apart in the plot, and that points that are close in the high dimensional space are close in the plot.

In its current form, the expression is difficult to deal with, but we can refine it. Because translation does not change the distances between points, it cannot change either of the $\mathcal{D}^{(2)}$ matrices. So it is enough to solve the case when the mean of the points $\mathbf{x}_i$ is zero. We can assume that

$$\frac{1}{N} \sum_i \mathbf{x}_i = \mathbf{0}.$$

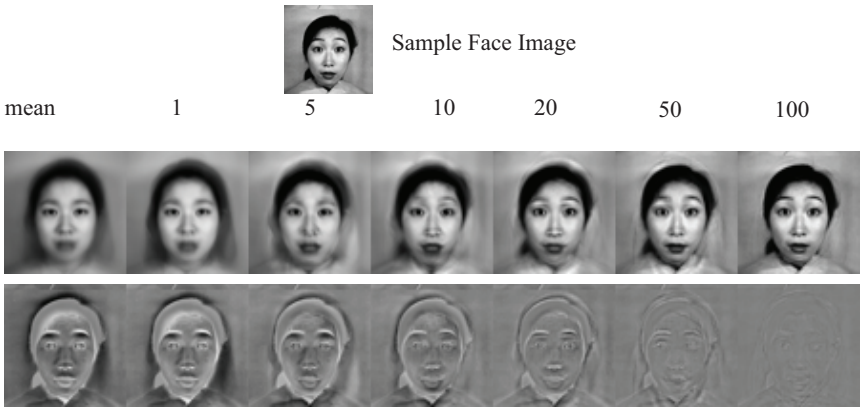

FIGURE 10.15: *Approximating a face image by the mean and some principal components; notice how good the approximation becomes with relatively few components.*

Now write $\mathbf{1}$ for the n-dimensional vector containing all ones, and $\mathcal{I}$ for the identity matrix. Notice that

$$D_{ij}^{(2)} = (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_i - 2\mathbf{x}_i \cdot \mathbf{x}_j + \mathbf{x}_j \cdot \mathbf{x}_j.$$

Now write

$$\mathcal{A} = \left[\mathcal{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\right].$$

Using this expression, you can show that the matrix $\mathcal{M}$, defined below,

$$\mathcal{M}(\mathbf{x}) = -\frac{1}{2}\mathcal{A}\mathcal{D}^{(2)}(\mathbf{x})\mathcal{A}^T$$

has i, jth entry $\mathbf{x}_i \cdot \mathbf{x}_j$ (exercises). I now argue that, to make $\mathcal{D}^{(2)}(\mathbf{v})$ is close to $\mathcal{D}^{(2)}(\mathbf{x})$, it is enough to make $\mathcal{M}(\mathbf{v})$ close to $\mathcal{M}(\mathbf{x})$. Proving this will take us out of our way unnecessarily, so I omit a proof.

We need some notation. Take the dataset of N d-dimensional column vectors $\mathbf{x}_i$, and form a matrix $\mathcal{X}$ by stacking the vectors, so

$$\mathcal{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \cdots \\ \mathbf{x}_N^T \end{bmatrix}.$$

In this notation, we have

$$\mathcal{M}(\mathbf{x}) = \mathcal{X}\mathcal{X}^T.$$

Notice $\mathcal{M}(\mathbf{x})$ is symmetric, and it is positive semidefinite. It can't be positive definite, because the data is zero mean, so $\mathcal{M}(\mathbf{x})\mathbf{1} = 0$.

We can choose a set of $\mathbf{v}_i$ that makes $\mathcal{D}^{(2)}(\mathbf{v})$ close to $\mathcal{D}^{(2)}(\mathbf{x})$ quite easily.

To obtain a $\mathcal{M}(\mathbf{v})$ that is close to $\mathcal{M}(\mathbf{x})$, we need to choose $\mathcal{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N]^T$ so that $\mathcal{V}\mathcal{V}^T$ is close to $\mathcal{M}(\mathbf{x})$. We are computing an approximate factorization of the matrix $\mathcal{M}(\mathbf{x})$.

10.4.2 Factoring a Dot-Product Matrix

We seek a set of k dimensional $\mathbf{v}$ that can be stacked into a matrix $\mathcal{V}$. This must produce a $\mathcal{M}(\mathbf{v}) = \mathcal{V}\mathcal{V}^T$ that must (a) be as close as possible to $\mathcal{M}(\mathbf{x})$ and (b) have rank at most k. It can't have rank larger than k because there must be some $\mathcal{V}$ which is $N \times k$ so that $\mathcal{M}(\mathbf{v}) = \mathcal{V}\mathcal{V}^T$. The rows of this $\mathcal{V}$ are our $\mathbf{v}_i^T$.

We can obtain the best factorization of $\mathcal{M}(\mathbf{x})$ from a diagonalization. Write write $\mathcal{U}$ for the matrix of eigenvectors of $\mathcal{M}(\mathbf{x})$ and Λ for the diagonal matrix of eigenvalues sorted in descending order, so we have

$$\mathcal{M}(\mathbf{x}) = \mathcal{U}\Lambda\mathcal{U}^T$$

and write $\Lambda^{(1/2)}$ for the matrix of positive square roots of the eigenvalues. Now we have

$$\mathcal{M}(\mathbf{x}) = \mathcal{U}\Lambda^{1/2}\Lambda^{1/2}\mathcal{U}^T = \left(\mathcal{U}\Lambda^{1/2}\right)\left(\mathcal{U}\Lambda^{1/2}\right)^T$$

which allows us to write

$$\mathcal{X} = \mathcal{U}\Lambda^{1/2}.$$

Now think about approximating $\mathcal{M}(\mathbf{x})$ by the matrix $\mathcal{M}(\mathbf{v})$. The error is a sum of squares of the entries,

$$\text{err}(\mathcal{M}(\mathbf{x}), \mathcal{A}) = \sum_{ij}(m_{ij} - a_{ij})^2.$$

Because $\mathcal{U}$ is a rotation, it is straightforward to show that

$$\text{err}(\mathcal{U}^T\mathcal{M}(\mathbf{x})\mathcal{U}, \mathcal{U}^T\mathcal{M}(\mathbf{v})\mathcal{U}) = \text{err}(\mathcal{M}(\mathbf{x}), \mathcal{M}(\mathbf{v})).$$

But

$$\mathcal{U}^T\mathcal{M}(\mathbf{x})\mathcal{U} = \Lambda$$

which means that we could find $\mathcal{M}(\mathbf{v})$ from the best rank k approximation to Λ. This is obtained by setting all but the k largest entries of Λ to zero. Call the resulting matrix Λ_k. Then we have

$$\mathcal{M}(\mathbf{v}) = \mathcal{U}\Lambda_k\mathcal{U}$$

and

$$\mathcal{V} = \mathcal{U}\Lambda_k^{(1/2)}.$$

The first k columns of $\mathcal{V}$ are non-zero. We drop the remaining $N - k$ columns of zeros. The rows of the resulting matrix are our $\mathbf{v}_i$, and we can plot these. This method for constructing a plot is known as **principal coordinate analysis**.

This plot might not be perfect, because reducing the dimension of the data points should cause some distortions. In many cases, the distortions are tolerable. In other cases, we might need to use a more sophisticated scoring system that penalizes some kinds of distortion more strongly than others. There are many ways to do this; the general problem is known as **multidimensional scaling**.

Procedure: 10.3 *Principal Coordinate Analysis*

Assume we have a matrix $D^{(2)}$ consisting of the squared differences between each pair of N points. We do not need to know the points. We wish to compute a set of points in r dimensions, such that the distances between these points are as similar as possible to the distances in $D^{(2)}$.

- Form $\mathcal{A} = \left[\mathcal{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\right]$.
- Form $\mathcal{W} = \frac{1}{2}\mathcal{A}\mathcal{D}^{(2)}\mathcal{A}^T$.
- Form $\mathcal{U}$, Λ, such that $\mathcal{W}\mathcal{U} = \mathcal{U}\Lambda$ (these are the eigenvectors and eigenvalues of $\mathcal{W}$). Ensure that the entries of Λ are sorted in decreasing order.
- Choose r, the number of dimensions you wish to represent. Form Λ_r, the top left $r \times r$ block of Λ. Form $\Lambda_r^{(1/2)}$, whose entries are the positive square roots of Λ_r. Form $\mathcal{U}_r$, the matrix consisting of the first r columns of $\mathcal{U}$.

Then

$$\mathcal{V}^T = \Lambda_r^{(1/2)}\mathcal{U}_r^T = [\mathbf{v}_1, \ldots, \mathbf{v}_N]$$

is the set of points to plot.

10.4.3 Example: Mapping with Multidimensional Scaling

Multidimensional scaling gets positions (the $\mathcal{V}$ of section 10.4.1) from distances (the $\mathcal{D}^{(2)}(\mathbf{x})$ of section 10.4.1). This means we can use the method to build maps from distances alone. I collected distance information from the web (I used http://www.distancefromto.net, but a google search on "city distances" yields a wide range of possible sources), then applied multidimensional scaling. I obtained distances between the South African provincial capitals, in kilometers. I then used principal coordinate analysis to find positions for each capital, and rotated, translated and scaled the resulting plot to check it against a real map (Figure 10.16).

One natural use of principal coordinate analysis is to see if one can spot any structure in a dataset. Does the dataset form a blob, or is it clumpy? This isn't a perfect test, but it's a good way to look and see if anything interesting is happening. In figure 10.17, I show a 3D plot of the spectral data, reduced to three dimensions using principal coordinate analysis. The plot is quite interesting. You should notice that the data points are spread out in 3D, but actually seem to lie on a complicated curved surface — they very clearly don't form a uniform blob. To me, the structure looks somewhat like a butterfly. I don't know why this occurs (perhaps the universe is doodling), but it certainly suggests that something worth investigating is going on. Perhaps the choice of samples that were measured is funny; perhaps the

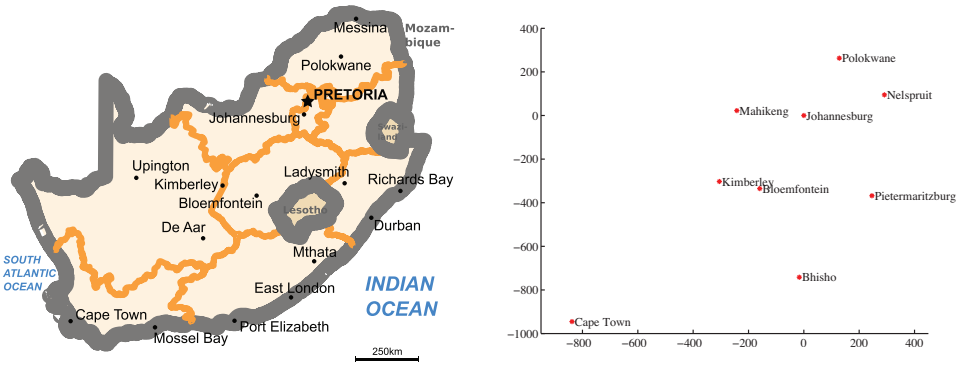

FIGURE 10.16: *On the* **left**, *a public domain map of South Africa, obtained from* http://commons.wikimedia.org/wiki/File:Map_of_South_Africa.svg, *and edited to remove surrounding countries. On the* **right**, *the locations of the cities inferred by multidimensional scaling, rotated, translated and scaled to allow a comparison to the map by eye. The map doesn't have all the provincial capitals on it, but it's easy to see that MDS has placed the ones that are there in the right places (use a piece of ruled tracing paper to check).*

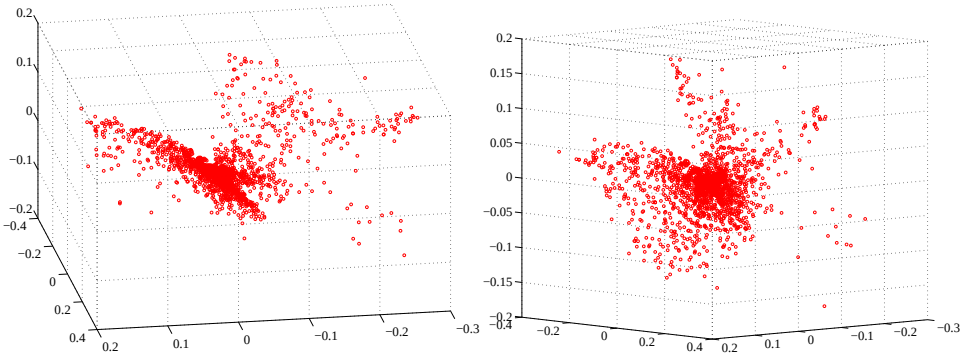

FIGURE 10.17: *Two views of the spectral data of section 10.3.1, plotted as a scatter plot by applying principal coordinate analysis to obtain a 3D set of points. Notice that the data spreads out in 3D, but seems to lie on some structure; it certainly isn't a single blob. This suggests that further investigation would be fruitful.*

measuring instrument doesn't make certain kinds of measurement; or perhaps there are physical processes that prevent the data from spreading out over the space.

Our algorithm has one really interesting property. In some cases, we do not actually know the datapoints as vectors. Instead, we *just* know distances between the datapoints. This happens often in the social sciences, but there are important cases in computer science as well. As a rather contrived example, one could survey people about breakfast foods (say, eggs, bacon, cereal, oatmeal, pancakes, toast, muffins, kippers and sausages for a total of 9 items). We ask each person to rate the similarity of each pair of distinct items on some scale. We advise people that similar

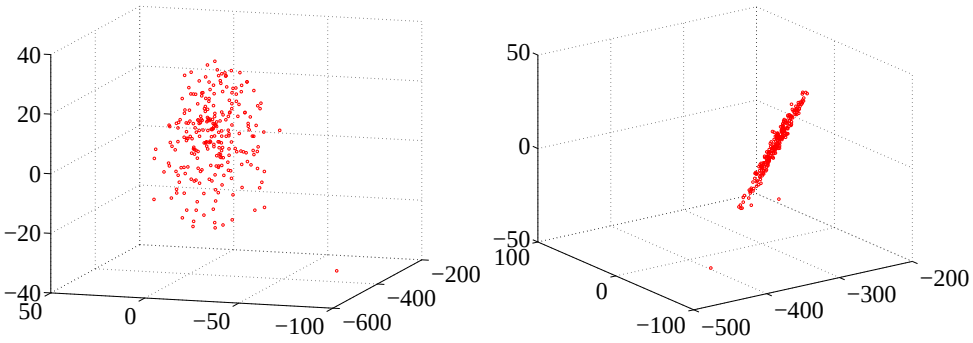

FIGURE 10.18: *Two views of a multidimensional scaling to three dimensions of the height-weight dataset. Notice how the data seems to lie in a flat structure in 3D, with one outlying data point. This means that the distances between data points can be (largely) explained by a 2D representation.*

items are ones where, if they were offered both, they would have no particular preference; but, for dissimilar items, they would have a strong preference for one over the other. The scale might be "very similar", "quite similar", "similar", "quite dissimilar", and "very dissimilar" (scales like this are often called **Likert scales**). We collect these similarities from many people for each pair of distinct items, and then average the similarity over all respondents. We compute distances from the similarities in a way that makes very similar items close and very dissimilar items distant. Now we have a table of distances between items, and can compute a $\mathcal{V}$ and produce a scatter plot. This plot is quite revealing, because items that most people think are easily substituted appear close together, and items that are hard to substitute are far apart. The neat trick here is that we did not start with a $\mathcal{X}$, but with just a set of distances; but we were able to associate a vector with "eggs", and produce a meaningful plot.

10.5 EXAMPLE: UNDERSTANDING HEIGHT AND WEIGHT

Recall the height-weight data set of section **??** (from http://www2.stetson.edu/~jrasp/data.htm; look for bodyfat.xls at that URL). This is, in fact, a 16-dimensional dataset. The entries are (in this order): *bodyfat; density; age; weight; height; adiposity; neck; chest; abdomen; hip; thigh; knee; ankle; biceps; forearm; wrist.* We know already that many of these entries are correlated, but it's hard to grasp a 16 dimensional dataset in one go. The first step is to investigate with a multidimensional scaling.

Figure **??** shows a multidimensional scaling of this dataset down to three dimensions. The dataset seems to lie on a (fairly) flat structure in 3D, meaning that inter-point distances are relatively well explained by a 2D representation. Two points seem to be special, and lie far away from the flat structure. The structure isn't perfectly flat, so there will be small errors in a 2D representation; but it's clear that a lot of dimensions are redundant. Figure 10.19 shows a 2D representation of these points. They form a blob that is stretched along one axis, and there is no sign

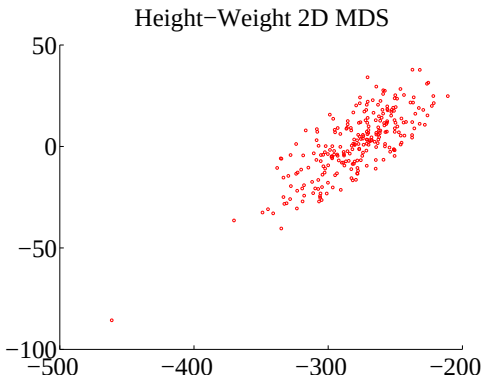

FIGURE 10.19: *A multidimensional scaling to two dimensions of the height-weight dataset. One data point is clearly special, and another looks pretty special. The data seems to form a blob, with one axis quite a lot more important than another.*

of multiple blobs. There's still at least one special point, which we shall ignore but might be worth investigating further. The distortions involved in squashing this dataset down to 2D seem to have made the second special point less obvious than it was in figure **??**.

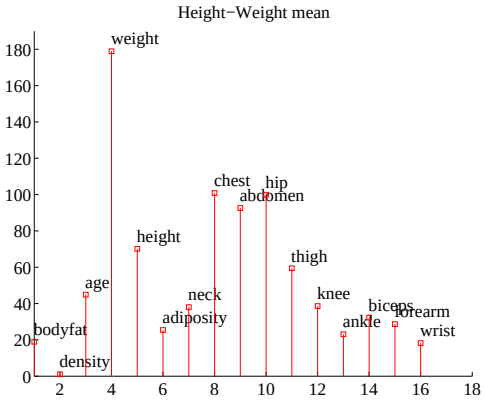

FIGURE 10.20: *The mean of the bodyfat.xls dataset. Each component is likely in a different unit (though I don't know the units), making it difficult to plot the data without being misleading. I've adopted one solution here, by plotting a stem plot. You shouldn't try to compare the values to one another. Instead, think of this plot as a compact version of a table.*

The next step is to try a principal component analysis. Figure 10.20 shows the mean of the dataset. The components of the dataset have different units, and shouldn't really be compared. But it is difficult to interpret a table of 16 numbers, so I have plotted the mean as a stem plot. Figure 10.21 shows the eigenvalues of the covariance for this dataset. Notice how one dimension is very important, and

after the third principal component, the contributions become small. Of course, I could have said "fourth", or "fifth", or whatever — the precise choice depends on how small a number you think is "small".

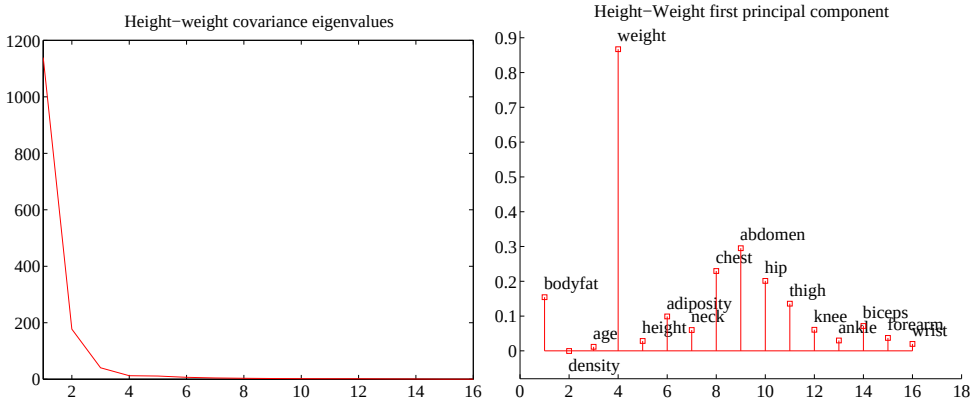

FIGURE 10.21: *On the* **left**, *the eigenvalues of the covariance matrix for the bodyfat data set. Notice how fast the eigenvalues fall off; this means that most principal components have very small variance, so that data can be represented well with a small number of principal components. On the* **right**, *the first principal component for this dataset, plotted using the same convention as for figure 10.20.*

Figure 10.21 also shows the first principal component. The eigenvalues justify thinking of each data item as (roughly) the mean plus some weight times this principal component. From this plot you can see that data items with a larger value of *weight* will also have larger values of most other measurements, except *age* and *density*. You can also see how much larger; if the weight goes up by 8.5 units, then the abdomen will go up by 3 units, and so on. This explains the main variation in the dataset.

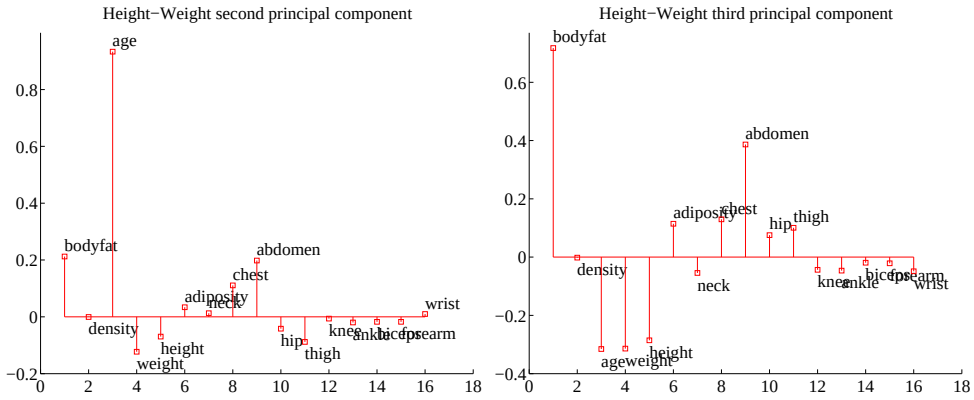

FIGURE 10.22: *On the* **left**, *the second principal component, and on the* **right** *the third principal component of the height-weight dataset.*

In the rotated coordinate system, the components are not correlated, and they have different variances (which are the eigenvalues of the covariance matrix). You can get some sense of the data by adding these variances; in this case, we get 1404. This means that, in the translated and rotated coordinate system, the average data point is about $37 = \sqrt{1404}$ units away from the center (the origin). Translations and rotations do not change distances, so the average data point is about 37 units from the center in the original dataset, too. If we represent a datapoint by using the mean and the first three principal components, there will be some error. We can estimate the average error from the component variances. In this case, the sum of the first three eigenvalues is 1357, so the mean square error in representing a datapoint by the first three principal components is $\sqrt{(1404 - 1357)}$, or 6.8. The relative error is $6.8/37 = 0.18$. Another way to represent this information, which is more widely used, is to say that the first three principal components explain all but $(1404 - 1357)/1404 = 0.034$, or 3.4% of the variance; notice that this is the square of the relative error, which will be a much smaller number.

All this means that explaining a data point as the mean and the first three principal components produces relatively small errors. Figure 10.23 shows the second and third principal component of the data. These two principal components suggest some further conclusions. As *age* gets larger, *height* and *weight* get slightly smaller, but the weight is redistributed; *abdomen* gets larger, whereas *thigh* gets smaller. A smaller effect (the third principal component) links *bodyfat* and *abdomen*. As *bodyfat* goes up, so does *abdomen*.

10.6 YOU SHOULD

10.6.1 remember these definitions:

10.6.2 remember these terms:

10.6.3 remember these facts:

10.6.4 remember these procedures:

10.6.5 be able to:

- Create, plot and interpret the first few principal components of a dataset.
- Compute the error resulting from ignoring some principal components.

PROBLEMS

Summaries

10.1. You have a dataset $\{\mathbf{x}\}$ of N vectors, $\mathbf{x}_i$, each of which is d-dimensional. We will consider a linear function of this dataset. Write $\mathbf{a}$ for a constant vector; then the value of this linear function evaluated on the i'th data item is $\mathbf{a}^T\mathbf{x}_i$. Write $f_i = \mathbf{a}^T\mathbf{x}_i$. We can make a new dataset $\{f\}$ out of the values of this linear function.

(a) Show that $\mathsf{mean}(\{f\}) = \mathbf{a}^T\mathsf{mean}(\{\mathbf{x}\})$ (easy).

(b) Show that $\mathsf{var}(\{f\}) = \mathbf{a}^T\mathsf{Covmat}(\{\mathbf{x}\})\mathbf{a}$ (harder, but just push it through the definition).

(c) Assume the dataset has the special property that there exists some $\mathbf{a}$ so that $\mathbf{a}^T\mathsf{Covmat}(\{\mathbf{x}\})\mathbf{a}$. Show that this means that the dataset lies on a hyperplane.

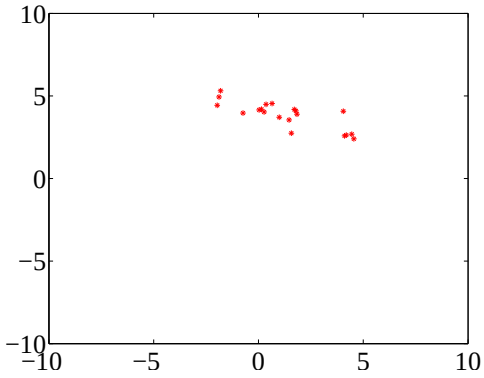

FIGURE 10.23: *Figure for the question*

10.2. On Figure 10.23, mark the mean of the dataset, the first principal component, and the second principal component.

10.3. You have a dataset $\{\mathbf{x}\}$ of N vectors, $\mathbf{x}_i$, each of which is d-dimensional. Assume that $\mathsf{Covmat}(\{\mathbf{x}\})$ has one non-zero eigenvalue. Assume that $\mathbf{x}_1$ and $\mathbf{x}_2$ do not have the same value.

(a) Show that you can choose a set of t_i so that you can represent *every* data item $\mathbf{x}_i$ *exactly*

$$\mathbf{x}_i = \mathbf{x}_1 + t_i(\mathbf{x}_2 - \mathbf{x}_1).$$

(b) Now consider the dataset of these t values. What is the relationship between (a) $\mathsf{std}(t)$ and (b) the non-zero eigenvalue of $\mathsf{Covmat}(\{\mathbf{x}\})$? Why?

PROGRAMMING EXERCISES

10.4. Obtain the iris dataset from the UC Irvine machine learning data repository at http://https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data.

(a) Plot a scatterplot matrix of this dataset, showing each species with a different marker.

(b) Now obtain the first two principal components of the data. Plot the data on those two principal components alone, again showing each species with a different marker. Has this plot introduced significant distortions? Explain

10.5. Take the wine dataset from the UC Irvine machine learning data repository at https://archive.ics.uci.edu/ml/datasets/Wine.

(a) Plot the eigenvalues of the covariance matrix in sorted order. How many principal components should be used to represent this dataset? Why?

(b) Construct a stem plot of each of the first 3 principal components (i.e. the eigenvectors of the covariance matrix with largest eigenvalues). What do you see?

(c) Compute the first two principal components of this dataset, and project it onto those components. Now produce a scatter plot of this two dimensional dataset, where data items of class 1 are plotted as a '1', class 2 as a '2', and so on.

10.6. Take the wheat kernel dataset from the UC Irvine machine learning data repository at http://archive.ics.uci.edu/ml/datasets/seeds. Compute the first two principal components of this dataset, and project it onto those components.

(a) Produce a scatterplot of this projection. Do you see any interesting phenomena?

(b) Plot the eigenvalues of the covariance matrix in sorted order. How many principal components should be used to represent this dataset? why?

10.7. The UC Irvine machine learning data repository hosts a collection of data on breast cancer diagnostics, donated by Olvi Mangasarian, Nick Street, and William H. Wolberg. You can find this data at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic). For each record, there is an id number, 10 continuous variables, and a class (benign or malignant). There are 569 examples. Separate this dataset randomly into 100 validation, 100 test, and 369 training examples. Plot this dataset on the first three principal components, using different markers for benign and malignant cases. What do you see?

10.8. The UC Irvine Machine Learning data archive hosts a dataset of measurements of abalone at http://archive.ics.uci.edu/ml/datasets/Abalone. Compute the principal components of all variables except Sex. Now produce a scatter plot of the measurements projected onto the first two principal components, plotting an "m" for male abalone, an "f" for female abalone and an "i" for infants. What do you see?

10.9. Choose a state. For the 15 largest cities in your chosen state, find the distance between cities and the road mileage between cities. These differ because of the routes that roads take; you can find these distances by careful use of the internet. Prepare a map showing these cities on the plane using principal coordinate analysis for each of these two distances. How badly does using the road network distort to make a map distort the state? Does this differ from state to state? Why?

10.10. CIFAR-10 is a dataset of 32x32 images in 10 categories, collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. It is often used to evaluate machine learning algorithms. You can download this dataset from https://www.cs.toronto.edu/~kriz/cifar.html.

(a) For each category, compute the mean image and the first 20 principal components. Plot the error resulting from representing the images of each category using the first 20 principal components against the category.

(b) Compute the distances between mean images for each pair of classes. Use principal coordinate analysis to make a 2D map of the means of each categories. For this exercise, compute distances by thinking of the images as vectors.

(c) Here is another measure of the similarity of two classes. For class A and class B, define $E(A \to B)$ to be the average error obtained by representing all the images of class A using the mean of class A and the first 20 principal components of class B. Now define the similarity between classes to be $(1/2)(E(A \to B)+E(B \to A))$. Use principal coordinate analysis to make a 2D map of the classes. Compare this map to the map in the previous exercise – are they different? why?

CHAPTER 11

Learning to Classify

A **classifier** is a procedure that accepts a set of features and produces a class label for them. There could be two, or many, classes. Classifiers are immensely useful, and find wide application, because many problems are naturally classification problems. For example, if you wish to determine whether to place an advert on a web-page or not, you would use a classifier (i.e. look at the page, and say yes or no according to some rule). As another example, if you have a program that you found for free on the web, you would use a classifier to decide whether it was safe to run it (i.e. look at the program, and say yes or no according to some rule). As yet another example, credit card companies must decide whether a transaction is good or fraudulent.

All these examples are two class classifiers, but in many cases it is natural to have more classes. You can think of sorting laundry as applying a multi-class classifier. You can think of doctors as complex multi-class classifiers: a doctor accepts a set of features (your complaints, answers to questions, and so on) and then produces a response which we can describe as a class. The grading procedure for any class is a multi-class classifier: it accepts a set of features — performance in tests, homeworks, and so on — and produces a class label (the letter grade).

Definition: 11.1 *Classifier*

A classifier is a procedure that accepts a set of features and produces a label.

11.1 CLASSIFICATION: THE BIG IDEAS

Classifiers are built by taking a set of labeled examples and using them to come up with a procedure that assigns a label to any new example. In the general problem, we have a training dataset of examples $(\mathbf{x}_i, y_i)$. For the i'th example, $\mathbf{x}_i$ represents the values taken by a collection of features. In the simplest case, $\mathbf{x}_i$ would be a vector of real numbers. In some cases, $\mathbf{x}_i$ will contain categorical data or even unobserved values. Although $\mathbf{x}_i$ isn't guaranteed to be a vector, it's usually referred to as a **feature vector**. The y_i are labels giving the type of the object that generated the example. We will use the labelled examples to come up with a classifier.

11.1.1 The Error Rate

We need to summarize the behavior of a classifier, so we can choose one that behaves well. Two values that are widely used are the **error** or **total error rate** (the percentage of classification attempts that gave the wrong answer) and the **accuracy** (the percentage of classification attempts that give the right answer).

For most practical cases, the best choice of classifier is guaranteed to make mistakes. As an example, consider an alien who tries to classify humans into male and female, using only height as a feature. However the alien's classifier uses that feature, it will make mistakes. This is because the classifier must choose, for each value of height, whether to label the humans with that height male or female. But for the vast majority of heights, there are some males and some females with that height, and so the alien's classifier must make some mistakes.

The example shows we are not guaranteed that a particular feature vector $\mathbf{x}$ always appears with the same label. We should think of labels as appearing with some probability conditioned on the observations, $P(y|\mathbf{x})$. If we knew this (which we seldom do), we could use it to compute the expected error rate for any particular rule. If there are parts of the feature space where $P(\mathbf{x})$ is relatively large (so we expect to see observations of that form) *and* where $P(y|\mathbf{x})$ has relatively large values for more than one label, even the best possible classifier will have a high error rate. The minimum expected error rate obtained with the best possible classifier applied to a particular problem is known as the **Bayes risk** for that problem. In most cases, it is hard to know what the Bayes risk is, because to compute it requires access to information that isn't available (the posterior probability of a class conditioned on the feature vector, for one thing).

11.1.2 Overfitting

Choosing and evaluating a classifier takes some care. What matters is not the classifier's error on the training data, but the error on future test data. For example, we might use a set of credit card records to make a classifier that predicts whether a transaction is fraudulent or not. This classifier is only useful if we can use it successfully on future examples, *where we might never know the true label.*

Classifiers that have small training error might not have small test error. One example of this problem is the (silly) classifier that takes any data point and, if it is the same as a point in the training set, emits the class of that point and otherwise chooses randomly between the classes. This has zero training error, but might have large test error.

Test error is usually worse than training error, because the classification procedure is chosen to do well on the training data. This effect is sometimes called **overfitting**. Other names include **selection bias**, because the training data has been selected and so isn't exactly like the test data, and **generalizing badly**, because the classifier must generalize from the training data to the test data. The effect occurs because the classifier has been trained to perform well *on the training dataset*, and the training dataset is not the same as the test dataset. First, it is quite likely smaller. Second, it might be biased through a variety of accidents. This means that small training error may have to do with quirks of the training dataset that don't occur in other sets of examples. One consequence of overfitting is that

classifiers should always be evaluated on data that was not used in training.

> **Remember this:** *Classifiers should always be evaluated on data that was not used in training.*

11.1.3 Cross-Validation

Now assume that we want to estimate the error rate of the classifier on test data. We cannot estimate the error rate of the classifier using data that was used to train the classifier, because the classifier has been trained to do well on that data, which will mean our error rate estimate will be too low. An alternative is to separate out some training data to form a **validation set**, then train the classifier on the rest of the data, and evaluate on the validation set. This has the difficulty that the classifier will not be the best estimate possible, because we have left out some training data when we trained it. This issue can become a significant nuisance when we are trying to tell which of a set of classifiers to use — did the classifier perform poorly on validation data because it is not suited to the problem representation or because it was trained on too little data?

We can resolve this problem with **cross-validation**, which involves repeatedly: splitting data into training and validation sets uniformly and at random, training a classifier on the training set, evaluating it on the validation set, and then averaging the error over all splits. This allows an estimate of the likely future performance of a classifier, at the expense of substantial computation. You should notice that cross-validation, in some sense, looks at the sensitivity of the classifier to a change in the training set. The most usual form of this algorithm involves omitting single items from the dataset and is known as **leave-one-out cross-validation**.

11.1.4 Is the Classifier Working Well?

The error rate of a classifier is not that meaningful on its own. There might be some other classifier with a better error rate. Furthermore, there might be some structure in the errors that suggests ways to improve the classifier. The simplest comparison is to a know-nothing strategy. Imagine classifying the data without using the feature vector at all — how well does this strategy do? If each of the C classes occurs with the same frequency, then it's enough to label the data by choosing a label uniformly and at random, and the error rate is $1 - 1/C$. If one class is more common than the others, the lowest error rate is obtained by labelling everything with that class. This comparison is often known as **comparing to chance**. Further comparisons can be obtained by building several different classifiers, and seeing which has the lowest error rate.

It is very common to deal with data where there are only two labels. You should keep in mind this means the highest possible error rate is 50% — if you have

a classifier with a higher error rate, you can improve it by switching the outputs. If one class is much more common than the other, training becomes more complicated because the best strategy – labelling everything with the common class – becomes hard to beat.

Analyzing performance involves looking at more than just the error rate. For a two-class classifier and a 0-1 loss function, one can report the **false positive rate** (the percentage of negative test data that was classified positive) and the **false negative rate** (the percentage of positive test data that was classified negative). Note that it is important to provide both, because a classifier with a low false positive rate tends to have a high false negative rate, and vice versa. As a result, you should be suspicious of reports that give one number but not the other. Alternative numbers that are reported sometimes include the **sensitivity** (the percentage of true positives that are classified positive) and the **specificity** (the percentage of true negatives that are classified negative).

Evaluating a multi-class classifier is more complex than evaluating a binary classifier. A multi-class classifier can make many more kinds of mistake than a binary classifier can. If the total error rate is low enough, or the accuracy is high enough, there's not much to worry about. But if it's not, you can look at the **class confusion matrix** to see what's going on.

	Predict 0	Predict 1	Predict 2	Predict 3	Predict 4	Class error
True 0	151	7	2	3	1	7.9%
True 1	32	5	9	9	0	91%
True 2	10	9	7	9	1	81%
True 3	6	13	9	5	2	86%
True 4	2	3	2	6	0	100%

TABLE 11.1: The class confusion matrix for a multiclass classifier. Further details about the dataset and this example appear in worked example 11.3.

Table 11.1 gives an example. This is a class confusion matrix from a classifier built on a dataset where one tries to predict the degree of heart disease from a collection of physiological and physical measurements. There are five classes ($0 \ldots 4$). The i, j'th cell of the table shows the number of data points of true class i that were classified to have class j. As I find it hard to recall whether rows or columns represent true or predicted classes, I have marked this on the table. For each row, there is a **class error rate**, which is the percentage of data points of that class that were misclassified. The first thing to look at in a table like this is the diagonal; if the largest values appear there, then the classifier is working well. This clearly isn't what is happening for table 11.1. Instead, you can see that the method is very good at telling whether a data point is in class 0 or not (the class error rate is rather small), but cannot distinguish between the other classes. This is a strong hint that the data can't be used to draw the distinctions that we want. It might be a lot better to work with a different set of classes.

11.2 CLASSIFYING WITH NEAREST NEIGHBORS

Assume we have a set of N example points $\mathbf{x}_i$. These points come with their class labels, which we write as y_i; thus, our dataset can be written as

$$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}.$$

We wish to predict the label y for any point $\mathbf{x}$. Generally we expect that if two points are close enough, then they will have the same label. This suggests a really effective strategy. If you want to classify a data item (sometimes called a query point), find the closest example, and report the class of that example. Alternatively, you could find the closest k examples, and vote.

How well can we expect this strategy to work? A precise analysis would take us way out of our way, but simple reasoning is informative. Assume we have only two labels to deal with, and that we use only a single nearest neighbor. Around each example point $\mathbf{x}_i$ is a cell of points to which our classifier gives the same label as $\mathbf{x}_i$. If we have enough examples, most of these cells are small. In places where $P(y = 1|\mathbf{x})$ is high, almost every example will have the label 1, and all the corresponding cells will have that label, too, and so the error rate will be low. In regions where $P(y = 1|\mathbf{x})$ is about the same as $P(y = -1|\mathbf{x})$, there will be about as many examples (and so, cells) with label 1 as with label -1. This means that in these regions the classifier will tend to make mistakes more often, as it should. Using a great deal more of this kind of reasoning, nearest neighbors can be shown to produce an error that is no worse than twice the best error rate, if the method has enough examples. There is no prospect of seeing enough examples in practice for this result to apply.

One important generalization is to find the k nearest neighbors, then choose a label from those. A (k, l) nearest neighbor classifier finds the k example points closest to the point being considered, and classifies this point with the class that has the highest number of votes, as long as this class has more than l votes (otherwise, the point is classified as unknown). In practice, one seldom uses more than three nearest neighbors. Finding the k nearest points for a particular query can be difficult, and Section 14.3 reviews this point.

There are three practical difficulties in building nearest neighbor classifiers. You need a lot of labelled examples. You need to be able to find the nearest neighbors for your query point. And you need to use a sensible choice of distance. For features that are obviously of the same type, such as lengths, the usual metric may be good enough. But what if one feature is a length, one is a color, and one is an angle? One possibility is to whiten the features (section **??**). This may be hard if the dimension is so large that the covariance matrix is hard to estimate. It is almost always a good idea to scale each feature independently so that the variance of each feature is the same, or at least consistent; this prevents features with very large scales dominating those with very small scales. Notice that nearest neighbors (fairly obviously) doesn't like categorical data. If you can't give a clear account of how far apart two things are, you shouldn't be doing nearest neighbors. It is possible to fudge this point a little, by (say) putting together a distance between the levels of each factor, but it's probably unwise.

Nearest neighbors is wonderfully flexible about the labels the classifier pre-

dicts. Nothing changes when you go from a two-class classifier to a multi-class classifier.

Cross-validation is straightforward with a nearest neighbor classifier. Split the labelled training data into two pieces, a (typically large) training set and a (typically small) validation set. Now take each element of the validation set and label it with the label of the closest element of the training set. Compute the fraction of these effects that produce an error (the true label and predicted labels differ). Now repeat this for a different split, and average the errors over splits. With care, the code you'll write is shorter than this description.

Worked example 11.1 *Classifying using nearest neighbors*

Build a nearest neighbor classifier to classify the digit data originally constructed by Yann Lecun. You can find it at several places. The original dataset is at http://yann.lecun.com/exdb/mnist/. The version I used was used for a Kaggle competition (so I didn't have to decompress Lecun's original format). I found it at http://www.kaggle.com/c/digit-recognizer.

Solution: As you'd expect, R has nearest neighbor code that seems quite good (I haven't had any real problems with it, at least). There isn't really all that much to say about the code. I used the R FNN package. There is sample code in listing **??**. I trained on 1000 of the 42000 examples, so you could see how in the code. I tested on the next 200 examples. For this (rather small) case, I found the following class confusion matrix

	0	1	2	3	4	5	6	7	8	9
0	12	0	0	0	0	0	0	0	0	0
1	0	20	4	1	0	1	0	2	2	1
2	0	0	20	1	0	0	0	0	0	0
3	0	0	0	12	0	0	0	0	4	0
4	0	0	0	0	18	0	0	0	1	1
5	0	0	0	0	0	19	0	0	1	0
6	1	0	0	0	0	0	18	0	0	0
7	0	0	1	0	0	0	0	19	0	2
8	0	0	1	0	0	0	0	0	16	0
9	0	0	0	2	3	1	0	1	1	14

There are no class error rates here, because I couldn't recall the magic line of R to get them. However, you can see the classifier works rather well for this case.

Remember this: *Nearest neighbor classifiers are often very effective. They can predict any kind of label. You do need to be careful to have enough data, and to have a meaningful distance function.*

11.3 CLASSIFYING WITH NAIVE BAYES

One straightforward source of a classifier is a probability model. For the moment, assume we know $p(y|\mathbf{x})$ for our data. Assume also that all errors in classification are equally important. Then the following rule produces smallest possible expected classification error rate:

> For a test example $\mathbf{x}$, report the class y that has the highest value of $p(y|\mathbf{x})$. If the largest value is achieved by more than one class, choose randomly from that set of classes.

Usually, we do not have $p(y|\mathbf{x})$. If we have $p(\mathbf{x}|y)$ (often called either a **likelihood** or **class conditional probability**), and $p(y)$ (often called a **prior**) then we can use Bayes' rule to form

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

(the **posterior**). This isn't much help in this form, but write x_j for the j'th component of $\mathbf{x}$. Now *assume* that features are conditionally independent conditioned on the class of the data item. Our assumption is

$$p(\mathbf{x}|y) = \prod_i p(x_i|y).$$

It is very seldom the case that this assumption is true, but it turns out to be fruitful to pretend that it is. This assumption means that

$$\begin{aligned} p(y|\mathbf{x}) &= \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})} \\ &= \frac{\prod_i p(x_i|y)p(y)}{p(\mathbf{x})} \\ &\propto \prod_i p(x_i|y)p(y). \end{aligned}$$

Now to make a decision, we need to choose the class that has the largest value of $p(y|\mathbf{x})$. In turn, this means we need only know the posterior values up to scale at $\mathbf{x}$, so we don't need to estimate $p(\mathbf{x})$. In the case of where all errors have the same cost, this yields the rule

$$\text{choose } y \text{ such that } \textstyle\prod_i p(x_i|y)p(y) \text{ is largest.}$$

We still need models for $p(x_i|y)$ for each x_i. It turns out that simple parametric models work really well here. For example, one could fit a normal distribution to each x_i in turn, for each possible value of y, using the training data. The logic of the measurements might suggest other distributions, too. If one of the x_i's was a count, we might fit a Poisson distribution. If it was a 0-1 variable, we might fit a

Bernoulli distribution. If it was a numeric variable that took one of several values, then we might use a multinomial model.

Naive bayes is particularly good when there are a large number of features, but there are some things to be careful about. You can't actually multiply a large number of probabilities and expect to get an answer that a floating point system thinks is different from zero. Instead, you should add the log probabilities. Notice that the logarithm function has one nice property: it is monotonic, meaning that $a > b$ is equivalent to $\log a > \log b$. In turn, this means you don't need to exponentiate when you've added up the log probabilities. If, for some reason, you need to know the values of the probabilities, you should not just add up all the log probabilities then exponentiate, or else you will find that each class has a posterior probability of zero. Instead, subtract the largest log from all the others, then exponentiate; you will obtain a vector proportional to the class probabilities, where the largest element has the value 1.

The usual way to find a model of $p(y)$ is to count the number of training examples in each class, then divide by the number of classes. If there are some classes with very little data, then the classifier is likely to work poorly, because you will have trouble getting reasonable estimates of the parameters for the $p(x_i|y)$.

Worked example 11.2 *Classifying breast tissue samples*

The "breast tissue" dataset at https://archive.ics.uci.edu/ml/datasets/Breast+Tissue contains measurements of a variety of properties of six different classes of breast tissue. Build and evaluate a naive bayes classifier to distinguish between the classes automatically from the measurements.

Solution: The main difficulty here is finding appropriate packages, understanding their documentation, and checking they're right, unless you want to write the source yourself (which really isn't all that hard). I used the R package `caret` to do train-test splits, cross-validation, etc. on the naive bayes classifier in the R package `klaR`. I separated out a test set randomly (approx 20% of the cases for each class, chosen at random), then trained with cross-validation on the remainder. The class-confusion matrix on the test set was:

Prediction	adi	car	con	fad	gla	mas
adi	2	0	0	0	0	0
car	0	3	0	0	0	1
con	2	0	2	0	0	0
fad	0	0	0	0	1	0
gla	0	0	0	0	2	1
mas	0	1	0	3	0	1

which is fairly good. The accuracy is 52%. In the training data, the classes are nearly balanced and there are six classes, meaning that chance is about 17%. These numbers, and the class-confusion matrix, will vary with test-train split. I have not averaged over splits, which would be the next thing.

Remember this: *Naive bayes classifiers are straightforward to build, and very effective. Dealing with missing data is easy. Experience has shown they are particularly effective at high dimensional data.*

11.3.1 Missing Data

Missing data occurs when some values in the training data are unknown. This can happen in a variety of ways. Someone didn't record the value; someone recorded it incorrectly, and you know the value is wrong but you don't know what the right one is; the dataset was damaged in storage or transmission; instruments failed; and so on. This is quite typical of data where the feature values are obtained by measuring effects in the real world. It's much less common where the feature values are computed from signals – for example, when one tries to classify digital images, or sound recordings.

Missing data can be a serious nuisance in classification problems, because many methods cannot handle incomplete feature vectors. If there are relatively few incomplete feature vectors, one could just drop them and proceed. Naive bayes is rather good at handling data where there are many incomplete feature vectors in quite a simple way. For example, assume for some i, we wish to fit $p(x_i|y)$ with a normal distribution. We need to estimate the mean and standard deviation of that normal distribution (which we do with maximum likelihood, as one should). If not every example has a known value of x_i, this really doesn't matter; we simply omit the unknown number from the estimate. Write $x_{i,j}$ for the value of x_i for the j'th example. To estimate the mean, we form

$$\frac{\sum_{j \in \text{cases with known values}} x_{i,j}}{\text{number of cases with known values}}$$

and so on.

Dealing with missing data during classification is easy, too. We need to look for the y that produces the largest value of $\sum_i \log p(x_i|y)$. We can't evaluate $p(x_i|y)$ if the value of that feature is missing - but it is missing for each class. We can just leave that term out of the sum, and proceed. This procedure is fine if data is missing as a result of "noise" (meaning that the missing terms are independent of class). If the missing terms depend on the class, there is much more we could do — for example, we might build a model of the class-conditional density of missing terms.

Notice that if some values of a discrete feature x_i don't appear for some class, you could end up with a model of $p(x_i|y)$ that had zeros for some values. This almost inevitably leads to serious trouble, because it means your model states you cannot ever observe that value for a data item of that class. This isn't a safe property: it is hardly ever the case that not observing something means you cannot observe it. A simple, but useful, fix is to add one to all small counts. More sophisticated methods are available, but well beyond our scope.

11.4 THE SUPPORT VECTOR MACHINE

Assume we have a set of N example points $\mathbf{x}_i$ that belong to two classes, which we indicate by 1 and -1. These points come with their class labels, which we write as y_i; thus, our dataset can be written as

$$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}.$$

We wish to predict the sign of y for any point $\mathbf{x}$. We will use a linear classifier, so that for a new data item $\mathbf{x}$, we will predict

$$\mathsf{sign}\,(\mathbf{a} \cdot \mathbf{x} + b)$$

and the particular classifier we use is given by our choice of $\mathbf{a}$ and b.

You should think of $\mathbf{a}$ and b as representing a hyperplane, given by the points where $\mathbf{a} \cdot \mathbf{x} + b = 0$. Notice that the magnitude of $\mathbf{a} \cdot \mathbf{x} + b$ grows as the point $\mathbf{x}$ moves further away from the hyperplane. This hyperplane separates the positive data from the negative data, and is an example of a **decision boundary**. When a point crosses the decision boundary, the label predicted for that point changes. All classifiers have decision boundaries. Searching for the decision boundary that yields the best behavior is a fruitful strategy for building classifiers.

Example: 11.1 *A linear model with a single feature*

Assume we use a linear model with one feature. Then the model has the form $y_i^{(p)} = \text{sign}(ax_i + b)$. For any particular example which has the feature value x^*, this means we will test whether x^* is larger than, or smaller than, $-b/a$.

Example: 11.2 *A linear model with two features*

Assume we use a linear model with two features. Then the model has the form $y_i^{(p)} = \text{sign}(\mathbf{a}^T\mathbf{x}_i + b)$. The sign changes along the line $\mathbf{a}^T\mathbf{x} + b = 0$. You should check that this is, indeed, a line. On one side of this line, the model makes positive predictions; on the other, negative. Which side is which can be swapped by multiplying $\mathbf{a}$ and b by -1.

This family of classifiers may look bad to you, and it is easy to come up with examples that it misclassifies badly. In fact, the family is extremely strong. First, it is easy to estimate the best choice of rule for very large datasets. Second, linear classifiers have a long history of working very well in practice on real data. Third, linear classifiers are fast to evaluate.

In practice, examples that are classified badly by the linear rule usually are classified badly because there are too few features. Remember the case of the alien

who classified humans into male and female by looking at their heights; if that alien had looked at their chromosomes as well, the error rate would be small. In practical examples, experience shows that the error rate of a poorly performing linear classifier can usually be improved by adding features to the vector $\mathbf{x}$.

11.4.1 Choosing a Classifier with the Hinge Loss

We will choose $\mathbf{a}$ and b by choosing values that minimize a cost function. We will adopt a cost function of the form:

$$\text{Training error cost} + \text{penalty term}.$$

For the moment, we will ignore the penalty term and focus on the training error cost. Write

$$\gamma_i = \mathbf{a}^T\mathbf{x}_i + b$$

for the value that the linear function takes on example i. Write $C(\gamma_i, y_i)$ for a function that compares γ_i with y_i. The training error cost will be of the form

$$(1/N)\sum_{i=1}^{N} C(\gamma_i, y_i).$$

A good choice of C should have some important properties. If γ_i and y_i have different signs, then C should be large, because the classifier will make the wrong prediction for this training example. Furthermore, if γ_i and y_i have different signs *and* γ_i has large magnitude, then the classifier will very likely make the wrong prediction for test examples that are close to $\mathbf{x}_i$. This is because the magnitude of $(\mathbf{a} \cdot \mathbf{x} + b)$ grows as $\mathbf{x}$ gets further from the decision boundary. So C should get larger as the magnitude of γ_i gets larger in this case.

If γ_i and y_i have the same signs, but γ_i has small magnitude, then the classifier will classify $\mathbf{x}_i$ correctly, but might not classify points that are nearby correctly. This is because a small magnitude of γ_i means that $\mathbf{x}_i$ is close to the decision boundary. So C should not be zero in this case. Finally, if γ_i and y_i have the same signs and γ_i has large magnitude, then C can be zero because $\mathbf{x}_i$ is on the right side of the decision boundary and so are all the points near to $\mathbf{x}_i$.

The choice

$$C(y_i, \gamma_i) = \max(0, 1 - y_i\gamma_i)$$

has these properties. If $y_i\gamma_i > 1$ (so the classifier predicts the sign correctly *and* $\mathbf{x}_i$ is far from the boundary) there is no cost. But in any other case, there is a cost. The cost rises if $\mathbf{x}_i$ moves toward the decision boundary from the correct side, and grows linearly as $\mathbf{x}_i$ moves further away from the boundary on the wrong side (Figure 11.1). This means that minimizing the loss will encourage the classifier to (a) make strong positive (or negative) predictions for positive (or negative) examples and (b) for examples it gets wrong, make the most positive (negative) prediction that it can. This choice is known as the **hinge loss**. A linear classifier trained with the hinge loss is known as a **support vector machine** or **SVM**.

The hinge loss has one odd property. Assume that the pair $\mathbf{a}$, b correctly classifies all training examples, so that $y_i(\mathbf{a}^T x_i + b) > 0$. Then we can always

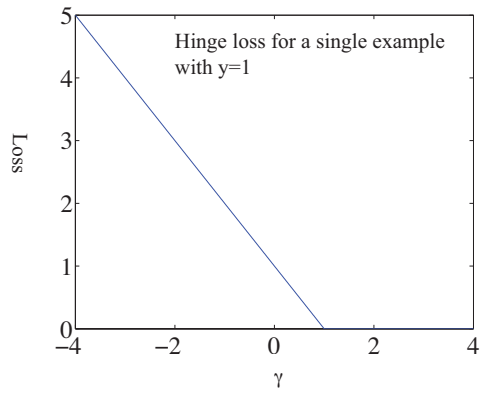

FIGURE 11.1: *The hinge loss, plotted for the case $y_i = 1$. The horizontal variable is the $\gamma_i = \mathbf{a} \cdot \mathbf{x}_i + b$ of the text. Notice that giving a strong negative response to this positive example causes a loss that grows linearly as the magnitude of the response grows. Notice also that giving an insufficiently positive response also causes a loss. Giving a strongly positive response is free.*

ensure that the hinge loss for the dataset is zero, by scaling $\mathbf{a}$ and b, because you can choose a scale so that $y_j(\mathbf{a}^T x_j + b) > 1$ for *every* example index j. This scale hasn't changed the result of the classification rule on the training data. But it should worry you, because it means we can't choose the classifier parameters uniquely.

Now think about future examples. We don't know what their feature values will be, and we don't know their labels. But we do know that the hinge loss for an example with feature vector $\mathbf{x}$ and unknown label y will be $\max(0, 1 - y\left[\mathbf{a}^T\mathbf{x} + b\right])$. Imagine we classify this example wrongly. If $\|\mathbf{a}\|$ is small, then at least the hinge loss will be small. By this argument, we would like to achieve a small value of the hinge loss using a $\mathbf{a}$ that has small length. It is much simpler to use the squared length (i.e $\mathbf{a}^T\mathbf{a} = \|\mathbf{a}\|^2$) than the length, and doing so is now usual.

The way to do this is to minimize

$$S(\mathbf{a}, b; \lambda) = \left[(1/N) \sum_{i=1}^{N} \max(0, 1 - y_i (\mathbf{a} \cdot \mathbf{x}_i + b)) \right] + \frac{\lambda}{2} \mathbf{a}^T \mathbf{a}$$

where λ is some weight that balances the importance of a small hinge loss against the importance of a small $\|\mathbf{a}\|$. There are now two problems to solve. First, assume we know λ; we will need to find $\mathbf{a}$ and $\mathbf{b}$ that minimize $S(\mathbf{a}, b; \lambda)$. Second, we will need to estimate λ.

11.4.2 Finding a Minimum: General Points

I will first summarize general recipes for finding a minimum. Write $\mathbf{u} = [\mathbf{a}, b]$ for the vector obtained by stacking the vector $\mathbf{a}$ together with b. We have a function $g(\mathbf{u})$, and we wish to obtain a value of $\mathbf{u}$ that achieves the minimum for that function. Sometimes we can solve this problem in closed form by constructing the gradient

and finding a value of $\mathbf{u}$ the makes the gradient zero. This happens mainly for specially chosen problems that occur in textbooks. For practical problems, we tend to need a numerical method.

Typical methods take a point $\mathbf{u}^{(i)}$, update it to $\mathbf{u}^{(i+1)}$, then check to see whether the result is a minimum. This process is started from a start point. The choice of start point may or may not matter for general problems, but for our problem it won't matter. The update is usually obtained by computing a direction $\mathbf{p}^{(i)}$ such that for small values of h, $g(\mathbf{u}^{(i)} + h\mathbf{p}^{(i)})$ is smaller than $g(\mathbf{u}^{(i)})$. Such a direction is known as a **descent direction**. We must then determine how far to go along the descent direction, a process known as **line search**.

One method to choose a descent direction is **gradient descent**, which uses the negative gradient of the function. Recall our notation that

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ u_d \end{pmatrix}$$

and that

$$\nabla g = \begin{pmatrix} \frac{\partial g}{\partial u_1} \\ \frac{\partial g}{\partial u_2} \\ \dots \\ \frac{\partial g}{\partial u_d} \end{pmatrix}.$$

We can write a Taylor series expansion for the function $g(\mathbf{u}^{(i)} + h\mathbf{p}^{(i)})$. We have that

$$g(\mathbf{u}^{(i)} + h\mathbf{p}^{(i)}) = g(\mathbf{u}^{(i)}) + h\left[(\nabla g)^T \mathbf{p}^{(i)}\right] + O(h^2)$$

This means that we can expect that if

$$\mathbf{p}^{(i)} = -\nabla g(\mathbf{u}^{(i)}),$$

we expect that, at least for small values of h, $g(\mathbf{u}^{(i)} + h\mathbf{p}^{(i)})$ will be less than $g(\mathbf{u}^{(i)})$. This works (as long as g is differentiable, and quite often when it isn't) because g must go down for at least small steps in this direction.

11.4.3 Finding a Minimum: Stochastic Gradient Descent

Assume we wish to minimize some function $g(\mathbf{u}) = g_0(\mathbf{u}) + (1/N)\sum_{i=1}^{N} g_i(\mathbf{u})$, as a function of $\mathbf{u}$. Gradient descent would require us to form

$$-\nabla g(\mathbf{u}) = -\left(\nabla g_0(\mathbf{u}) + (1/N)\sum_{i=1}^{N} \nabla g_i(\mathbf{u})\right)$$

and then take a small step in this direction. But if N is large, this is unattractive, as we might have to sum a lot of terms. This happens a lot in building classifiers, where you might quite reasonably expect to deal with millions of examples. For some cases, there might be trillions of examples. Touching each example at each step really is impractical.

Instead, assume that, at each step, we choose a number k in the range $1 \ldots N$ uniformly and at random, and form

$$\mathbf{p}_k = -\left(\nabla g_0(\mathbf{u}) + \nabla g_k(\mathbf{u})\right)$$

and then take a small step along $\mathbf{p}_k$. Our new point becomes

$$\mathbf{a}^{(i+1)} = \mathbf{a}^{(i)} + \eta \mathbf{p}_k^{(i)},$$

where η is called the **step size** (or sometimes **steplength** or **learning rate**, even though it isn't the size or the length of the step we take, or a rate!). Here k is known as the **batch size**. This is often chosen using considerations of computer architecture (how many examples fit neatly into cache?) or of database design (how many examples are recovered in one disk cycle?).

It is easy to show that

$$\mathbb{E}[\mathbf{p}_k] = \nabla g(\mathbf{u})$$

(where the expectation is over the random choice of k). This implies that if we take many small steps along $\mathbf{p}_k$, they should average out to a step backwards along the gradient. This approach is known as **stochastic gradient descent** (because we're not going along the gradient, but along a random vector which is the gradient only in expectation). It isn't obvious that stochastic gradient descent is a good idea. Although each step is easy to take, we may need to take more steps. The question is then whether we gain in the increased speed of the step what we lose by having to take more steps. Not much is known theoretically, but in practice the approach is hugely successful for training classifiers.

Choosing a steplength η takes some work. We can't search for the step that gives us the best value of g, because we don't want to evaluate the function g (doing so involves looking at each of the g_i terms). Instead, we use a steplength that is large at the start — so that the method can explore large changes in the values of the classifier parameters — and small steps later — so that it settles down.

One useful strategy is to divide training into **seasons**. Each season is a block of a fixed number of iterations. Each iteration is one of the steps given above, with fixed steplength. However, the steplength changes from season to season. In particular, in the r'th season, the steplength is

$$\eta^{(r)} = \frac{m}{r+n}$$

where m and n are constants chosen by experiment with small subsets of the dataset. Often, but not always, the examples chosen are randomized by permuting the dataset randomly, which means that you can tell how many steps are required to have seen the whole dataset. An **epoch** is the number of steps required to have passed through the whole dataset once.

One cannot really test whether stochastic gradient descent has converged to the right answer. A better approach is to plot the error as a function of iteration on a validation set. This should vary randomly, but generally go down as the training proceeds. I have summarized this discussion in box 11.1. You should be aware that the recipe there admits many useful variations, though.

Procedure: 11.1 *Stochastic Gradient Descent*

We have a dataset containing N pairs $(\mathbf{x}_i, y_i)$. Each x_i is a d-dimensional feature vector, and each y_i is a label, either 1 or -1. Choose a set of possible values of the regularization weight λ. We wish to train a model that minimizes a cost function of the form $g(\mathbf{u}) = \frac{\lambda}{2}\mathbf{u}^T\mathbf{u} + (\frac{1}{N})\sum_{i=1}^{N} g_i(\mathbf{u})$. Separate the data into three sets: test, training and validation. For each value of the regularization weight, train a model, and evaluate the model on validation data. Keep the model that produces the lowest error rate on the validation data, and report its performance on the test data.
Train a model by choosing a fixed number of items per batch N_i, a fixed number of seasons N_s, and the number of steps per season N_t. Choose a random start point, $\mathbf{u}_0 = [\mathbf{a}, b]$. For each season, first compute the step size. In the r'th season, the step size is typically $\eta = \frac{m}{r+n}$ for constants m and n chosen by small-scale experiments (you try training a model with different values and see what happens). For the r'th season, choose a subset of the training set for validation for that season. Now repeat until the model has been updated N_s times:

- Take k steps. Each step is taken by selecting a N_i data items uniformly and at random. Write $\mathcal{D}$ for this set. We then compute

$$\mathbf{p} = -\frac{1}{N_i}\left(\sum_{i\in\mathcal{D}} \nabla g_i(\mathbf{u})\right) - \lambda\mathbf{u},$$

  and update the model by computing

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \eta\mathbf{p}$$

- Evaluate the current model by computing the accuracy on the validation set for that season. Plot the accuracy as a function of step number.

11.4.4 Example: Training an SVM with Stochastic Gradient Descent

I have summarized stochastic gradient descent in algorithm 11.1, but here is an example in more detail. We need to choose $\mathbf{a}$ and b to minimize

$$C(\mathbf{a}, b) = (1/N)\sum_{i=1}^{N} \max(0, 1 - y_i(\mathbf{a}\cdot\mathbf{x}_i + b)) + \frac{\lambda}{2}\mathbf{a}^T\mathbf{a}.$$

This is a support vector machine, because it uses hinge loss. For a support vector machine, stochastic gradient descent is particularly easy. We have estimates $\mathbf{a}^{(n)}$

and $b^{(n)}$ of the classifier parameters, and we want to improve the estimates. We pick the k'th example at random. We must now compute

$$\nabla \left(\max(0, 1 - y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right)) + \frac{\lambda}{2} \mathbf{a}^T \mathbf{a} \right).$$

Assume that $y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right) > 1$. In this case, the classifier predicts a score with the right sign, and a magnitude that is greater than one. Then the first term is zero, and the gradient of the second term is easy. Now if $y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right) < 1$, we can ignore the max, and the first term is $1 - y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right)$; the gradient is again easy. But what if $y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right) = 1$? there are two distinct values we could choose for the gradient, because the max term isn't differentiable. It turns out not to matter which term we choose (Figure **??**), so we can write the gradient as

$$p_k = \begin{cases} \begin{bmatrix} \lambda \mathbf{a} \\ 0 \end{bmatrix} & \text{if } y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right) \geq 1 \\ \begin{bmatrix} \lambda \mathbf{a} - y_k \mathbf{x} \\ -y_k \end{bmatrix} & \text{otherwise} \end{cases}$$

We choose a steplength η, and update our estimates using this gradient. This yields:

$$\mathbf{a}^{(n+1)} = \mathbf{a}^{(n)} - \eta \begin{cases} \lambda \mathbf{a} & \text{if } y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right) \geq 1 \\ \lambda \mathbf{a} - y_k \mathbf{x} & \text{otherwise} \end{cases}$$

and

$$b^{(n+1)} = b^{(n)} - \eta \begin{cases} 0 & \text{if } y_k \left(\mathbf{a} \cdot \mathbf{x}_k + b \right) \geq 1 \\ -y_k & \text{otherwise} \end{cases}.$$

To construct figures, I downloaded the dataset at http://archive.ics.uci.edu/ml/datasets/Adult. This dataset apparently contains 48, 842 data items, but I worked with only the first 32, 000. Each consists of a set of numeric and categorical features describing a person, together with whether their annual income is larger than or smaller than 50K\$. I ignored the categorical features to prepare these figures. This isn't wise if you want a good classifier, but it's fine for an example. I used these features to predict whether income is over or under 50K\$. I split the data into 5, 000 test examples, and 27,000 training examples. It's important to do so at random. There are 6 numerical features. I subtracted the mean (which doesn't usually make much difference) and rescaled each so that the variance was 1 (which is often very important). I used two different training regimes.

In the first training regime, there were 100 seasons. In each season, I applied 426 steps. For each step, I selected one data item uniformly at random (sampling with replacement), then stepped down the gradient. This means the method sees a total of 42, 600 data items. This means that there is a high probability it has touched each data item once (27, 000 isn't enough, because we are sampling with replacement, so some items get seen more than once). I chose 5 different values for the regularization parameter and trained with a steplength of $1/(0.01 * e + 50)$, where r is the season. At the end of each season, I computed $\mathbf{a}^T \mathbf{a}$ and the accuracy (fraction of examples correctly classified) of the current classifier on the held out

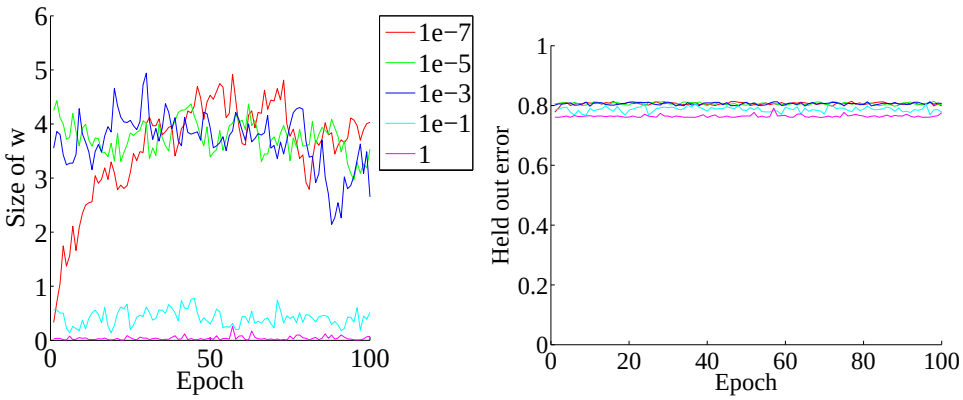

FIGURE 11.2: *On the* **left**, *the magnitude of the weight vector* $\mathbf{a}$ *at the end of each season for the first training regime described in the text. On the* **right**, *the accuracy on held out data at the end of each season. Notice how different choices of regularization parameter lead to different magnitudes of* $\mathbf{a}$*; how the method isn't particularly sensitive to choice of regularization parameter (they change by factors of 100); how the accuracy settles down fairly quickly; and how overlarge values of the regularization parameter do lead to a loss of accuracy.*

test examples. Figure 11.2 shows the results. You should notice that the accuracy changes slightly each season; that for larger regularizer values $\mathbf{a}^T\mathbf{a}$ is smaller; and that the accuracy settles down to about 0.8 very quickly.

In the second training regime, there were 100 seasons. In each season, I applied 50 steps. For each step, I selected one data item uniformly at random (sampling with replacement), then stepped down the gradient. This means the method sees a total of 5,000 data items, and about 3,000 unique data items — it hasn't seen the whole training set. I chose 5 different values for the regularization parameter and trained with a steplength of $1/(0.01 * r + 50)$, where r is the season. At the end of each season, I computed $\mathbf{a}^T\mathbf{a}$ and the accuracy (fraction of examples correctly classified) of the current classifier on the held out test examples. Figure 11.3 shows the results. You should notice that the accuracy changes slightly each season; that for larger regularizer values $\mathbf{a}^T\mathbf{a}$ is smaller; and that the accuracy settles down to about 0.8 very quickly; and that there isn't much difference between the two training regimes. All of these points are relatively typical of stochastic gradient descent with very large datasets.

Remember this: *Linear SVM's are a go-to classifier. When you have a binary classification problem, the first step should be to try a linear SVM. There is an immense quantity of good software available.*

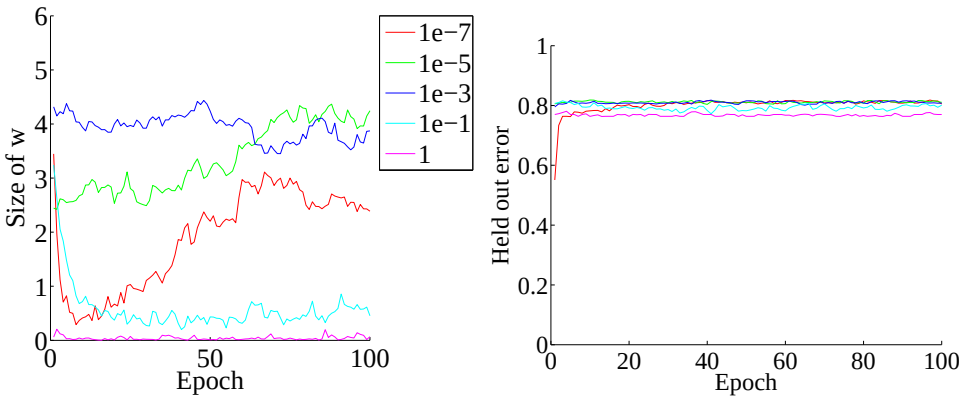

FIGURE 11.3: *On the* **left**, *the magnitude of the weight vector* **a** *at the end of each season for the second training regime described in the text. On the* **right**, *the accuracy on held out data at the end of each season. Notice how different choices of regularization parameter lead to different magnitudes of* **a**; *how the method isn't particularly sensitive to choice of regularization parameter (they change by factors of 100); how the accuracy settles down fairly quickly; and how overlarge values of the regularization parameter do lead to a loss of accuracy.*

11.4.5 Multi-Class Classification with SVMs

I have shown how one trains a linear SVM to make a binary prediction (i.e. predict one of two outcomes). But what if there are three, or more, labels? In principle, you could write a binary code for each label, then use a different SVM to predict each bit of the code. It turns out that this doesn't work terribly well, because an error by one of the SVM's is usually catastrophic.

There are two methods that are widely used. In the **all-vs-all** approach, we train a binary classifier for each pair of classes. To classify an example, we present it to each of these classifiers. Each classifier decides which of two classes the example belongs to, then records a vote for that class. The example gets the class label with the most votes. This approach is simple, but scales very badly with the number of classes (you have to build $O(N^2)$ different SVM's for N classes).

In the **one-vs-all** approach, we build a binary classifier for each class. This classifier must distinguish its class from all the other classes. We then take the class with the largest classifier score. One can think up quite good reasons this approach shouldn't work. For one thing, the classifier isn't told that you intend to use the score to tell similarity between classes. In practice, the approach works rather well and is quite widely used. This approach scales a bit better with the number of classes ($O(N)$).

Remember this: *It is straightforward to build a multi-class classifier out of binary classifiers. Any decent SVM package will do this for you.*

11.5 CLASSIFYING WITH RANDOM FORESTS

I described a classifier as a rule that takes a feature, and produces a class. One way to build such a rule is with a sequence of simple tests, where each test is allowed to use the results of all previous tests. This class of rule can be drawn as a tree (Figure **??**), where each node represents a test, and the edges represent the possible outcomes of the test. To classify a test item with such a tree, you present it to the first node; the outcome of the test determines which node it goes to next; and so on, until the example arrives at a leaf. When it does arrive at a leaf, we label the test item with the most common label in the leaf. This object is known as a **decision tree**. Notice one attractive feature of this decision tree: it deals with multiple class labels quite easily, because you just label the test item with the most common label in the leaf that it arrives at when you pass it down the tree.

Figure 11.5 shows a simple 2D dataset with four classes, next to a decision tree that will correctly classify at least the training data. Actually classifying data with a tree like this is straightforward. We take the data item, and pass it down the tree. Notice it can't go both left and right, because of the way the tests work. This means each data item arrives at a single leaf. We take the most common label at the leaf, and give that to the test item. In turn, this means we can build a geometric structure on the feature space that corresponds to the decision tree. I have illustrated that structure in figure 11.5, where the first decision splits the feature space in half (which is why the term split is used so often), and then the next decisions split each of those halves into two.

The important question is how to get the tree from data. It turns out that the best approach for building a tree incorporates a great deal of randomness. As a result, we will get a different tree each time we train a tree on a dataset. None of the individual trees will be particularly good (they are often referred to as "weak learners"). The natural thing to do is to produce many such trees (a **decision forest**), and allow each to vote; the class that gets the most votes, wins. This strategy is extremely effective.

11.5.1 Building a Decision Tree

There are many algorithms for building decision trees. We will use an approach chosen for simplicity and effectiveness; be aware there are others. We will always use a binary tree, because it's easier to describe and because that's usual (it doesn't change anything important, though). Each node has a **decision function**, which takes data items and returns either 1 or -1.

We train the tree by thinking about its effect on the training data. We pass the whole pool of training data into the root. Any node splits its incoming data into two pools, left (all the data that the decision function labels 1) and right (ditto,

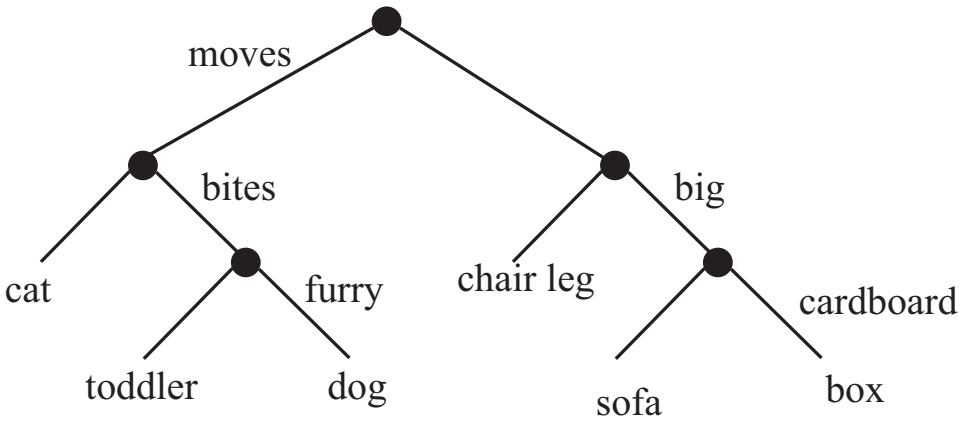

FIGURE 11.4: *This — the household robot's guide to obstacles — is a typical decision tree. I have labelled only one of the outgoing branches, because the other is the negation. So if the obstacle moves, bites, but isn't furry, then it's a toddler. In general, an item is passed down the tree until it hits a leaf. It is then labelled with the leaf's label.*

-1). Finally, each leaf contains a pool of data, which it can't split because it is a leaf.

Training the tree uses a straightforward algorithm. First, we choose a class of decision functions to use at each node. It turns out that a very effective algorithm is to choose a single feature at random, then test whether its value is larger than, or smaller than a threshold. For this approach to work, one needs to be quite careful about the choice of threshold, which is what we describe in the next section. Some minor adjustments, described below, are required if the feature chosen isn't ordinal. Surprisingly, being clever about the choice of *feature* doesn't seem add a great deal of value. We won't spend more time on other kinds of decision function, though there are lots.

Now assume we use a decision function as described, and we know how to choose a threshold. We start with the root node, then recursively either split the pool of data at that node, passing the left pool left and the right pool right, or stop splitting and return. Splitting involves choosing a decision function from the class to give the "best" split for a leaf. The main questions are how to choose the best split (next section), and when to stop.

Stopping is relatively straightforward. Quite simple strategies for stopping are very good. It is hard to choose a decision function with very little data, so we must stop splitting when there is too little data at a node. We can tell this is the case by testing the amount of data against a threshold, chosen by experiment. If all the data at a node belongs to a single class, there is no point in splitting. Finally, constructing a tree that is too deep tends to result in generalization problems, so we usually allow no more than a fixed depth D of splits. Choosing the best splitting threshold is more complicated.

Figure 11.6 shows two possible splits of a pool of training data. One is quite

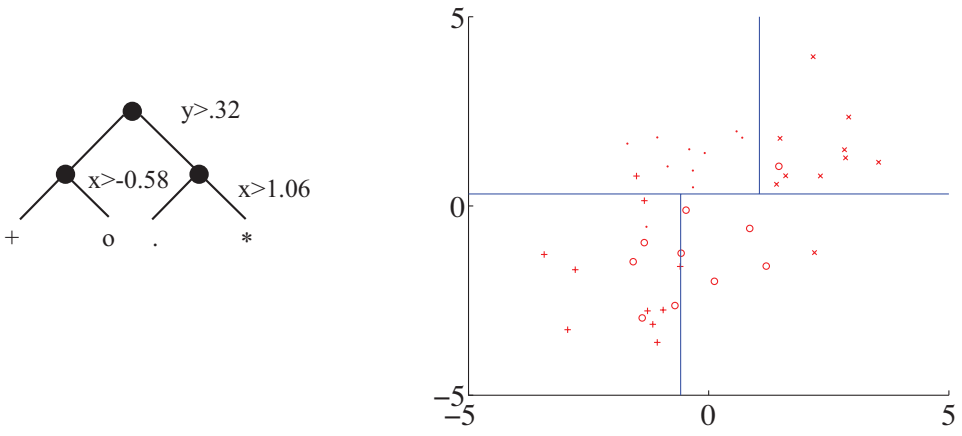

FIGURE 11.5: *A straightforward decision tree, illustrated in two ways. On the* **left**, *I have given the rules at each split; on the* **right**, *I have shown the data points in two dimensions, and the structure that the tree produces in the feature space.*

obviously a lot better than the other. In the good case, the split separates the pool into positives and negatives. In the bad case, each side of the split has the same number of positives and negatives. We cannot usually produce splits as good as the good case here. What we are looking for is a split that will make the proper label more certain.

Figure 11.7 shows a more subtle case to illustrate this. The splits in this figure are obtained by testing the horizontal feature against a threshold. In one case, the left and the right pools contain about the same fraction of positive ('x') and negative ('o') examples. In the other, the left pool is all positive, and the right pool is mostly negative. This is the better choice of threshold. If we were to label any item on the left side positive and any item on the right side negative, the error rate would be fairly small. If you count, the best error rate for the informative split is 20% on the training data, and for the uninformative split it is 40% on the training data.

But we need some way to score the splits, so we can tell which threshold is best. Notice that, in the uninformative case, knowing that a data item is on the left (or the right) does not tell me much more about the data than I already knew. We have that $p(1|\text{left pool, uninformative}) = 2/3 \approx 3/5 = p(1|\text{parent pool})$ and $p(1|\text{right pool, uninformative}) = 1/2 \approx 3/5 = p(1|\text{parent pool})$. For the informative pool, knowing a data item is on the left classifies it completely, and knowing that it is on the right allows us to classify it an error rate of $1/3$. The informative split means that my uncertainty about what class the data item belongs to is significantly reduced if I know whether it goes left or right. To choose a good threshold, we need to keep track of how informative the split is.

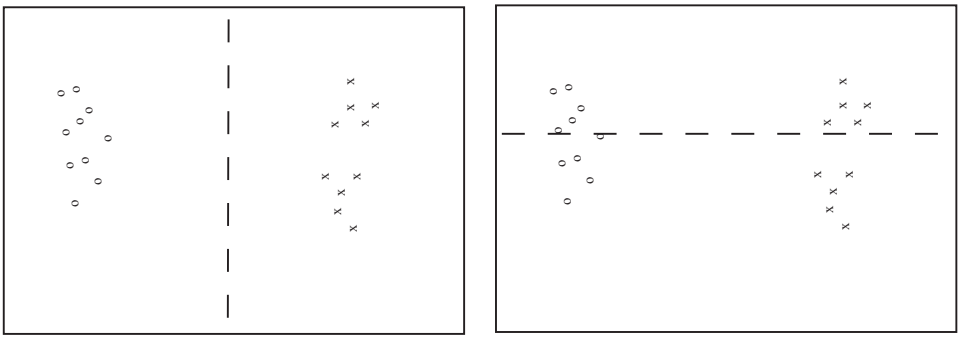

FIGURE 11.6: *Two possible splits of a pool of training data. Positive data is represented with an 'x', negative data with a 'o'. Notice that if we split this pool with the informative line, all the points on the left are 'o's, and all the points on the right are 'x's. This is an excellent choice of split — once we have arrived in a leaf, everything has the same label. Compare this with the less informative split. We started with a node that was half 'x' and half 'o', and now have two nodes each of which is half 'x' and half 'o' — this isn't an improvement, because we do not know more about the label as a result of the split.*

11.5.2 Choosing a Split with Information Gain

Write $\mathcal{P}$ for the set of all data at the node. Write $\mathcal{P}_l$ for the left pool, and $\mathcal{P}_r$ for the right pool. The entropy of a pool $\mathcal{C}$ scores how many bits would be required to represent the class of an item in that pool, on average. Write $n(i; \mathcal{C})$ for the number of items of class i in the pool, and $N(\mathcal{C})$ for the number of items in the pool. Then the entropy $H(\mathcal{C})$ of the pool $\mathcal{C}$ is

$$-\sum_i \frac{n(i;\mathcal{C})}{N(\mathcal{C})} \log_2 \frac{n(i;\mathcal{C})}{N(\mathcal{C}}.$$

It is straightforward that $H(\mathcal{P})$ bits are required to classify an item in the parent pool $\mathcal{P}$. For an item in the left pool, we need $H(\mathcal{P}_l)$ bits; for an item in the right pool, we need $H(\mathcal{P}_r)$ bits. If we split the parent pool, we expect to encounter items in the left pool with probability

$$\frac{N(\mathcal{P}_l)}{N(\mathcal{P})}$$

and items in the right pool with probability

$$\frac{N(\mathcal{P}_r)}{N(\mathcal{P})}.$$

This means that, on average, we must supply

$$\frac{N(\mathcal{P}_l)}{N(\mathcal{P})} H(\mathcal{P}_l) + \frac{N(\mathcal{P}_r)}{N(\mathcal{P})} H(\mathcal{P}_r)$$

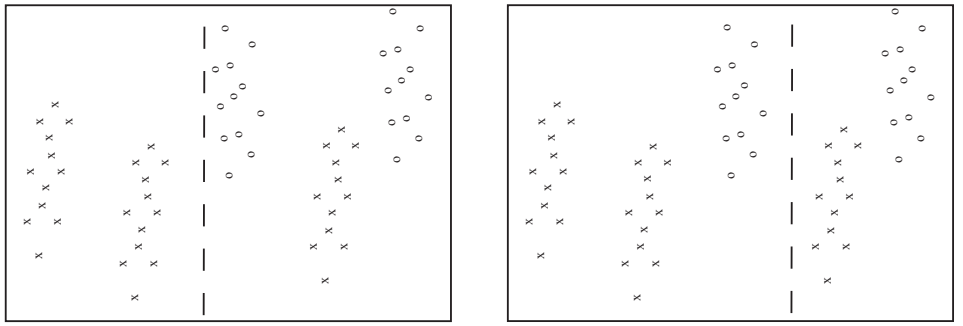

FIGURE 11.7: *Two possible splits of a pool of training data. Positive data is represented with an 'x', negative data with a 'o'. Notice that if we split this pool with the informative line, all the points on the left are 'x's, and two-thirds of the points on the right are 'o's. This means that knowing which side of the split a point lies would give us a good basis for estimating the label. In the less informative case, about two-thirds of the points on the left are 'x's and about half on the right are 'x's — knowing which side of the split a point lies is much less useful in deciding what the label is.*

bits to classify data items if we split the parent pool. Now a good split is one that results in left and right pools that are informative. In turn, we should need fewer bits to classify once we have split than we need before the split. You can see the difference

$$I(\mathcal{P}_l, \mathcal{P}_r; \mathcal{P}) = H(\mathcal{P}) - \left(\frac{N(\mathcal{P}_l)}{N(\mathcal{P})} H(\mathcal{P}_l) + \frac{N(\mathcal{P}_r)}{N(\mathcal{P})} H(\mathcal{P}_r) \right)$$

as the **information gain** caused by the split. This is the average number of bits that you *don't* have to supply if you know which side of the split an example lies. Better splits have larger information gain.

Recall that our decision function is to choose a feature at random, then test its value against a threshold. Any data point where the value is larger goes to the left pool; where the value is smaller goes to the right. This may sound much too simple to work, but it is actually effective and popular. Assume that we are at a node, which we will label k. We have the pool of training examples that have reached that node. The i'th example has a feature vector $\mathbf{x}_i$, and each of these feature vectors is a d dimensional vector.

We choose an integer j in the range $1 \ldots d$ uniformly and at random. We will split on this feature, and we store j in the node. Recall we write $x_i^{(j)}$ for the value of the j'th component of the i'th feature vector. We will choose a threshold t_k, and split by testing the sign of $x_i^{(j)} - t_k$. Choosing the value of t_k is easy. Assume there are N_k examples in the pool. Then there are $N_k - 1$ possible values of t_k that lead to different splits. To see this, sort the N_k examples by $x^{(j)}$, then choose values of t_k halfway between example values (Figure 11.8). For each of these values, we compute the information gain of the split. We then keep the threshold with the

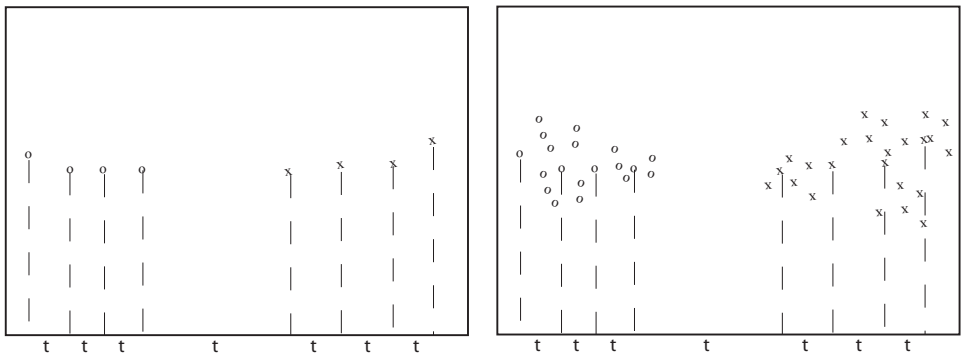

FIGURE 11.8: *We search for a good splitting threshold by looking at values of the chosen component that yield different splits. On the* **left***, I show a small dataset and its projection onto the chosen splitting component (the horizontal axis). For the 8 data points here, there are only 7 threshold values that produce interesting splits, and these are shown as 't's on the axis. On the* **right***, I show a larger dataset; in this case, I have projected only a subset of the data, which results in a small set of thresholds to search.*

best information gain.

We can elaborate this procedure in a useful way, by choosing m features at random, finding the best split for each, then keeping the feature and threshold value that is best. It is important that m is a lot smaller than the total number of features — a usual root of thumb is that m is about the square root of the total number of features. It is usual to choose a single m, and choose that for all the splits.

Now assume we happen to have chosen to work with a feature that isn't ordinal, and so can't be tested against a threshold. A natural, and effective, strategy is as follows. We can split such a feature into two pools by flipping an unbiased coin for each value — if the coin comes up H, any data point with that value goes left, and if it comes up T, any data point with that value goes right. We chose this split at random, so it might not be any good. We can come up with a good split by repeating this procedure F times, computing the information gain for each split, then keeping the one that has the best information gain. We choose F in advance, and it usually depends on the number of values the categorical variable can take.

We now have a relatively straightforward blueprint for an algorithm, which I have put in a box. It's a blueprint, because there are a variety of ways in which it can be revised and changed.

Procedure: 11.2 *Building a decision tree*

We have a dataset containing N pairs $(\mathbf{x}_i, y_i)$. Each x_i is a d-dimensional feature vector, and each y_i is a label. Call this dataset a **pool**. Now recursively apply the following procedure:

- If the pool is too small, or if all items in the pool have the same label, or if the depth of the recursion has reached a limit, stop.
- Otherwise, search the features for a good split that divides the pool into two, then apply this procedure to each child.

We search for a good split by the following procedure:

- Choose a subset of the feature components at random. Typically, one uses a subset whose size is about the square root of the feature dimension.
- For each component of this subset, search for the best splitting threshold. Do so by selecting a set of possible values for the threshold, then for each value splitting the dataset (every data item with a value of the component below the threshold goes left, others go right), and computing the information gain for the split. Keep the threshold that has the largest information gain.

A good set of possible values for the threshold will contain values that separate the data "reasonably". If the pool of data is small, you can project the data onto the feature component (i.e. look at the values of that component alone), then choose the $N - 1$ distinct values that lie between two data points. If it is big, you can randomly select a subset of the data, then project that subset on the feature component and choose from the values between data points.

11.5.3 Forests

A single decision tree tends to yield poor classifications. One reason is because the tree is not chosen to give the best classification of its training data. We used a random selection of splitting variables at each node, so the tree can't be the "best possible". Obtaining the best possible tree presents significant technical difficulties. It turns out that the tree that gives the best possible results on the training data can perform rather poorly on test data. The training data is a small subset of possible examples, and so must differ from the test data. The best possible tree on the training data might have a large number of small leaves, built using carefully chosen splits. But the choices that are best for training data might not be best for test data.

Rather than build the best possible tree, we have built a tree efficiently, but

with number of random choices. If we were to rebuild the tree, we would obtain a different result. This suggests the following extremely effective strategy: build many trees, and classify by merging their results.

11.5.4 Building and Evaluating a Decision Forest

There are two important strategies for building and evaluating decision forests. I am not aware of evidence strongly favoring one over the other, but different software packages use different strategies, and you should be aware of the options. In one strategy, we separate labelled data into a training and a test set. We then build multiple decision trees, training each using the whole training set. Finally, we evaluate the forest on the test set. In this approach, the forest has not seen some fraction of the available labelled data, because we used it to test. However, each tree has seen every training data item.

Procedure: 11.3 *Building a decision forest*

We have a dataset containing N pairs $(\mathbf{x}_i, y_i)$. Each $\mathbf{x}_i$ is a d-dimensional feature vector, and each y_i is a label. Separate the dataset into a test set and a training set. Train multiple distinct decision trees on the training set, recalling that the use of a random set of components to find a good split means you will obtain a distinct tree each time.

In the other strategy, sometimes called **bagging**, each time we train a tree we randomly subsample the labelled data with replacement, to yield a training set the same size as the original set of labelled data. Notice that there will be duplicates in this training set, which is like a bootstrap replicate. This training set is often called a **bag**. We keep a record of the examples that do not appear in the bag (the "out of bag" examples). Now to evaluate the forest, we evaluate each tree on its out of bag examples, and average these error terms. In this approach, the entire forest has seen all labelled data, and we also get an estimate of error, but no tree has seen all the training data.

Procedure: 11.4 *Building a decision forest using bagging*

We have a dataset containing N pairs $(\mathbf{x}_i, y_i)$. Each $\mathbf{x}_i$ is a d-dimensional feature vector, and each y_i is a label. Now build k bootstrap replicates of the training data set. Train one decision tree on each replicate.

11.5.5 Classifying Data Items with a Decision Forest

Once we have a forest, we must classify test data items. There are two major strategies. The simplest is to classify the item with each tree in the forest, then take the class with the most votes. This is effective, but discounts some evidence that might be important. For example, imagine one of the trees in the forest has a leaf with many data items with the same class label; another tree has a leaf with exactly one data item in it. One might not want each leaf to have the same vote.

Procedure: 11.5 *Classification with a decision forest*

Given a test example $\mathbf{x}$, pass it down each tree of the forest. Now choose one of the following strategies.

- Each time the example arrives at a leaf, record one vote for the label that occurs most often at the leaf. Now choose the label with the most votes.
- Each time the example arrives at a leaf, record N_l votes for each of the labels that occur at the leaf, where N_l is the number of times the label appears in the training data at the leaf. Now choose the label with the most votes.

An alternative strategy that takes this observation into account is to pass the test data item down each tree. When it arrives at a leaf, we record one vote for each of the training data items in that leaf. The vote goes to the class of the training data item. Finally, we take the class with the most votes. This approach allows big, accurate leaves to dominate the voting process. Both strategies are in use, and I am not aware of compelling evidence that one is always better than the other. This may be because the randomness in the training process makes big, accurate leaves uncommon in practice.

Worked example 11.3 *Classifying heart disease data*

Build a random forest classifier to classify the "heart" dataset from the UC Irvine machine learning repository. The dataset is at http://archive.ics.uci.edu/ml/datasets/Heart+Disease. There are several versions. You should look at the processed Cleveland data, which is in the file "processed.cleveland.data.txt".

Solution: I used the R random forest package. This uses a bagging strategy. This package makes it quite simple to fit a random forest, as you can see. In this dataset, variable 14 (V14) takes the value 0, 1, 2, 3 or 4 depending on the severity of the narrowing of the arteries. Other variables are physiological and physical measurements pertaining to the patient (read the details on the website). I tried to predict all five levels of variable 14, using the random forest as a multivariate classifier. This works rather poorly, as the out-of-bag class confusion matrix below shows. The total out-of-bag error rate was 45%.

	Predict 0	Predict 1	Predict 2	Predict 3	Predict 4	Class error
True 0	151	7	2	3	1	7.9%
True 1	32	5	9	9	0	91%
True 2	10	9	7	9	1	81%
True 3	6	13	9	5	2	86%
True 4	2	3	2	6	0	100%

This is the example of a class confusion matrix from table 11.1. Fairly clearly, one can predict narrowing or no narrowing from the features, but not the degree of narrowing (at least, not with a random forest). So it is natural to quantize variable 14 to two levels, 0 (meaning no narrowing), and 1 (meaning any narrowing, so the original value could have been 1, 2, or 3). I then built a random forest to predict this from the other variables. The total out-of-bag error rate was 19%, and I obtained the following out-of-bag class confusion matrix

	Predict 0	Predict 1	Class error
True 0	138	26	16%
True 1	31	108	22%

Notice that the false positive rate (16%, from 26/164) is rather better than the false negative rate (22%). Looking at these class confusion matrices, you might wonder whether it is better to predict $0, \ldots, 4$, then quantize. But this is not a particularly good idea. While the false positive rate is 7.9%, the false negative rate is much higher (36%, from 50/139). In this application, a false negative is likely more of a problem than a false positive, so the tradeoff is unattractive.

Remember this: *Random forests are straightforward to build, and very effective. They can predict any kind of label. Good software implementations are easily available.*

11.6 YOU SHOULD

11.6.1 remember these definitions:

11.6.2 remember these terms:

11.6.3 remember these facts:

11.6.4 remember these procedures:

11.6.5 be able to:

- build a nearest neighbors classifier using your preferred software package, and produce a cross-validated estimate of its error rate or its accuracy;
- build a naive bayes classifier using your preferred software package, and produce a cross-validated estimate of its error rate or its accuracy;
- build an SVM using your preferred software package, and produce a cross-validated estimate of its error rate or its accuracy;
- write code to train an SVM using stochastic gradient descent, and produce a cross-validated estimate of its error rate or its accuracy;
- and build a decision forest using your preferred software package, and produce a cross-validated estimate of its error rate or its accuracy.

PROBLEMS

PROGRAMMING EXERCISES

11.1. The UC Irvine machine learning data repository hosts a famous collection of data on whether a patient has diabetes (the Pima Indians dataset), originally owned by the National Institute of Diabetes and Digestive and Kidney Diseases and donated by Vincent Sigillito. This can be found at http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes. This data has a set of attributes of patients, and a categorical variable telling whether the patient is diabetic or not. For several attributes in this data set, a value of 0 may indicate a missing value of the variable.

- **(a)** Build a simple naive Bayes classifier to classify this data set. You should hold out 20% of the data for evaluation, and use the other 80% for training. You should use a normal distribution to model each of the class-conditional distributions. You should write this classifier yourself (it's quite straightforward), but you may find the function `createDataPartition` in the R package `caret` helpful to get the random partition.
- **(b)** Now adjust your code so that, for attribute 3 (Diastolic blood pressure), attribute 4 (Triceps skin fold thickness), attribute 6 (Body mass index), and attribute 8 (Age), it regards a value of 0 as a missing value when estimating the class-conditional distributions, and the posterior. R uses a special number `NA` to flag a missing value. Most functions handle this number in special, but sensible, ways; but you'll need to do a bit of looking at manuals to check. Does this affect the accuracy of your classifier?
- **(c)** Now use the `caret` and `klaR` packages to build a naive bayes classifier for this data, assuming that no attribute has a missing value. The `caret` package does cross-validation (look at `train`) and can be used to hold out data. The `klaR` package can estimate class-conditional densities using a density estimation procedure that I will describe much later in the course. Use the cross-validation mechanisms in `caret` to estimate the accuracy of your classifier. I have not been able to persuade the combination of `caret` and `klaR` to handle missing values the way I'd like them to, but that may be ignorance (look at the `na.action` argument).
- **(d)** Now install SVMLight, which you can find at http://svmlight.joachims.org, via the interface in `klaR` (look for `svmlight` in the manual) to train and evaluate an SVM to classify this data. You don't need to understand much about SVM's to do this — we'll do that in following exercises. You should hold out 20% of the data for evaluation, and use the other 80% for training. You should NOT substitute `NA` values for zeros for attributes 3, 4, 6, and 8.

11.2. The UC Irvine machine learning data repository hosts a collection of data on student performance in Portugal, donated by Paulo Cortez, University of Minho, in Portugal. You can find this data at https://archive.ics.uci.edu/ml/datasets/Student+Performance. It is described in P. Cortez and A. Silva. Using Data Mining to Predict Secondary School Student Performance. In A. Brito and J. Teixeira Eds., Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008) pp. 5-12, Porto, Portugal, April, 2008, EUROSIS, ISBN 978-9077381-39-7.

There are two datasets (for grades in mathematics and for grades in Portugese). There are 30 attributes each for 649 students, and 3 values that can

be predicted (G1, G2 and G3). Of these, ignore G1 and G2.

(a) Use the mathematics dataset. Take the G3 attribute, and quantize this into two classes, $G3 > 12$ and $G3 \leq 12$. Build and evaluate a naive bayes classifier that predicts G3 from all attributes except G1 and G2. You should build this classifier from scratch (i.e. DON'T use the packages described in the code snippets). For binary attributes, you should use a binomial model. For the attributes described as "numeric", which take a small set of values, you should use a multinomial model. For the attributes described as "nominal", which take a small set of values, you should again use a multinomial model. Ignore the "absence" attribute. Estimate accuracy by cross-validation. You should use at least 10 folds, excluding 15% of the data at random to serve as test data, and average the accuracy over those folds. Report the mean and standard deviation of the accuracy over the folds.

(b) Now revise your classifier of the previous part so that, for the attributes described as "numeric", which take a small set of values, you use a multinomial model. For the attributes described as "nominal", which take a small set of values, you should still use a multinomial model. Ignore the "absence" attribute. Estimate accuracy by cross-validation. You should use at least 10 folds, excluding 15% of the data at random to serve as test data, and average the accuracy over those folds. Report the mean and standard deviation of the accuracy over the folds.

(c) Which classifier do you believe is more accurate and why?

11.3. The UC Irvine machine learning data repository hosts a collection of data on heart disease. The data was collected and supplied by Andras Janosi, M.D., of the Hungarian Institute of Cardiology, Budapest; William Steinbrunn, M.D., of the University Hospital, Zurich, Switzerland; Matthias Pfisterer, M.D., of the University Hospital, Basel, Switzerland; and Robert Detrano, M.D., Ph.D., of the V.A. Medical Center, Long Beach and Cleveland Clinic Foundation. You can find this data at https://archive.ics.uci.edu/ml/datasets/Heart+Disease.

Use the processed Cleveland dataset, where there are a total of 303 instances with 14 attributes each. The irrelevant attributes described in the text have been removed in these. The 14'th attribute is the disease diagnosis. There are records with missing attributes, and you should drop these.

(a) Take the disease attribute, and quantize this into two classes, num $= 0$ and num > 0. Build and evaluate a naive bayes classifier that predicts the class from all other attributes Estimate accuracy by cross-validation. You should use at least 10 folds, excluding 15% of the data at random to serve as test data, and average the accuracy over those folds. Report the mean and standard deviation of the accuracy over the folds.

(b) Now revise your classifier to predict each of the possible values of the disease attribute (0-4 as I recall). Estimate accuracy by cross-validation. You should use at least 10 folds, excluding 15% of the data at random to serve as test data, and average the accuracy over those folds. Report the mean and standard deviation of the accuracy over the folds.

11.4. The UC Irvine machine learning data repository hosts a collection of data on breast cancer diagnostics, donated by Olvi Mangasarian, Nick Street, and William H. Wolberg. You can find this data at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic). For each record, there is an id number, 10 continuous variables, and a class (benign or malignant). There are 569 examples. Separate this dataset randomly into 100 validation, 100

test, and 369 training examples.
Write a program to train a support vector machine on this data using stochastic gradient descent. You should not use a package to train the classifier (you don't really need one), but your own code. You should ignore the id number, and use the continuous variables as a feature vector. You should search for an appropriate value of the regularization constant, trying at least the values $\lambda = [1e-3, 1e-2, 1e-1, 1]$. Use the validation set for this search
You should use at least 50 epochs of at least 100 steps each. In each epoch, you should separate out 50 training examples at random for evaluation. You should compute the accuracy of the current classifier on the set held out for the epoch every 10 steps. You should produce:

(a) A plot of the accuracy every 10 steps, for each value of the regularization constant.

(b) Your estimate of the best value of the regularization constant, together with a brief description of why you believe that is a good value.

(c) Your estimate of the accuracy of the best classifier on held out data

11.5. The UC Irvine machine learning data repository hosts a collection of data on adult income, donated by Ronny Kohavi and Barry Becker. You can find this data at https://archive.ics.uci.edu/ml/datasets/Adult For each record, there is a set of continuous attributes, and a class (¿=50K or ¡50K). There are 48842 examples. You should use only the continous attributes (see the description on the web page) and drop examples where there are missing values of the continuous attributes. Separate the resulting dataset randomly into 10% validation, 10% test, and 80% training examples.
Write a program to train a support vector machine on this data using stochastic gradient descent. You should not use a package to train the classifier (you don't really need one), but your own code. You should ignore the id number, and use the continuous variables as a feature vector. You should search for an appropriate value of the regularization constant, trying at least the values $\lambda = [1e-3, 1e-2, 1e-1, 1]$. Use the validation set for this search
You should use at least 50 epochs of at least 300 steps each. In each epoch, you should separate out 50 training examples at random for evaluation. You should compute the accuracy of the current classifier on the set held out for the epoch every 30 steps. You should produce:

(a) A plot of the accuracy every 30 steps, for each value of the regularization constant.

(b) Your estimate of the best value of the regularization constant, together with a brief description of why you believe that is a good value.

(c) Your estimate of the accuracy of the best classifier on held out data

11.6. The UC Irvine machine learning data repository hosts a collection of data on the whether p53 expression is active or inactive.
You can find out what this means, and more information about the dataset, by reading: Danziger, S.A., Baronio, R., Ho, L., Hall, L., Salmon, K., Hatfield, G.W., Kaiser, P., and Lathrop, R.H. (2009) Predicting Positive p53 Cancer Rescue Regions Using Most Informative Positive (MIP) Active Learning, PLOS Computational Biology, 5(9); Danziger, S.A., Zeng, J., Wang, Y., Brachmann, R.K. and Lathrop, R.H. (2007) Choosing where to look next in a mutation sequence space: Active Learning of informative p53 cancer rescue mutants, Bioinformatics, 23(13), 104-114; and Danziger, S.A., Swamidass, S.J., Zeng, J., Dearth, L.R., Lu, Q., Chen, J.H., Cheng, J., Hoang, V.P., Saigo, H., Luo, R., Baldi, P., Brachmann, R.K. and Lathrop, R.H. (2006) Functional

census of mutation sequence spaces: the example of p53 cancer rescue mutants, IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM, 3, 114-125.

You can find this data at https://archive.ics.uci.edu/ml/datasets/p53+Mutants. There are a total of 16772 instances, with 5409 attributes per instance. Attribute 5409 is the class attribute, which is either active or inactive. There are several versions of this dataset. You should use the version `K8.data`.

(a) Train an SVM to classify this data, using stochastic gradient descent. You will need to drop data items with missing values. You should estimate a regularization constant using cross-validation, trying at least 3 values. Your training method should touch at least 50% of the training set data. You should produce an estimate of the accuracy of this classifier on held out data consisting of 10% of the dataset, chosen at random.

(b) Now train a naive bayes classifier to classify this data. You should produce an estimate of the accuracy of this classifier on held out data consisting of 10% of the dataset, chosen at random.

(c) Compare your classifiers. Which one is better? why?

11.7. The UC Irvine machine learning data repository hosts a collection of data on whether a mushroom is edible, donated by Jeff Schlimmer and to be found at http://archive.ics.uci.edu/ml/datasets/Mushroom. This data has a set of categorical attributes of the mushroom, together with two labels (poisonous or edible). Use the R random forest package (as in the example in the chapter) to build a random forest to classify a mushroom as edible or poisonous based on its attributes.

(a) Produce a class-confusion matrix for this problem. If you eat a mushroom based on your classifier's prediction it is edible, what is the probability of being poisoned?

CHAPTER 12

Clustering: Models of High Dimensional Data

High-dimensional data comes with problems. Data points tend not to be where you think; they can scattered quite far apart, and can be quite far from the mean. Except in special cases, the only really reliable probability model is the Gaussian (or Gaussian blob, or blob).

There is an important rule of thumb for coping with high dimensional data: **Use simple models**. A blob is a good simple model. Modelling data as a blob involves computing its mean and its covariance. Sometimes, as we shall see, the covariance can be hard to compute. Even so, a blob model is really useful. It is natural to try and extend this model to cover datasets that don't obviously consist of a single blob.

One very good, very simple, model for high dimensional data is to assume that it consists of multiple blobs. To build models like this, we must determine (a) what the blob parameters are and (b) which datapoints belong to which blob. Generally, we will collect together data points that are close and form blobs out of them. This process is known as **clustering**.

Clustering is a somewhat puzzling activity. It is extremely useful to cluster data, and it seems to be quite important to do it reasonably well. But it surprisingly hard to give crisp criteria for a good (resp. bad) clustering of a dataset. Usually, clustering is part of building a model, and the main way to know that the clustering algorithm is bad is that the model is bad.

12.1 THE CURSE OF DIMENSION

High dimensional models display uninituitive behavior (or, rather, it can take years to make your intuition see the true behavior of high-dimensional models as natural). In these models, most data lies in places you don't expect. We will do several simple calculations with an easy high-dimensional distribution to build some intuition.

12.1.1 The Curse: Data isn't Where You Think it is

Assume our data lies within a cube, with edge length two, centered on the origin. This means that each component of $\mathbf{x}_i$ lies in the range $[-1, 1]$. One simple model for such data is to assume that each dimension has uniform probability density in this range. In turn, this means that $P(x) = \frac{1}{2^d}$. The mean of this model is at the origin, which we write as $\mathbf{0}$.

The first surprising fact about high dimensional data is that most of the data can lie quite far away from the mean. For example, we can divide our dataset into two pieces. $\mathcal{A}(\epsilon)$ consists of all data items where *every* component of the data has

a value in the range $[-(1-\epsilon),(1-\epsilon)]$. $\mathcal{B}(\epsilon)$ consists of all the rest of the data. If you think of the data set as forming a cubical orange, then $\mathcal{B}(\epsilon)$ is the rind (which has thickness ϵ) and $\mathcal{A}(\epsilon)$ is the fruit.

Your intuition will tell you that there is more fruit than rind. This is true, for three dimensional oranges, but not true in high dimensions. The fact that the orange is cubical just simplifies the calculations, but has nothing to do with the real problem.

We can compute $P(\{\mathbf{x} \in \mathcal{A}(\epsilon)\})$ and $P(\{\mathbf{x} \in \mathcal{A}(\epsilon)\})$. These probabilities tell us the probability a data item lies in the fruit (resp. rind). $P(\{\mathbf{x} \in \mathcal{A}(\epsilon)\})$ is easy to compute as

$$P(\{\mathbf{x} \in \mathcal{A}(\epsilon)\}) = (2(1-\epsilon)))^d \left(\frac{1}{2^d}\right) = (1-\epsilon)^d$$

and

$$P(\{\mathbf{x} \in \mathcal{B}(\epsilon)\}) = 1 - P(\{\mathbf{x} \in \mathcal{A}(\epsilon)\}) = 1 - (1-\epsilon)^d.$$

But notice that, as $d \to \infty$,

$$P(\{\mathbf{x} \in \mathcal{A}(\epsilon)\}) \to 0.$$

This means that, for large d, we expect most of the data to be in $\mathcal{B}(\epsilon)$. Equivalently, for large d, we expect that at least one component of each data item is close to either 1 or -1.

This suggests (correctly) that much data is quite far from the origin. It is easy to compute the average of the squared distance of data from the origin. We want

$$\mathbb{E}\left[\mathbf{x}^T\mathbf{x}\right] = \int_{\text{box}} \left(\sum_i x_i^2\right) P(\mathbf{x}) d\mathbf{x}$$

but we can rearrange, so that

$$\mathbb{E}\left[\mathbf{x}^T\mathbf{x}\right] = \sum_i \mathbb{E}\left[x_i^2\right] = \sum_i \int_{\text{box}} x_i^2 P(\mathbf{x}) d\mathbf{x}.$$

Now each component of $\mathbf{x}$ is independent, so that $P(\mathbf{x}) = P(x_1)P(x_2)\ldots P(x_d)$. Now we substitute, to get

$$\mathbb{E}\left[\mathbf{x}^T\mathbf{x}\right] = \sum_i \mathbb{E}\left[x_i^2\right] = \sum_i \int_{-1}^{1} x_i^2 P(x_i) dx_i = \sum_i \frac{1}{2}\int_{-1}^{1} x_i^2 dx_i = \frac{d}{3},$$

so as d gets bigger, most data points will be further and further from the origin. Worse, as d gets bigger, data points tend to get further and further from one another. We can see this by computing the average of the squared distance of data points from one another. Write $\mathbf{u}$ for one data point and $\mathbf{v}$; we can compute

$$\mathbb{E}\left[d(\mathbf{u},\mathbf{v})^2\right] = \int_{\text{box}}\int_{\text{box}} \sum_i (u_i - v_i)^2 d\mathbf{u} d\mathbf{v} = \mathbb{E}\left[\mathbf{u}^T\mathbf{u}\right] + \mathbb{E}\left[\mathbf{v}^T\mathbf{v}\right] - 2\mathbb{E}\left[\mathbf{u}^T\mathbf{v}\right]$$

but since $\mathbf{u}$ and $\mathbf{v}$ are independent, we have $\mathbb{E}\left[\mathbf{u}^T\mathbf{v}\right] = \mathbb{E}[\mathbf{u}]^T\mathbb{E}[\mathbf{v}] = 0$. This yields

$$\mathbb{E}\left[d(\mathbf{u},\mathbf{v})^2\right] = 2\frac{d}{3}.$$

This means that, for large d, we expect our data points to be quite far apart.

12.1.2 Minor Banes of Dimension

High dimensional data presents a variety of important practical nuisances which follow from the curse of dimension. It is hard to estimate covariance matrices, and it is hard to build histograms.

Covariance matrices are hard to work with for two reasons. The number of entries in the matrix grows as the square of the dimension, so the matrix can get big and so difficult to store. More important, the amount of data we need to get an accurate estimate of all the entries in the matrix grows fast. As we are estimating more numbers, we need more data to be confident that our estimates are reasonable. There are a variety of straightforward work-arounds for this effect. In some cases, we have so much data there is no need to worry. In other cases, we assume that the covariance matrix has a particular form, and just estimate those parameters. There are two strategies that are usual. In one, we assume that the covariance matrix is diagonal, and estimate only the diagonal entries. In the other, we assume that the covariance matrix is a scaled version of the identity, and just estimate this scale. You should see these strategies as acts of desperation, to be used only when computing the full covariance matrix seems to produce more problems than using these approaches.

It is difficult to build histogram representations for high dimensional data. The strategy of dividing the domain into boxes, then counting data into them, fails miserably because there are too many boxes. In the case of our cube, imagine we wish to divide each dimension in half (i.e. between $[-1, 0]$ and between $[0, 1]$). Then we must have 2^d boxes. This presents two problems. First, we will have difficulty representing this number of boxes. Second, unless we are exceptionally lucky, most boxes must be empty because we will not have 2^d data items.

However, one representation is extremely effective. We can represent data as a collection of **clusters** — coherent blobs of similar datapoints that could, under appropriate circumstances, be regarded as the same. We could then represent the dataset by, for example, the center of each cluster and the number of data items in each cluster. Since each cluster is a blob, we could also report the covariance of each cluster, if we can compute it.

Remember this: *High dimensional data does not behave in a way that is consistent with most people's intuition. Points are always close to the boundary and further apart than you think. This property makes a nuisance of itself in a variety of ways. The most important is that only the simplest models work well in high dimensions.*

12.2 THE MULTIVARIATE NORMAL DISTRIBUTION

All the nasty facts about high dimensional data, above, suggest that we need to use quite simple probability models. By far the most important model is the multivariate normal distribution, which is quite often known as the multivariate gaussian distribution. There are two sets of parameters in this model, the mean μ and the covariance Σ. For a d-dimensional model, the mean is a d-dimensional column vector and the covariance is a $d \times d$ dimensional matrix. The covariance is a symmetric matrix. For our definitions to be meaningful, the covariance matrix must be positive definite.

The form of the distribution $p(\mathbf{x}|\mu, \Sigma)$ is

$$p(\mathbf{x}|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right).$$

The following facts explain the names of the parameters:

Useful Facts: 12.1 *Parameters of a Multivariate Normal Distribution*

Assuming a multivariate normal distribution, we have

- $\mathbb{E}[\mathbf{x}] = \mu$, meaning that the mean of the distribution is μ.
- $\mathbb{E}\left[(\mathbf{x}-\mu)(\mathbf{x}-\mu)^T\right] = \Sigma$, meaning that the entries in Σ represent covariances.

Assume I know have a dataset of items $\mathbf{x}_i$, where i runs from 1 to N, and we wish to model this data with a multivariate normal distribution. The maximum likelihood estimate of the mean, $\hat{\mu}$, is

$$\hat{\mu} = \frac{\sum_i \mathbf{x}_i}{N}$$

(which is quite easy to show). The maximum likelihood estimate of the covariance, $\hat{\Sigma}$, is

$$\hat{\Sigma} = \frac{\sum_i (\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^T}{N}$$

(which is rather a nuisance to show, because you need to know how to differentiate a determinant). These facts mean that we already know most of what is interesting about multivariate normal distributions (or gaussians).

12.2.1 Affine Transformations and Gaussians

Gaussians behave very well under affine transformations. In fact, we've already worked out all the math. Assume I have a dataset $\mathbf{x}_i$. The mean of the maximum likelihood gaussian model is $\mathsf{mean}(\{\mathbf{x}_i\})$, and the covariance is $\mathsf{Covmat}(\{\mathbf{x}_i\})$. I can now transform the data with an affine transformation, to get $\mathbf{y}_i = \mathcal{A}\mathbf{x}_i + \mathbf{b}$.

The mean of the maximum likelihood gaussian model for the transformed dataset is $\mathsf{mean}\left(\{\mathbf{y}_i\}\right)$, and we've dealt with this; similarly, the covariance is $\mathsf{Covmat}\left(\{\mathbf{y}_i\}\right)$, and we've dealt with this, too.

A very important point follows in an obvious way. I can apply an affine transformation to any multivariate gaussian to obtain one with (a) zero mean and (b) independent components. In turn, this means that, *in the right coordinate system*, any gaussian is a product of zero mean one-dimensional normal distributions. This fact is quite useful. For example, it means that simulating multivariate normal distributions is quite straightforward — you could simulate a standard normal distribution for each component, then apply an affine transformation.

12.2.2 Plotting a 2D Gaussian: Covariance Ellipses

There are some useful tricks for plotting a 2D Gaussian, which are worth knowing both because they're useful, and they help to understand Gaussians. Assume we are working in 2D; we have a Gaussian with mean μ (which is a 2D vector), and covariance Σ (which is a 2x2 matrix). We could plot the collection of points $\mathbf{x}$ that has some fixed value of $p(\mathbf{x}|\mu, \Sigma)$. This set of points is given by:

$$\frac{1}{2}\left((\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right) = c^2$$

where c is some constant. I will choose $c^2 = \frac{1}{2}$, because the choice doesn't matter, and this choice simplifies some algebra. You might recall that a set of points $\mathbf{x}$ that satisfies a quadratic like this is a conic section. Because Σ (and so Σ^{-1}) is positive definite, the curve is an ellipse. There is a useful relationship between the geometry of this ellipse and the Gaussian.

This ellipse — like all ellipses — has a major axis and a minor axis. These are at right angles, and meet at the center of the ellipse. We can determine the properties of the ellipse in terms of the Gaussian quite easily. The geometry of the ellipse isn't affected by rotation or translation, so we will translate the ellipse so that $\mu = \mathbf{0}$ (i.e. the mean is at the origin) and rotate it so that Σ^{-1} is diagonal. Writing $\mathbf{x} = [x, y]$ we get that the set of points on the ellipse satisfies

$$\frac{1}{2}(\frac{1}{k_1^2}x^2 + \frac{1}{k_2^2}y^2) = \frac{1}{2}$$

where $\frac{1}{k_1^2}$ and $\frac{1}{k_2^2}$ are the diagonal elements of Σ^{-1}. We will assume that the ellipse has been rotated so that $k_1 < k_2$. The points $(k_1, 0)$ and $(-k_1, 0)$ lie on the ellipse, as do the points $(0, k_2)$ and $(0, -k_2)$. The major axis of the ellipse, in this coordinate system, is the x-axis, and the minor axis is the y-axis. In this coordinate system, x and y are independent. If you do a little algebra, you will see that the standard deviation of x is $\mathsf{abs}\left(k_1\right)$ and the standard deviation of y is $\mathsf{abs}\left(k_2\right)$. So the ellipse is longer in the direction of largest standard deviation and shorter in the direction of smallest standard deviation.

Now rotating the ellipse is means we will pre- and post-multiply the covariance matrix with some rotation matrix. Translating it will move the origin to the mean. As a result, the ellipse has its center at the mean, its major axis is in the direction of the eigenvector of the covariance with largest eigenvalue, and its minor axis is

in the direction of the eigenvector with smallest eigenvalue. A plot of this ellipse, which can be coaxed out of most programming environments with relatively little effort, gives us a great deal of information about the underlying Gaussian. These ellipses are known as **covariance ellipses**.

Remember this: *The multivariate normal distribution has the form*

$$p(\mathbf{x}|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^d det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right).$$

Assume you wish to model a dataset $\{\mathbf{x}\}$ *with a multivariate normal distribution. The maximum likelihood estimate of the mean is* $\mathsf{mean}(\{\mathbf{x}\})$. *The maximum likelihood estimate of the covariance* Σ *is* $\mathsf{Covmat}(\{\mathbf{x}\})$.

12.3 AGGLOMERATIVE AND DIVISIVE CLUSTERING

There are two natural recipes you can use to produce clustering algorithms. In **agglomerative clustering**, you start with each data item being a cluster, and then merge clusters recursively to yield a good clustering (procedure 12.2). The difficulty here is that we need to know a good way to measure the distance between clusters, which can be somewhat harder than the distance between points. In **divisive clustering**, you start with the entire data set being a cluster, and then split clusters recursively to yield a good clustering (procedure **??**). The difficulty here is we need to know some criterion for splitting clusters.

Procedure: 12.1 *Agglomerative Clustering*

Choose an inter-cluster distance. Make each point a separate cluster. Now, until the clustering is satisfactory,

- Merge the two clusters with the smallest inter-cluster distance.

Procedure: 12.2 *Divisive Clustering*

Choose a splitting criterion. Regard the entire dataset as a single cluster. Now, until the clustering is satisfactory,

- choose a cluster to split;
- then split this cluster into two parts.

To turn these recipes into algorithms requires some more detail. For agglomerative clustering, we need to choose a good inter-cluster distance to fuse nearby clusters. Even if a natural distance between data points is available, there is no canonical inter-cluster distance. Generally, one chooses a distance that seems appropriate for the data set. For example, one might choose the distance between the closest elements as the inter-cluster distance, which tends to yield extended clusters (statisticians call this method **single-link clustering**). Another natural choice is the maximum distance between an element of the first cluster and one of the second, which tends to yield rounded clusters (statisticians call this method **complete-link clustering**). Finally, one could use an average of distances between elements in the cluster, which also tends to yield rounded clusters (statisticians call this method **group average clustering**).

For divisive clustering, we need a splitting method. This tends to be something that follows from the logic of the application, because the ideal is an efficient method to find a natural split in a large dataset. We won't pursue this question further.

Finally, we need to know when to stop. This is an intrinsically difficult task if there is no model for the process that generated the clusters. The recipes I have described generate a hierarchy of clusters. Usually, this hierarchy is displayed to a user in the form of a **dendrogram**—a representation of the structure of the hierarchy of clusters that displays inter-cluster distances—and an appropriate choice of clusters is made from the dendrogram (see the example in Figure 12.1).

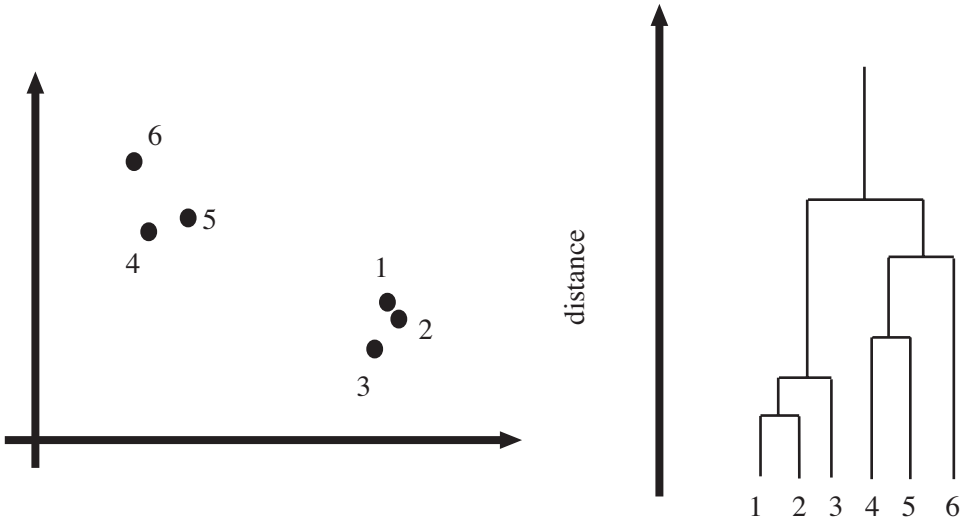

FIGURE 12.1: **Left**, *a data set;* **right**, *a dendrogram obtained by agglomerative clustering using single-link clustering. If one selects a particular value of distance, then a horizontal line at that distance splits the dendrogram into clusters. This representation makes it possible to guess how many clusters there are and to get some insight into how good the clusters are.*

Another important thing to notice about clustering from the example of figure 12.1 is that there is no right answer. There are a variety of different clusterings of the same data. For example, depending on what scales in that figure mean, it might be right to zoom out and regard all of the data as a single cluster, or to zoom in and regard each data point as a cluster. Each of these representations may be useful.

12.3.1 Clustering and Distance

In the algorithms above, and in what follows, we assume that the features are scaled so that distances (measured in the usual way) between data points are a good representation of their similarity. This is quite an important point. For example, imagine we are clustering data representing brick walls. The features might contain several distances: the spacing between the bricks, the length of the wall, the height of the wall, and so on. If these distances are given in the same set of units, we could have real trouble. For example, assume that the units are centimeters. Then the spacing between bricks is of the order of one or two centimeters, but the heights of the walls will be in the hundreds of centimeters. In turn, this means that the distance between two datapoints is likely to be completely dominated by the height and length data. This could be what we want, but it might also not be a good thing.

There are some ways to manage this issue. One is to know what the features measure, and know how they should be scaled. Usually, this happens because you have a deep understanding of your data. If you don't (which happens!), then it is often a good idea to try and normalize the scale of the data set. There are two good strategies. The simplest is to translate the data so that it has zero mean (this is just for neatness - translation doesn't change distances), then scale each direction so that it has unit variance. More sophisticated is to translate the data so that it has zero mean, then transform it so that each direction is independent and has unit variance. Doing so is sometimes referred to as **decorrelation** or **whitening** (because you make the data more like white noise); I described how to do this in section **??**.

Worked example 12.1 *Agglomerative clustering*

Cluster the seed dataset from the UC Irvine Machine Learning Dataset Repository (you can find it at http://archive.ics.uci.edu/ml/datasets/seeds).

Solution: Each item consists of seven measurements of a wheat kernel; there are three types of wheat represented in this dataset. For this example, I used Matlab, but many programming environments will provide tools that are useful for agglomerative clustering. I show a dendrogram in figure **??**). I deliberately forced Matlab to plot the whole dendrogram, which accounts for the crowded look of the figure (you can allow it to merge small leaves, etc.). As you can see from the dendrogram and from Figure 12.3, this data clusters rather well.

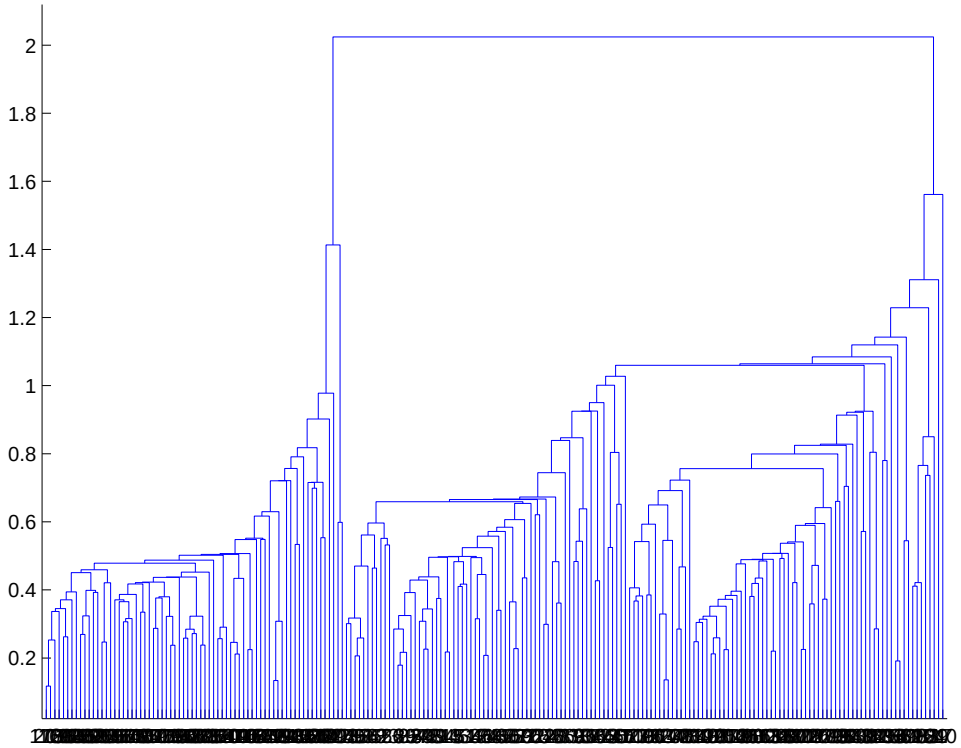

FIGURE 12.2: *A dendrogram obtained from the seed dataset, using single link clustering. Recall that the data points are on the horizontal axis, and that the vertical axis is distance; there is a horizontal line linking two clusters that get merged, established at the height at which they're merged. I have plotted the entire dendrogram, despite the fact it's a bit crowded at the bottom, because you can now see how clearly the data set clusters into a small set of clusters — there are a small number of vertical "runs".*

Remember this: *Agglomerative clustering starts with each data point a cluster, then recursively merges. There are three main ways to compute the distance between clusters. Divisive clustering starts with all in one cluster, then recursively splits. Choosing a split can be tricky.*

12.4 THE K-MEANS ALGORITHM AND VARIANTS

Assume we have a dataset that, we believe, forms many clusters that look like blobs. If we knew where the center of each of the clusters was, it would be easy to

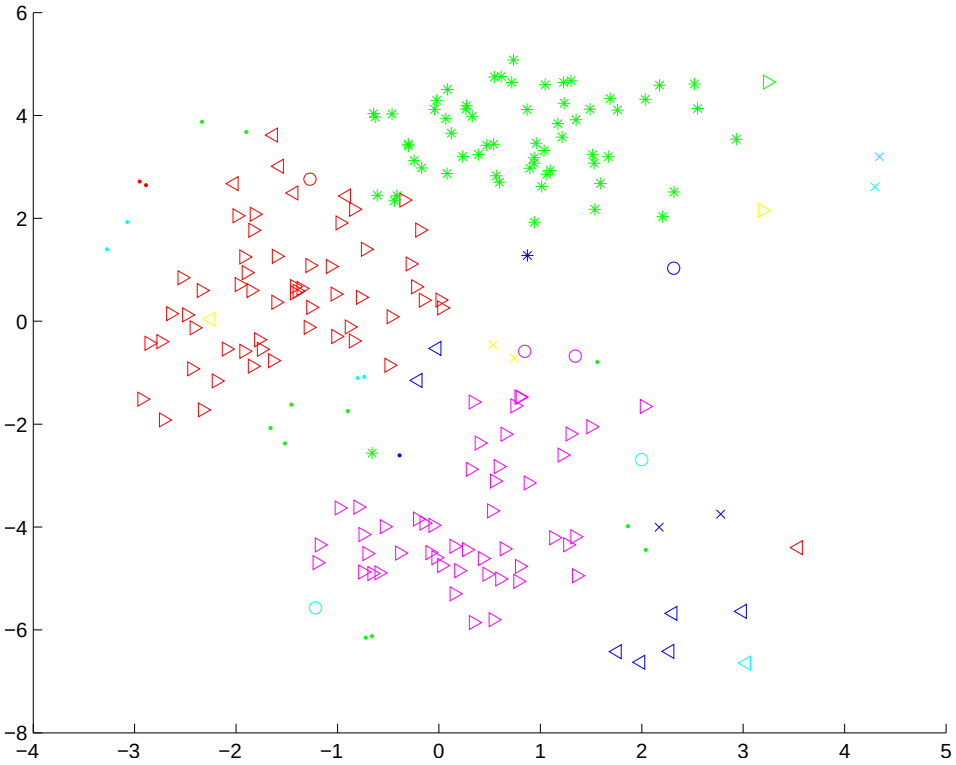

FIGURE 12.3: *A clustering of the seed dataset, using agglomerative clustering, single link distance, and requiring a maximum of 30 clusters. I have plotted each cluster with a distinct marker (though some markers differ only by color; you might need to look at the PDF version to see this figure at its best). Notice that there are a set of fairly natural isolated clusters. The original data is 8 dimensional, which presents plotting problems; I show a scatter plot on the first two principal components (though I computed distances for clustering in the original 8 dimensional space).*

tell which cluster each data item belonged to — it would belong to the cluster with the closest center. Similarly, if we knew which cluster each data item belonged to, it would be easy to tell where the cluster centers were — they'd be the mean of the data items in the cluster. This is the point closest to every point in the cluster.

We can turn these observations into an algorithm. Assume that we know how many clusters there are in the data, and write k for this number. The ith data item to be clustered is described by a feature vector $\mathbf{x}_i$. We write $\mathbf{c}_j$ for the center of the jth cluster. We assume that most data items are close to the center of their cluster. This suggests that we cluster the data by minimizing the the cost function

$$\Phi(\text{clusters}, \text{data}) = \sum_{j \in \text{clusters}} \left\{ \sum_{i \in i\text{th cluster}} (\mathbf{x}_i - \mathbf{c}_j)^T (\mathbf{x}_i - \mathbf{c}_j) \right\}.$$

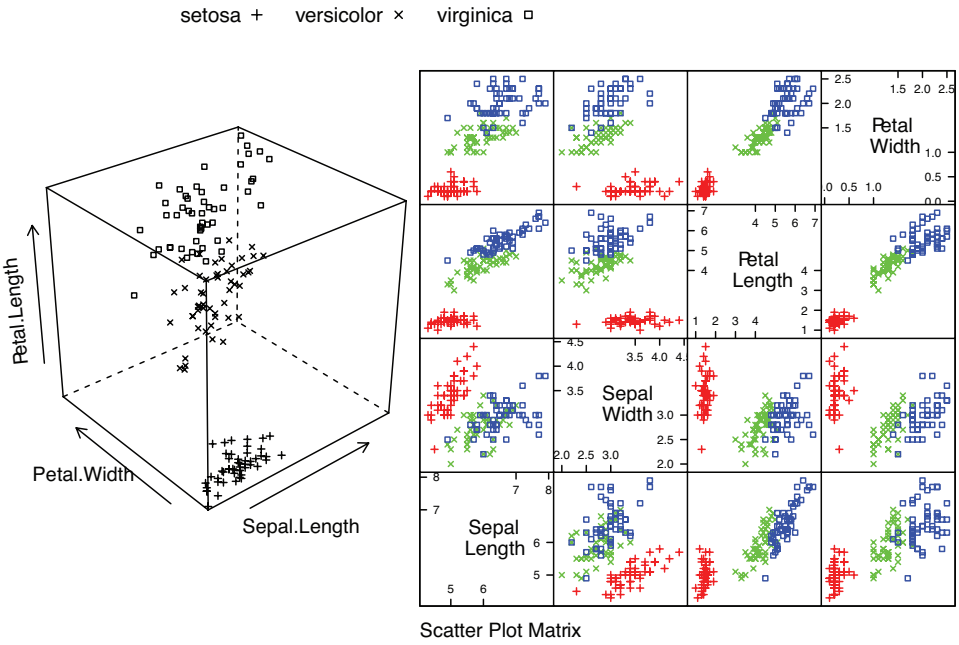

FIGURE 12.4: **Left:** *a 3D scatterplot for the famous Iris data, collected by Edgar Anderson in 1936, and made popular amongst statisticians by Ronald Fisher in that year. I have chosen three variables from the four, and have plotted each species with a different marker. You can see from the plot that the species cluster quite tightly, and are different from one another.* **Right:** *a scatterplot matrix for the Iris data. There are four variables, measured for each of three species of iris. I have plotted each species with a different marker. You can see from the plot that the species cluster quite tightly, and are different from one another.*

Notice that if we know the center for each cluster, it is easy to determine which cluster is the best choice for each point. Similarly, if the allocation of points to clusters is known, it is easy to compute the best center for each cluster. However, there are far too many possible allocations of points to clusters to search this space for a minimum. Instead, we define an algorithm that iterates through two activities:

- Assume the cluster centers are known and, allocate each point to the closest cluster center.
- Assume the allocation is known, and choose a new set of cluster centers. Each center is the mean of the points allocated to that cluster.

We then choose a start point by randomly choosing cluster centers, and then iterate these stages alternately. This process eventually converges to a local minimum of the objective function (the value either goes down or is fixed at each step, and it is bounded below). It is not guaranteed to converge to the global minimum of the objective function, however. It is also not guaranteed to produce k clusters,

unless we modify the allocation phase to ensure that each cluster has some nonzero number of points. This algorithm is usually referred to as **k-means** (summarized in Algorithm 12.3).

Procedure: 12.3 *K-Means Clustering*

Choose k. Now choose k data points $\mathbf{c}_j$ to act as cluster centers. Until the cluster centers change very little

- Allocate each data point to cluster whose center is nearest.
- Now ensure that every cluster has at least one data point; one way to do this is by supplying empty clusters with a point chosen at random from points far from their cluster center.
- Replace the cluster centers with the mean of the elements in their clusters.

Usually, we are clustering high dimensional data, so that visualizing clusters can present a challenge. If the dimension isn't too high, then we can use panel plots. An alternative is to project the data onto two principal components, and plot the clusters there; the process for plotting 2D covariance ellipses from section 12.2.2 comes in useful here. A natural dataset to use to explore k-means is the iris data, where we know that the data should form three clusters (because there are three species). Recall this dataset from section **??**. I reproduce figure 10.3 from that section as figure 12.8, for comparison. Figures 12.5, **??** and **??** show different k-means clusterings of that data.

Worked example 12.2 *K-means clustering in R*

Cluster the iris dataset into two clusters using k-means, then plot the results on the first two principal components

Solution: I used the code fragment in listing 12.1, which produced figure **??**

12.4.1 How to choose K

The iris data is just a simple example. We know that the data forms clean clusters, and we know there should be three of them. Usually, we don't know how many clusters there should be, and we need to choose this by experiment. One strategy is to cluster for a variety of different values of k, then look at the value of the cost function for each. If there are more centers, each data point can find a center that is close to it, so we expect the value to go down as k goes up. This means that

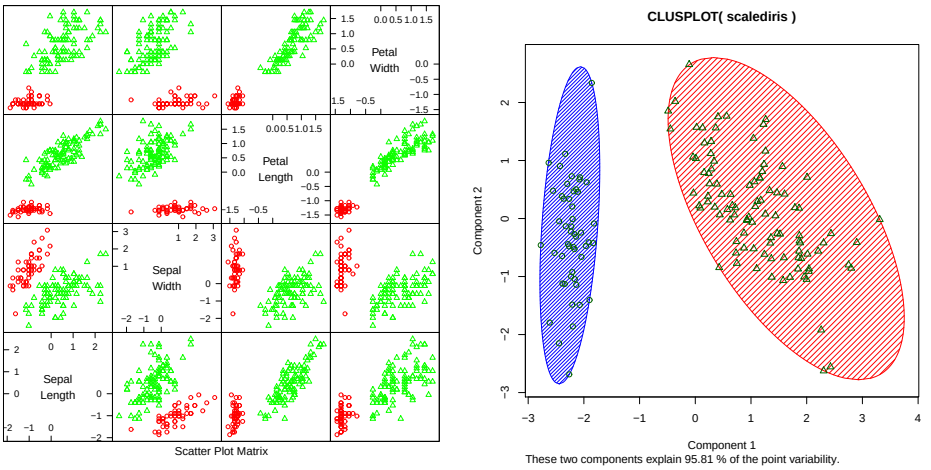

FIGURE 12.5: *On the* **left**, *a panel plot of the iris data clustered using k-means with* $k = 2$. *By comparison with figure 12.8, notice how the* versicolor *and* verginica *clusters appear to have been merged. On the* **right**, *this data set projected onto the first two principal components, with one blob drawn over each cluster.*

looking for the k that gives the smallest value of the cost function is not helpful, because that k is always the same as the number of data points (and the value is then zero). However, it can be very helpful to plot the value as a function of k, then look at the "knee" of the curve. Figure 12.8 shows this plot for the iris data. Notice that $k = 3$ — the "true" answer — doesn't look particularly special, but $k = 2$, $k = 3$, or $k = 4$ all seem like reasonable choices. It is possible to come up with a procedure that makes a more precise recommendation by penalizing clusterings that use a large k, because they may represent inefficient encodings of the data. However, this is often not worth the bother.

Listing 12.1: R code for iris example.

```
setwd('/users/daf/Current/courses/Probcourse/Clustering/RCode')
#library('lattice')
# work with iris dataset this is famous, and included in R
# there are three species
head(iris)
#
library('cluster')
numiris=iris[,c(1, 2, 3, 4)]  #the numerical values
#scalediris<-scale(numiris) # scale to unit variance
nclus<-2
sfit<-kmeans(numiris, nclus)
colr<-c('red', 'green', 'blue', 'yellow', 'orange')
clusplot(numiris, sfit$cluster, color=TRUE, shade=TRUE,
         labels=0, lines=0)
```

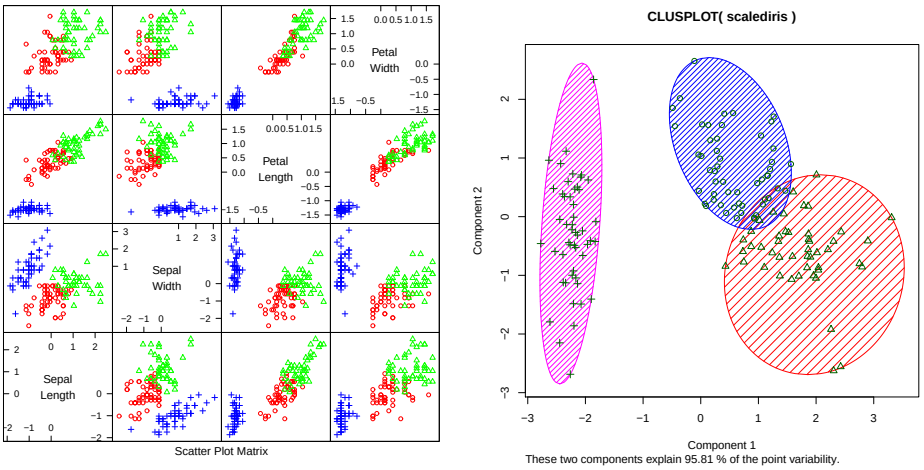

FIGURE 12.6: *On the* **left**, *a panel plot of the iris data clustered using k-means with* $k = 3$. *By comparison with figure 12.8, notice how the clusters appear to follow the species labels. On the* **right**, *this data set projected onto the first two principal components, with one blob drawn over each cluster.*

In some special cases (like the iris example), we might know the right answer to check our clustering against. In such cases, one can evaluate the clustering by looking at the number of different labels in a cluster (sometimes called the purity), and the number of clusters. A good solution will have few clusters, all of which have high purity. Mostly, we don't have a right answer to check against. An alternative strategy, which might seem crude to you, for choosing k is extremely important in practice. Usually, one clusters data to use the clusters in an application (one of the most important, vector quantization, is described in section 12.6). There are usually natural ways to evaluate this application. For example, vector quantization is often used as an early step in texture recognition or in image matching; here one can evaluate the error rate of the recognizer, or the accuracy of the image matcher. One then chooses the k that gets the best evaluation score on validation data. In this view, the issue is not how good the clustering is; it's how well the system that uses the clustering works.

12.4.2 Soft Assignment

One difficulty with k-means is that each point must belong to exactly one cluster. But, given we don't know how many clusters there are, this seems wrong. If a point is close to more than one cluster, why should it be forced to choose? This reasoning suggests we assign points to cluster centers with weights.

We allow each point to carry a total weight of 1. In the conventional k-means algorithm, it must choose a single cluster, and assign its weight to that cluster alone. In soft-assignment k-means, the point can allocate some weight to each cluster center, as long as (a) the weights are all non-negative and (b) the weights sum to one. Write $w_{i,j}$ for the weight connecting point i to cluster center j. We

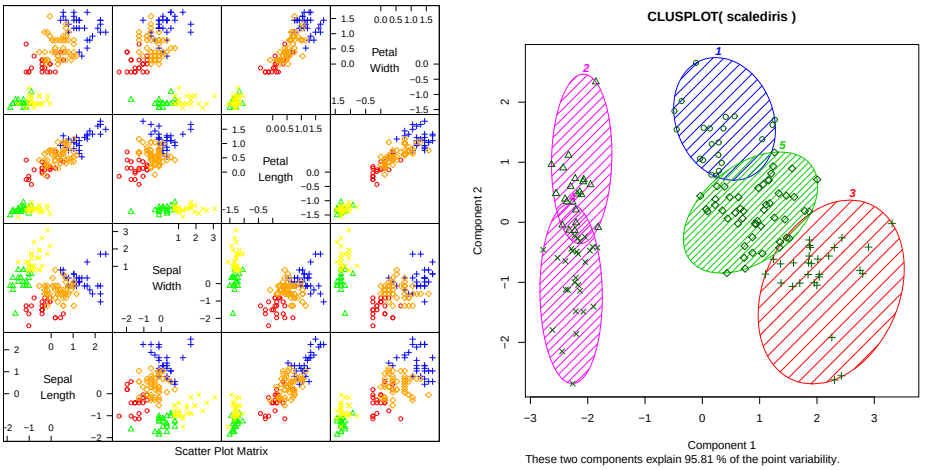

FIGURE 12.7: *On the* **left**, *a panel plot of the iris data clustered using k-means with* $k = 5$. *By comparison with figure 12.8, notice how* setosa *seems to have been broken in two groups, and* versicolor *and* verginica *into a total of three . On the* **right**, *this data set projected onto the first two principal components, with one blob drawn over each cluster.*

interpret these weights as the degree to which the point participates in a particular cluster. We require $w_{i,j} \geq 0$ and $\sum_j w_{i,j} = 1$.

We would like $w_{i,j}$ to be large when $\mathbf{x}_i$ is close to $\mathbf{c}_j$, and small otherwise. Write $d_{i,j}$ for the distance $\|\mathbf{x}_i - \mathbf{c}_j\|$ between these two. Write

$$s_{i,j} = e^{\frac{-d_{i,j}^2}{2\sigma^2}}$$

where $\sigma > 0$ is a choice of scaling parameter. This is often called the affinity between the point i and the center j. Now a natural choice of weights is

$$w_{i,j} = \frac{s_{i,j}}{\sum_{l=1}^{k} s_{i,l}}.$$

All these weights are non-negative, they sum to one, and the weight is large if the point is much closer to one center than to any other. The scaling parameter σ sets the meaning of "much closer" — we measure distance in units of σ.

Once we have weights, re-estimating the cluster centers is easy. We use a weights to compute a weighted average of the points. In particular, we re-estimate the j'th cluster center by

$$\frac{\sum_i w_{i,j}\mathbf{x}_i}{\sum_i w_{i,j}}.$$

Notice that k-means is a special case of this algorithm where σ limits to zero. In this case, each point has a weight of one for some cluster, and zero for all others, and the weighted mean becomes an ordinary mean. I have collected the description into a box (procedure 12.4) for convenience.

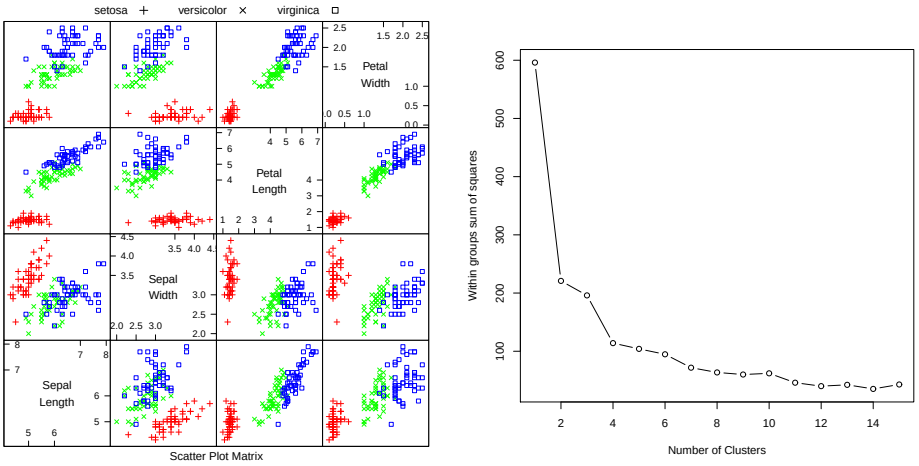

FIGURE 12.8: *On the* **left**, *the scatterplot matrix for the Iris data, for reference. On the* **right**, *a plot of the value of the cost function for each of several different values of* k. *Notice how there is a sharp drop in cost going from* $k = 1$ *to* $k = 2$, *and again at* $k = 4$; *after that, the cost falls off slowly. This suggests using* $k = 2$, $k = 3$, *or* $k = 4$, *depending on the precise application.*

Procedure: 12.4 *K-Means with Soft Weights*

Choose k. Choose k data points $\mathbf{c}_j$ to act as initial cluster centers. Until the cluster centers change very little:

- First, we estimate the weights. For each pair of a data point $\mathbf{x}_i$ and a cluster $\mathbf{c}_j$, compute the affinity

$$s_{i,j} = e^{\frac{-\|\mathbf{x}_i - \mathbf{c}_j\|}{2\sigma^2}}.$$

- Now for each pair of a data point $\mathbf{x}_i$ and a cluster $\mathbf{c}_j$ compute the soft weight linking the data point to the center

$$w_{i,j} = s_{i,j} / \sum_{l=1} k s_{i,l}.$$

- For each cluster, compute a new center

$$\mathbf{c}_j = \frac{\sum_i w_{i,j}\mathbf{x}_i}{\sum_i w_{i,j}}$$

12.4.3 General Comments on K-Means

If you experiment with k-means, you will notice one irritating habit of the algorithm. It almost always produces either some rather spread out clusters, or some single element clusters. Most clusters are usually rather tight and blobby clusters, but there is usually one or more bad cluster. This is fairly easily explained. Because every data point must belong to some cluster, data points that are far from all others (a) belong to some cluster and (b) very likely "drag" the cluster center into a poor location. This applies even if you use soft assignment, because every point must have total weight one. If the point is far from all others, then it will be assigned to the closest with a weight very close to one, and so may drag it into a poor location, or it will be in a cluster on its own.

There are ways to deal with this. If k is very big, the problem is often not significant, because then you simply have many single element clusters that you can ignore. It isn't always a good idea to have too large a k, because then some larger clusters might break up. An alternative is to have a junk cluster. Any point that is too far from the closest true cluster center is assigned to the junk cluster, and the center of the junk cluster is not estimated. Notice that points should not be assigned to the junk cluster permanently; they should be able to move in and out of the junk cluster as the cluster centers move.

12.4.4 K-Mediods

In some cases, we want to cluster objects that can't be averaged. For example, you can compute distances between two trees but you can't meaningfully average two trees. One case where this happens is when you have a table of distances between objects, but do not know vectors representing the objects. For example, you could collect data giving the distances between cities, without knowing where the cities are (as in Section 10.4.3, particularly Figure 10.16), then try and cluster using this data. As another example, you could collect data giving similarities between breakfast items as in Section 10.4.3, then turn the similarities into distances by taking the negative logarithm. This gives a useable table of distances. You still can't average kippers with oatmeal, so you couldn't use k-means to cluster this data. A variant of k-means, known as k-medoids, applies to this case.

In k-medoids, the cluster centers are data items rather than averages, and so are called "mediods". The rest of the algorithm has a familiar form. We assume k, the number of cluster centers, is known. We initialize the cluster centers by choosing examples at random. We then iterate two procedures. In the first, we allocate each data point to the closest mediod. In the second, we choose the best medoid for each cluster by finding the data point that minimizes the sum of distances of points in the cluster to that medoid. This point can be found by simply searching all points.

Remember this: *K-means clustering is the "go-to" clustering algorithm. You should see it as a basic recipe from which many algorithms can be concocted. The recipe is: iterate: allocate each data point to the closest cluster center; re-estimate cluster centers from their data points. There are many variations, improvements, etc. that are possible on this recipe. We have seen soft weights and k-mediods. K-means is not usually best implemented with the method I described (which isn't particularly efficient, but gets to the heart of what is going on). Implementations of k-means differ in important ways from my rather high-level description of the algorithm; for any but tiny problems, you should use a package, and you should look for a package that uses the Lloyd-Hartigan method.*

12.5 APPLICATION EXAMPLE: CLUSTERING DOCUMENTS

It is really useful to be able to cluster together documents that are "similar". We cannot do so with k-means as we currently understand it, because we do not have a distance between documents. But the k-means recipe (i.e. iterate: allocate each data point to the closest cluster center; re-estimate cluster centers from their data points) is spectacularly flexible and powerful. I will demonstrate an application of this recipe to text clustering here.

For many kinds of document, we obtain a good representation by (a) choosing a vocabulary (a list of different words) then (b) representing the document by a vector of word counts, where we simply ignore every word outside the vocabulary. This is a viable representation for many applications because quite often, most of the words people actually use come from a relatively short list (typically 100s to 1000s, depending on the particular application). The vector has one component for each word in the list, and that component contains the number of times that particular word is used. This model is sometimes known as a **bag-of-words** model.

We could try to cluster on the distance between word vectors, but this turns out to be a poor idea. This is because quite small changes in word use might lead to large differences between count vectors. For example, some authors might write "car" when others write "auto". In turn, two documents might have a large (resp. small) count for "car" and a small (resp. large) count for "auto". Just looking at the counts would significantly overstate the difference between the vectors. However, the counts are informative: a document that uses the word "car" often, and the word "lipstick" seldom, is likely quite different from a document that uses "lipstick" often and "car" seldom.

Details of how you put the vocabulary together can be quite important. It is not a good idea to count extremely common words, sometimes known as **stop words**, because every document has lots of them and the counts don't tell you very much. Typical stop words include "and", "the", "he", "she", and so on. These are left out of the vocabulary. Notice that the choice of stop words can be quite important, and depends somewhat on the application. It's often, but not

always, helpful to stem words – a process that takes "winning" to "win", "hugely" to "huge", and so on. This can create confusion (for example, a search for "stock" may be looking for quite different things than a search for "stocking"). We will always use datasets that have been preprocessed to produce word counts, but you should be aware that pre-processing this data is hard and involves choices that can have significant effects on the application.

One alternative – which is quite a good idea, but has now been superseded by better ideas – is to compute small set of principal components for the word vectors, project each onto its principal components, then work with distances in the low dimensional space spanned by these principal components. This turns out to work rather well, because the projection onto principal components ensures that a document that uses "car" a lot will tend to end up close to one that uses "auto" a lot. It should be fairly clear to you roughly how to make this method work. We will look at an alternative, that has led to systems that can cluster fairly large collections of documents (of the scale of, say, most of the 20'th century's scientific literature in English).

12.5.1 A Topic Model

We get a useful notion of the differences between documents by pretending that the count vector for each document comes from one of a small set of underlying topics. Each topic generates words as independent, identically distributed samples from a multinomial distribution, with one probability per word in the vocabulary. Each topic will be a cluster center. If two documents come from the same topic, they should have "similar" word distributions. Topics are one way to deal with changes in word use. For example, one topic might have quite high probability of generating the word "car" *and* a high probability of generating the word "auto"; another might have low probability of generating those words, but a high probability of generating "lipstick".

Now think about the key elements of the k-means algorithm. First, we need to be able to tell the distance between any data item and any cluster center. Second, we need to be able to come up with a cluster center for any collection of data items. We can do each of these with our topic model, by thinking about it as a probabilistic model of how a document is generated. We will have t topics. Each document will be a vector of word counts. We will have N vectors of word counts (i.e. documents), and write $\mathbf{x}_i$ for the i'th such vector. To generate a document, we first choose a topic, choosing the j'th topic with probability π_j. Then we will obtain a set of words for the document by repeatedly drawing IID samples from that topic, and recording the count of each word in a count vector.

Each topic is a multinomial probability distribution. Write $\mathbf{p}_j$ for the vector of word probabilities for the j'th topic. We assume that words are generated independently, conditioned on the topic. Write x_{ik} for the k'th component of $\mathbf{x}_i$, and so on. Notice that $\mathbf{x}_i^T\mathbf{1}$ is the sum of entries in $\mathbf{x}_i$, and so the number of words in document i.

Distance from document to topic: The probability of observing the

counts in $\mathbf{x}_i$ when the document was generated by topic j is

$$p(\mathbf{x}_i|\mathbf{p}_j) = \left(\frac{(\mathbf{x}_i^T\mathbf{1})!}{\prod_v x_{iv}!} \right) \prod_u p_{ju}^{x_{iu}}.$$

This number will be non-negative, but less than one. It will be big when the document is close to the topic, and small otherwise. We can obtain a distance from this expression by taking the negative log of the probability. This will be small when the document is close to the topic, and big otherwise.

Turning a set of documents into topic probabilities: Now assume we have a set of documents that we assert belong to topic j. We must estimate $\mathbf{p}_j$. We can do this by counting, because we assumed that each word was generated independent of all others, given the topic. One difficulty we will run into is that some (likely, most) counts will be zero, because most words are rare. But the fact you don't see a word in all the documents in the topic doesn't mean it never occurs. We've seen this issue before (section **??**). You should not allow any element of $\mathbf{p}_j$ to be zero; one way to deal with this problem is to add some small number (likely less than one) to each count.

Initialization: One possible initialization is to take a subset of documents, allocate elements randomly to clusters, then compute initial cluster centers.

We now have a k-means algorithm adapted to clustering documents. This isn't the state of the art, by any manner of means. The next step would be to compute soft weights linking a document to every cluster, then re-estimating the cluster centers taking the soft weights into account. This involves some fairly alarming equations (though nothing difficult or interesting happens), and would take us out of our way.

12.6 DESCRIBING REPETITION WITH VECTOR QUANTIZATION

Repetition is an important feature of many interesting signals. For example, images contain *textures*, which are orderly patterns that look like large numbers of small structures that are repeated. Examples include the spots of animals such as leopards or cheetahs; the stripes of animals such as tigers or zebras; the patterns on bark, wood, and skin. Similarly, speech signals contain *phonemes* — characteristic, stylised sounds that people assemble together to produce speech (for example, the "ka" sound followed by the "tuh" sound leading to "cat"). Another example comes from accelerometers. If a subject wears an accelerometer while moving around, the signals record the accelerations during their movements. So, for example, brushing one's teeth involves a lot of repeated twisting movements at the wrist, and walking involves swinging the hand back and forth.

Repetition occurs in subtle forms. The essence is that a small number of local patterns can be used to represent a large number of examples. You see this effect in pictures of scenes. If you collect many pictures of, say, a beach scene, you will expect most to contain some waves, some sky, and some sand. The individual patches of wave, sky or sand can be surprisingly similar, and different images are made by selecting some patches from a vocabulary of patches, then placing them down to form an image. Similarly, pictures of living rooms contain chair patches, TV patches, and carpet patches. Many different living rooms can be made from

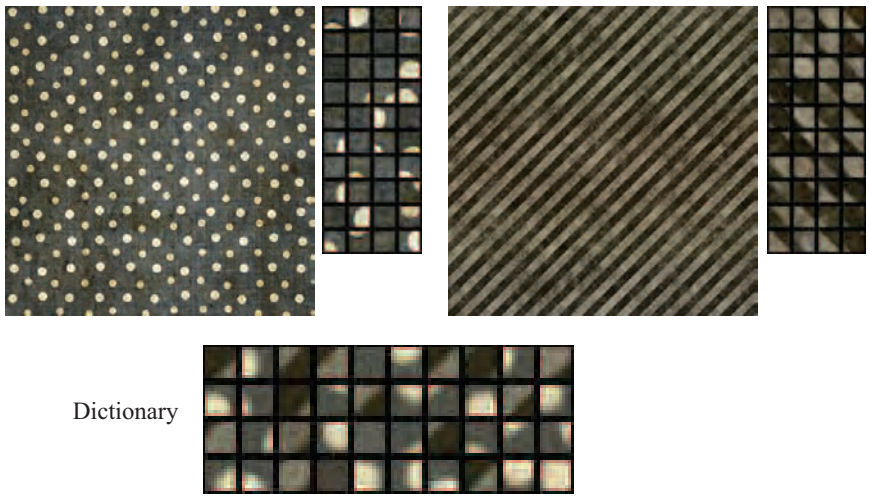

FIGURE 12.9: *It is hard to hide repetition in a signal, meaning that you don't need to be careful about whether patches overlap, etc. when vector quantizing a signal.* **Top:** *two images with rather exaggerated repetion, published on flickr.com with a creative commons license by* `webtreats`. *Next to these images, I have placed zoomed sampled 10x10 patches from those images; although the spots (resp. stripes) aren't necessarily centered in the patches, it's pretty clear which image each patch comes from.* **Bottom:** *a 40 patch dictionary computed using k-means from 4000 samples from each image. If you look closely, you'll see that some dictionary entries are clearly stripe entries, others clearly spot entries. Stripe images will have patches represented by stripe entries, spot images by spot entries; so the histogram construction in the text should make differences apparent.*

small vocabularies of patches; but you won't often see wave patches in living rooms, or carpet patches in beach scenes.

An important part of representing signals that repeat is building a vocabulary of patterns that repeat, then describing the signal in terms of those patterns. For many problems, problems, knowing what vocabulary elements appear and how often is much more important than knowing where they appear. For example, if you want to tell the difference between zebras and leopards, you need to know whether stripes or spots are more common, but you don't particularly need to know where they appear. As another example, if you want to tell the difference between brushing teeth and walking using accelerometer signals, knowing that there are lots of (or few) twisting movements is important, but knowing how the movements are linked together in time may not be.

12.6.1 Vector Quantization

It is natural to try and find patterns by looking for small pieces of signal of fixed size that appear often. In an image, a piece of signal might be a 10x10 patch; in a sound file, which is likely represented as a vector, it might be a subvector of fixed size.

A 3-axis accelerometer signal is represented as a $3 \times r$ dimensional array (where r is the number of samples); in this case, a piece might be a 3×10 subarray. But finding patterns that appear often is hard to do, because the signal is continuous — each pattern will be slightly different, so we cannot simply count how many times a particular pattern occurs.

Here is a strategy. First, we take a training set of signals, and cut each signal into pieces of fixed size. We could use d dimensional vectors for a sound file; $d \times d$ dimensional patches for an image; or $3 \times d$ dimensional subarrays for an accelerometer signal. In each case, it is easy to compute the distance between two pieces: you use the sum of squared distances. It doesn't seem to matter too much if these pieces overlap or not. We then build a set of clusters out of these pieces. This set of clusters is often thought of as a dictionary. We can now now describe any new piece with the cluster center closest to that piece. This means that a piece of signal is described with a number in the range $[1, \ldots, k]$ (where you get to choose k), and two pieces that are close should be described by the same number. This strategy is known as **vector quantization**.

Procedure: 12.5 *Vector Quantization - Building a Dictionary*

Take a training set of signals, and cut each signal into pieces of fixed size. The size of the piece will affect how well your method works, and is usually chosen by experiment. It does not seem to matter much if the pieces overlap. Cluster all the example pieces, and record the k cluster centers. It is usual, but not required, to use k-means clustering.

We can now build features that represent important repeated structure in signals. We take a signal, and cut it up into vectors of length d. These might overlap, or be disjoint; we follow whatever strategy we used in building the dictionary. We then take each vector, and compute the number that describes it (i.e. the number of the closest cluster center, as above). We then compute a histogram of the numbers we obtained for all the vectors in the signal. This histogram describes the signal.

Procedure: 12.6 *Vector Quantization - Representing a Signal*

Take your signal, and cut it into pieces of fixed size. The size of the piece will affect how well your method works, and is usually chosen by experiment. It does not seem to matter much if the pieces overlap. For each piece, record the closest cluster center in the dictionary. Represent the signal with a histogram of these numbers, which will be a k dimensional vector.

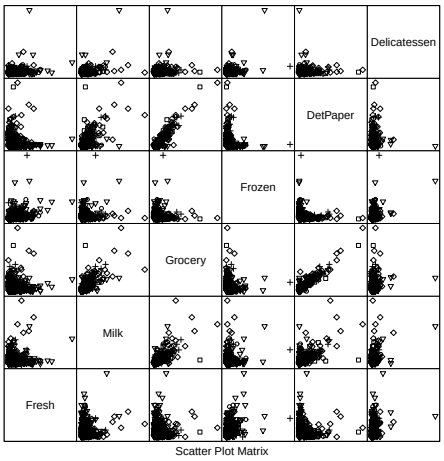

FIGURE 12.10: *A panel plot of the wholesale customer data of http://archive.ics.uci.edu/ml/datasets/Wholesale+customers, which records sums of money spent annually on different commodities by customers in Portugal. This data is recorded for six different groups (two channels each within three regions). I have plotted each group with a different marker, but you can't really see much structure here, for reasons explained in the text.*

Notice several nice features to this construction. First, it can be applied to anything that can be thought of in terms of fixed size pieces, so it will work for speech signals, sound signals, accelerometer signals, images, and so on. Another nice feature is the construction can accept signals of different length, and produce a description of fixed length. One accelerometer signal might cover 100 time intervals; another might cover 200; but the description is always a histogram with k buckets, so it's always a vector of length k.

Yet another nice feature is that we don't need to be all that careful how we cut the signal into fixed length vectors. This is because it is hard to hide repetition. This point is easier to make with a figure than in text, so look at figure **??**.

12.6.2 Example: Groceries in Portugal

At http://archive.ics.uci.edu/ml/datasets/Wholesale+customers, you will find a dataset giving sums of money spent annually on different commodities by customers in Portugal. The commodities are divided into a set of categories (fresh; milk; grocery; frozen; detergents and paper; and delicatessen) relevant for the study. These customers are divided by channel (two channels) and by region (three regions). You can think of the data as being divided into six groups (one for each pair of channel and region). There are 440 customer records, and there are many customers in each group. Figure 12.10 shows a panel plot of the customer data; the data has been clustered, and I gave each of 20 clusters its own marker.

People tend to like to live near people who are "like" them, so you could expect people in a region to be somewhat similar; you could reasonably expect

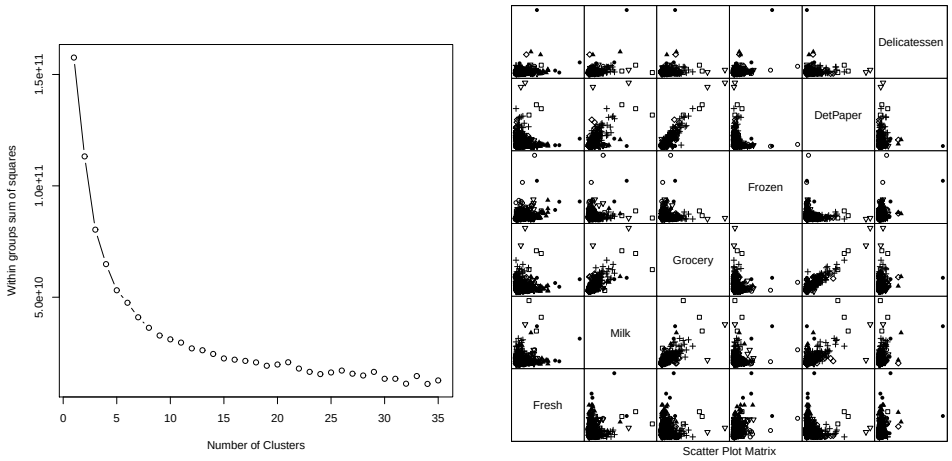

FIGURE 12.11: *On the* **left**, *the cost function (of section 12.4) for clusterings of the customer data with k-means for k running from 2 to 35. This suggests using a k somewhere in the range 10-30; I chose 20. On the* **right**, *I have clustered this data to 20 cluster centers with k-means. The clusters do seem to be squashed together, but the plot on the left suggests that clusters do capture some important information. Using too few clusters will clearly lead to problems. Notice that I did not scale the data, because each of the measurements is in a comparable unit. For example, it wouldn't make sense to scale expenditures on fresh and expenditures on grocery with a different scale.*

differences between regions (regional preferences; differences in wealth; and so on). Retailers have different channels to appeal to different people, so you could expect people using different channels to be different. But you don't see this in the plot of clusters. In fact, the plot doesn't really show much structure.

Here is a way to think about structure in the data. There are likely to be different "types" of customer. For example, customers who prepare food at home might spend more money on fresh or on grocery, and those who mainly buy prepared food might spend more money on delicatessan; similarly, coffee drinkers with cats or with children might spend more on milk than the lactose-intolerant, and so on. So we can expect customers to cluster in types. An effect like this is hard to see on a panel plot (Figure 12.10). The plot for this dataset is hard to read, because the dimension is fairly high for a panel plot and the data is squashed together in the bottom left corner. However, you can see the effect when you cluster the data and look at the cost function in representing the data with different values of k — quite a small set of clusters gives quite a good representation (Figure 12.12). The panel plot of cluster membership (also in that figure) isn't particularly informative. The dimension is quite high, and clusters get squashed together.

There is another effect, which isn't apparent in these plots. Some of what cause customers to cluster in types are driven by things like wealth and the tendency of people to have neighbors who are similar to them. This means that differ-

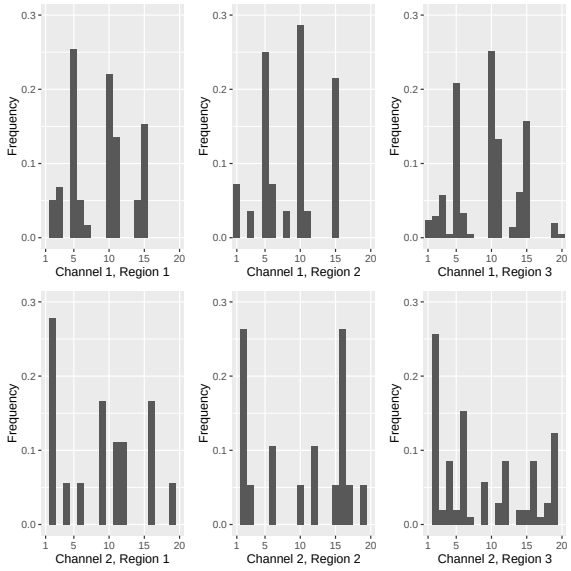

FIGURE 12.12: *The histogram of different types of customer, by group, for the customer data. Notice how the distinction between the groups is now apparent — the groups do appear to contain quite different distributions of customer type. It looks as though the channels (rows in this figure) are more different than the regions (columns in this figure).*

ent groups should have different fractions of each type of customer. There might be more deli-spenders in wealthier regions; more milk-spenders and detergent-spenders in regions where it is customary to have many children; and so on. This sort of structure will not be apparent in a panel plot. A group of a few milk-spenders and many detergent-spenders will have a few data points with high milk expenditure values (and low other values) and also many data points with high detergent expenditure values (and low other values). In a panel plot, this will look like two blobs; but if there is a second group with many milk-spenders and few detergent-spenders will also look like two blobs, lying roughly on top of the first set of blobs. It will be hard to spot the difference between the groups.

An easy way to see this difference is to look at histograms of the types of customer within each group. I described the each group of data by the histogram of customer types that appeared in that group (Figure **??**). Notice how the distinction between the groups is now apparent — the groups do appear to contain quite different distributions of customer type. It looks as though the channels (rows in this figure) are more different than the regions (columns in this figure). To be more confident in this analysis, we would need to be sure that different types of customer really are different. We could do this by repeating the analysis for fewer clusters, or by looking at the similarity of customer types.

12.6.3 Efficient Clustering and Hierarchical K Means

One important difficulty occurs in applications. We might need to have an enormous dataset (millions of image patches are a real possibility), and so a very large k. In this case, k means clustering becomes difficult because identifying which cluster center is closest to a particular data point scales linearly with k (and we have to do this for every data point at every iteration). There are two useful strategies for dealing with this problem.

The first is to notice that, if we can be reasonably confident that each cluster contains many data points, some of the data is redundant. We could randomly subsample the data, cluster that, then keep the cluster centers. This works, but doesn't scale particularly well.

A more effective strategy is to build a hierarchy of k-means clusters. We randomly subsample the data (typically, quite aggressively), then cluster this with a small value of k. Each data item is then allocated to the closest cluster center, and the data in each cluster is clustered again with k-means. We now have something that looks like a two-level tree of clusters. Of course, this process can be repeated to produce a multi-level tree of clusters. It is easy to use this tree to vector quantize a query data item. We vector quantize at the first level. Doing so chooses a branch of the tree, and we pass the data item to this branch. It is either a leaf, in which case we report the number of the leaf, or it is a set of clusters, in which case we vector quantize, and pass the data item down. This procedure is efficient both when one clusters and at run time.

12.6.4 Example: Activity from Accelerometer Data

A complex example dataset appears at https://archive.ics.uci.edu/ml/datasets/Dataset+for+ADL+Recognition+w
This dataset consists of examples of the signal from a wrist mounted accelerometer, produced as different subjects engaged in different activities of daily life. Activities include: brushing teeth, climbing stairs, combing hair, descending stairs, and so on. Each is performed by sixteen volunteers. The accelerometer samples the data at 32Hz (i.e. this data samples and reports the acceleration 32 times per second). The accelerations are in the x, y and z-directions. Figure 12.13 shows the x-component of various examples of toothbrushing.

There is an important problem with using data like this. Different subjects take quite different amounts of time to perform these activities. For example, some subjects might be more thorough tooth-brushers than other subjects. As another example, people with longer legs walk at somewhat different frequencies than people with shorter legs. This means that the same activity performed by different subjects will produce data vectors *that are of different lengths*. It's not a good idea to deal with this by warping time and resampling the signal. For example, doing so will make a thorough toothbrusher look as though they are moving their hands very fast (or a careless toothbrusher look ludicrously slow: think speeding up or slowing down a movie). So we need a representation that can cope with signals that are a bit longer or shorter than other signals.

Another important property of these signals is that all examples of a particular activity should contain repeated patterns. For example, brushing teeth should show fast accelerations up and down; walking should show a strong signal at somewhere

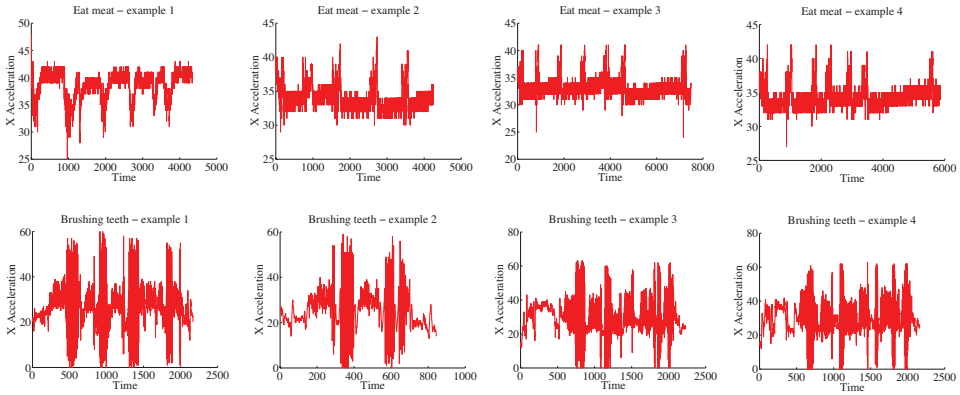

FIGURE 12.13: *Some examples from the accelerometer dataset at https://archive.ics.uci.edu/ml/datasets/Dataset+for+ADL+Recognition+with+Wrist-worn+Accelerometer. I have labelled each signal by the activity. These show acceleration in the X direction (Y and Z are in the dataset, too). There are four examples for* **brushing teeth** *and four for* **eat meat**. *You should notice that the examples don't have the same length in time (some are slower and some faster eaters, etc.), but that there seem to be characteristic features that are shared within a category (brushing teeth seems to involve faster movements than eating meet).*

around 2 Hz; and so on. These two points should suggest vector quantization to you. Representing the signal in terms of stylized, repeated structures is probably a good idea because the signals probably contain these structures. And if we represent the signal in terms of the relative frequency with which these structures occur, the representation will have a fixed length, even if the signal doesn't. To do so, we need to consider (a) over what time scale we will see these repeated structures and (b) how to ensure we segment the signal into pieces so that we see these structures.

Generally, repetition in activity signals is so obvious that we don't need to be smart about segment boundaries. I broke these signals into 32 sample segments, one following the other. Each segment represents one second of activity. This is long enough for the body to do something interesting, but not so long that our representation will suffer if we put the segment boundaries in the wrong place. This resulted in about 40, 000 segments. I then used hierarchical k-means to cluster these segments. I used two levels, with 40 cluster centers at the first level, and 12 at the second. Figure 12.14 shows some cluster centers at the second level.

I then computed histogram representations for different example signals (Figure 12.15). You should notice that when the activity label is different, the histogram looks different, too.

Another useful way to check this representation is to compare the average within class chi-squared distance with the average between class chi-squared distance. I computed the histogram for each example. Then, for each pair of examples, I computed the chi-squared distance between the pair. Finally, for each pair of *activity labels*, I computed the average distance between pairs of examples where one

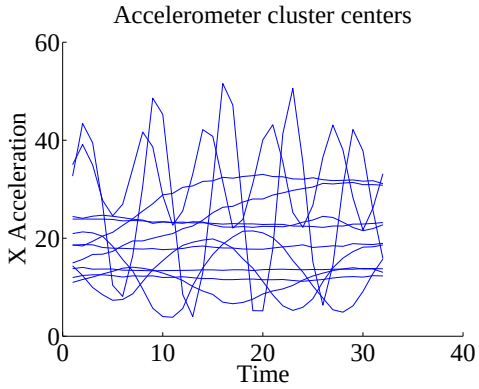

FIGURE 12.14: *Some cluster centers from the accelerometer dataset. Each cluster center represents a one-second burst of activity. There are a total of 480 in my model, which I built using hierarchical k-means. Notice there are a couple of centers that appear to represent movement at about 5Hz; another few that represent movement at about 2Hz; some that look like 0.5Hz movement; and some that seem to represent much lower frequency movement. These cluster centers are samples (rather than chosen to have this property).*

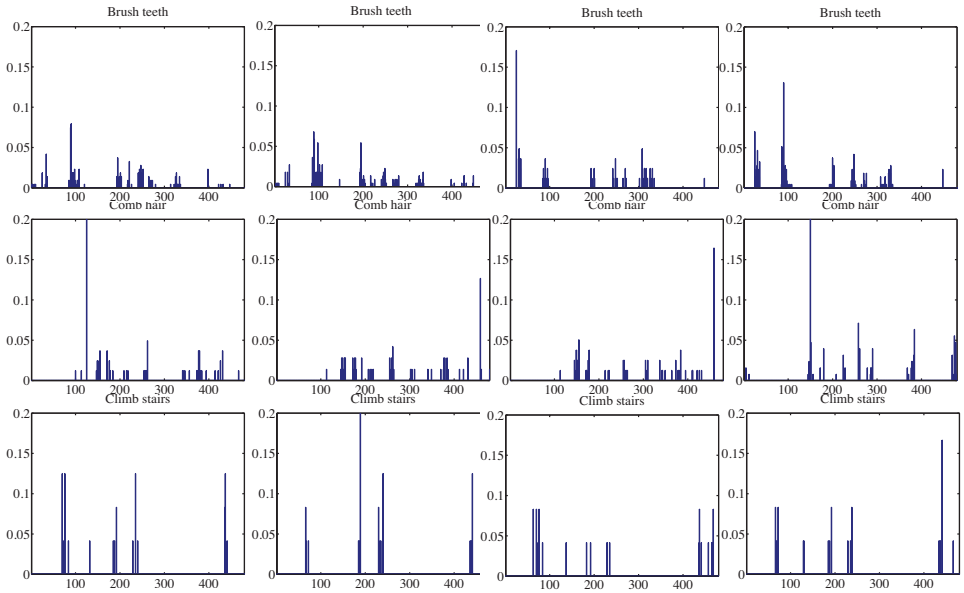

FIGURE 12.15: *Histograms of cluster centers for the accelerometer dataset, for different activities. You should notice that (a) these histograms look somewhat similar for different actors performing the same activity and (b) these histograms look somewhat different for different activities.*

example has one of the activity labels and the other example has the other activity label. In the ideal case, all the examples with the same label would be very close to one another, and all examples with different labels would be rather different. Table 12.1 shows what happens with the real data. You should notice that for some pairs of activity label, the mean distance between examples is smaller than one would hope for (perhaps some pairs of examples are quite close?). But generally, examples of activities with different labels tend to be further apart than examples of activities with the same label.

0.9	2.0	1.9	2.0	2.0	2.0	1.9	2.0	1.9	1.9	2.0	2.0	2.0	2.0
	1.6	2.0	1.8	2.0	2.0	2.0	1.9	1.9	2.0	1.9	1.9	2.0	1.7
		1.5	2.0	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	2.0
			1.4	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	1.8
				1.5	1.8	1.7	1.9	1.9	1.8	1.9	1.9	1.8	2.0
					0.9	1.7	1.9	1.9	1.8	1.9	1.9	1.9	2.0
						0.3	1.9	1.9	1.5	1.9	1.9	1.9	2.0
							1.8	1.8	1.9	1.9	1.9	1.9	1.9
								1.7	1.9	1.9	1.9	1.9	1.9
									1.6	1.9	1.9	1.9	2.0
										1.8	1.9	1.9	1.9
											1.8	2.0	1.9
												1.5	2.0
													1.5

TABLE 12.1: *Each column of the table represents an activity for the activity dataset https://archive.ics.uci.edu/ml/datasets/Dataset+for+ADL+Recognition+with+Wrist-worn+Accelerometer, as does each row. In each of the upper diagonal cells, I have placed the average chi-squared distance between histograms of examples from that pair of classes (I dropped the lower diagonal for clarity). Notice that in general the diagonal terms (average within class distance) are rather smaller than the off diagonal terms. This quite strongly suggests we can use these histograms to classify examples successfully.*

Yet another way to check the representation is to try classification with nearest neighbors, using the chi-squared distance to compute distances. I split the dataset into 80 test pairs and 360 training pairs; using 1-nearest neighbors, I was able to get a held-out error rate of 0.79. This suggests that the representation is fairly good at exposing what is important.

12.7 YOU SHOULD

12.7.1 remember these definitions:

12.7.2 remember these terms:

12.7.3 remember these facts:

12.7.4 remember these procedures:

PROGRAMMING EXERCISES

12.1. You can find a dataset dealing with European employment in 1979 at http://dasl.datadesk.com/data/view/47. This dataset gives the percentage of people employed in each of a set of areas in 1979 for each of a set of European countries.

- **(a)** Use an agglomerative clusterer to cluster this data. Produce a dendrogram of this data for each of single link, complete link, and group average clustering. You should label the countries on the axis. What structure in the data does each method expose? it's fine to look for code, rather than writing your own. **Hint:** I made plots I liked a lot using R's `hclust` clustering function, and then turning the result into a phylogenetic tree and using a fan plot, a trick I found on the web; try `plot(as.phylo(hclustresult), type=''fan'')`. You should see dendrograms that "make sense" (at least if you remember some European history), and have interesting differences.
- **(b)** Using k-means, cluster this dataset. What is a good choice of k for this data and why?

12.2. Obtain the activities of daily life dataset from the UC Irvine machine learning website (https://archive.ics.uci.edu/ml/datasets/Dataset+for+ADL+Recognition+with+Wrist-worn+Acceleron data provided by Barbara Bruno, Fulvio Mastrogiovanni and Antonio Sgorbissa).

- **(a)** Build a classifier that classifies sequences into one of the 14 activities provided. To make features, you should vector quantize, then use a histogram of cluster centers (as described in the subsection; this gives a pretty explicit set of steps to follow). You will find it helpful to use hierarchical k-means to vector quantize. You may use whatever multi-class classifier you wish, though I'd start with R's decision forest, because it's easy to use and effective. You should report (a) the total error rate and (b) the class confusion matrix of your classifier.
- **(b)** Now see if you can improve your classifier by (a) modifying the number of cluster centers in your hierarchical k-means and (b) modifying the size of the fixed length samples that you use.

12.3. This is a fairly ambitious exercise. It will demonstrate how to use vector quantization to handle extremely sparse data. The 20 newsgroups dataset is a famous text dataset. It consists of posts collected from 20 different newsgroups. There are a variety of tricky data issues that this presents (for example, what aspects of the header should one ignore? should one reduce words to their stems, so "winning" goes to "win", "hugely" to "huge", and so on?). We will ignore these issues, and deal with a cleaned up version of the dataset. This consists of three items each for train and test: a document-word matrix, a set of labels, and a map. You can find this cleaned up version of the dataset at http://qwone.com/~jason/20Newsgroups/. You should look for the cleaned up version, identified as `20news-bydate-matlab.tgz` on that page. The usual task is to label a test article with which newsgroup it came from. Instead, we will assume you have a set of test articles, all from the same newsgroup, and you need to identify the newsgroup. The document-word matrix is a table of counts of how many times a particular word appears in a particular document. The collection of words is very large (53975 distinct words), and most words do not appear in most documents, so most entries of this matrix are zero. The file `train.data` contains this matrix for a collection of training data; each row represents a distinct document (there are 11269), and each column represents

a distinct word.

(a) Cluster the rows of this matrix to get a set of cluster centers using k-means. You should have about one center for every 10 documents. Use k-means, and you should find an efficient package rather than using your own implementation. In particular, implementations of k-means differ in important ways from my rather high-level description of the algorithm; you should look for a package that uses the Lloyd-Hartigan method. **Hint:** Clustering all these points is a bit of a performance; check your code on small subsets of the data first, because the size of this dataset means that clustering the whole thing will be slow.

(b) You can now think of each cluster center as a document "type". For each newsgroup, plot a histogram of the "types" of document that appear in the training data for that newsgroup. You'll need to use the file `train.label`, which will tell you what newsgroup a particular item comes from.

(c) Now train a classifier that accepts a small set of documents (10-100) from a single newsgroup, and predicts which of 20 newsgroups it comes from. You should use the histogram of types from the previous sub-exercise as a feature vector. Compute the performance of this classifier on the test data (`test.data` and `test.label`).

12.4. This is another fairly ambitious exercise. We will use the document clustering method of section 12.5 to identify clusters of documents, which we will associate with topics. The 20 newsgroups dataset is a famous text dataset. It consists of posts collected from 20 different newsgroups. There are a variety of tricky data issues that this presents (for example, what aspects of the header should one ignore? should one reduce words to their stems, so "winning" goes to "win", "hugely" to "huge", and so on?). We will ignore these issues, and deal with a cleaned up version of the dataset. This consists of three items each for train and test: a document-word matrix, a set of labels, and a map. You can find this cleaned up version of the dataset at http://qwone.com/~jason/20Newsgroups/. You should look for the cleaned up version, identified as `20news-bydate-matlab.tgz` on that page. The usual task is to label a test article with which newsgroup it came from. The document-word matrix is a table of counts of how many times a particular word appears in a particular document. The collection of words is very large (53975 distinct words), and most words do not appear in most documents, so most entries of this matrix are zero. The file `train.data` contains this matrix for a collection of training data; each row represents a distinct document (there are 11269), and each column represents a distinct word.

(a) Cluster the rows of this matrix, using the method of section 12.5, to get a set of cluster centers which we will identify as topics. **Hint:** Clustering all these points is a bit of a performance; check your code on small subsets of the data first, because the size of this dataset means that clustering the whole thing will be slow.

(b) You can now think of each cluster center as a document "type". Assume you have k clusters (topics). Represent each document by a k-dimensional vector. Each entry of the vector should be the negative log-probability of the document under that cluster model. Now use this information to build a classifier that identifies the newsgroup using the vector. You'll need to use the file `train.label`, which will tell you what newsgroup a particular item comes from. I advise you use a randomized decision forest, but other choices are plausible. Evaluate your classifier usingthe test data

(`test.data` and `test.label`).

CHAPTER 13

Regression

Classification tries to predict a class from a data item. **Regression** tries to predict a value. For example, we know the zip code of a house, the square footage of its lot, the number of rooms and the square footage of the house, and we wish to predict its likely sale price. As another example, we know the cost and condition of a trading card for sale, and we wish to predict a likely profit in buying it and then reselling it. As yet another example, we have a picture with some missing pixels – perhaps there was text covering them, and we want to replace it – and we want to fill in the missing values. As a final example, you can think of classification as a special case of regression, where we want to predict either $+1$ or -1; this isn't usually the best way to classify, however. Predicting values is very useful, and so there are many examples like this.

Some formalities are helpful here. In the simplest case, we have a dataset consisting of a set of N pairs $(\mathbf{x}_i, y_i)$. We want to use the examples we have — the **training examples** — to build a model of the dependence between y and $\mathbf{x}$. This model will be used to predict values of y for new values of $\mathbf{x}$, which are usually called **test examples**. We think of y_i as the value of some function evaluated at $\mathbf{x}_i$, but with some random component. This means there might be two data items where the $\mathbf{x}_i$ are the same, and the y_i are different. We refer to the $\mathbf{x}_i$ as **explanatory variables** and the y_i as a **dependent variable**. We regularly say that we are regressing the dependent variable against the explanatory variables.

13.0.1 Regression to Make Predictions

Now imagine that we have one independent variable. An appropriate choice of $\mathbf{x}$ and of model (details below) will mean that the predictions made by this model will lie on a straight line. Figure 13.1 shows two regressions. The data are plotted with a scatter plot, and the line gives the prediction of the model for each value on the x axis.

We cannot guarantee that different values of $\mathbf{x}$ produce different values of y. Data just isn't like this (see the crickets example Figure 13.1). This means you can't think of a regression as predicting the true value of y from $\mathbf{x}$ because usually there isn't one. Instead, you should think of a regression as predicting the expected value of y conditioned on $\mathbf{x}$. Some regression models can produce more information about the probability distribution for y conditioned on $\mathbf{x}$. For example, it might be very valuable to get both the mean and variance of the distribution of the likely sale value of a house from independent variables.

It should be clear that none of this will work if there is not some relationship between the training examples and the test examples. If I collect training data on the height and weight of children, I'm unlikely to get good predictions of the weight of adults from their height. We can be more precise with a probabilistic framework. We think of $\mathbf{x}_i$ as IID samples from some (usually unknown) probability distribution

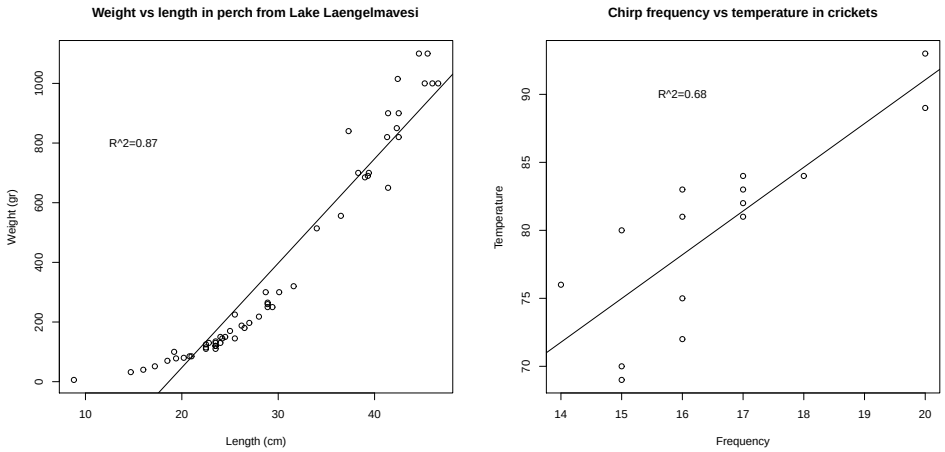

FIGURE 13.1: *On the* **left**, *a regression of weight against length for perch from a Finnish lake (you can find this dataset, and the back story at* http://www.amstat.org/publications/jse/jse_data_archive.htm*; look for "fishcatch" on that page). Notice that the linear regression fits the data fairly well, meaning that you should be able to predict the weight of a perch from its length fairly well. On the* **right**, *a regression of air temperature against chirp frequency for crickets. The data is fairly close to the line, meaning that you should be able to tell the temperature from the pitch of cricket's chirp fairly well. This data is from* http://mste.illinois.edu/patel/amar430/keyprob1.html*. The* R^2 *you see on each figure is a measure of the goodness of fit of the regression (section 13.1.5).*

$P(X)$. Then the test examples should also be IID samples from $P(X)$, or, at least, rather like them – you usually can't check this point with any certainty.

A probabilistic formalism can help be precise about the y_i, too. Assume another random variable Y has joint distribution with X given by $P(Y, X)$. We think of each y_i as a sample from $P(Y|\{X = \mathbf{x}_i\})$. Then our modelling problem would be: given the training data, build a model that takes a test example $\mathbf{x}$ and yields $\mathbb{E}[Y|\{X = \mathbf{x}_i\}]$.

Thinking about the problem this way should make it clear that we're not relying on any exact, physical, or causal relationship between Y and X. It's enough that their joint probability makes useful predictions possible, something we will test by experiment. This means that you can build regressions that work in somewhat surprising circumstances. For example, regressing childrens' reading ability against their foot size can be quite successful. This isn't because having big feet somehow helps you read. It's because on the whole, older children read better, and also have bigger feet. Regression isn't magic. Figure 13.2 shows two regressions where the predictions aren't particularly accurate.

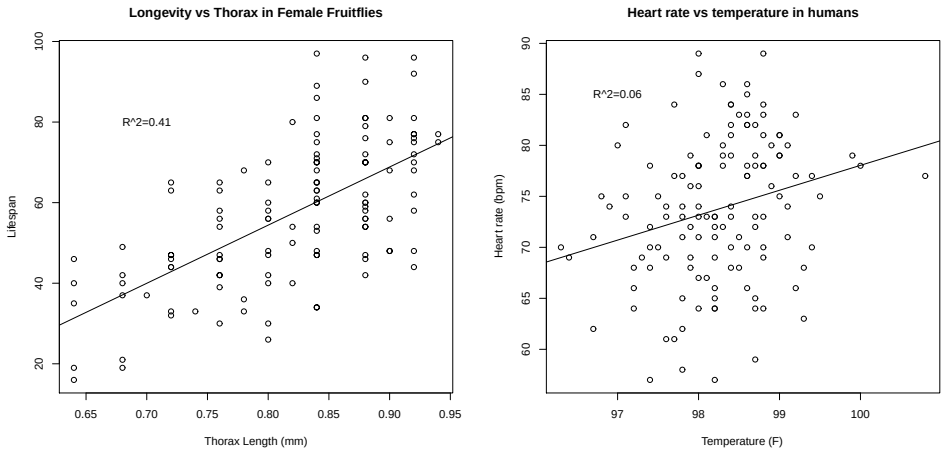

FIGURE 13.2: *Regressions do not necessarily yield good predictions or good model fits. On the* **left**, *a regression of the lifespan of female fruitflies against the length of their torso as adults (apparently, this doesn't change as a fruitfly ages; you can find this dataset, and the back story at http://www.amstat.org/publications/jse/jse_data_archive.htm; look for "fruitfly" on that page). The figure suggests you can make some prediction of how long your fruitfly will last by measuring its torso, but not a particularly accurate one. On the* **right**, *a regression of heart rate against body temperature for adults. You can find the data at http://www.amstat.org/publications/jse/jse_data_archive.htm as well; look for "temperature" on that page. Notice that predicting heart rate from body temperature isn't going to work that well, either.*

13.0.2 Regression to Spot Trends

Regression isn't only used to predict values. Another reason to build a regression model is to compare trends in data. Doing so can make it clear what is really happening. Here is an example from Efron ("Computer-Intensive methods in statistical regression", B. Efron, SIAM Review, 1988). The table in the appendix shows some data from medical devices, which sit in the body and release a hormone. The data shows the amount of hormone currently in a device after it has spent some time in service, and the time the device spent in service. The data describes devices from three production lots (A, B, and C). Each device, from each lot, is supposed to have the same behavior. The important question is: Are the lots the same? The amount of hormone changes over time, so we can't just compare the amounts currently in each device. Instead, we need to determine the relationship between time in service and hormone, and see if this relationship is different between batches. We can do so by regressing hormone against time.

Figure 13.3 shows how a regression can help. In this case, we have modelled the amount of hormone in the device as

$$a \times (\text{time in service}) + b$$

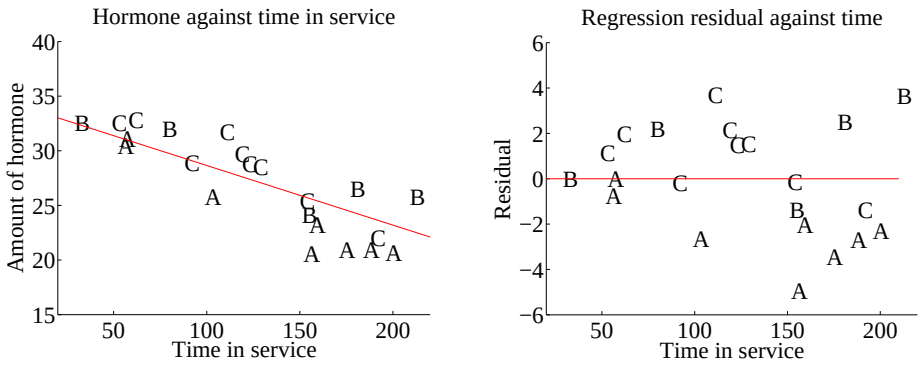

FIGURE 13.3: *On the* **left**, *a scatter plot of hormone against time for devices from tables 13.1 and 13.1. Notice that there is a pretty clear relationship between time and amount of hormone (the longer the device has been in service the less hormone there is). The issue now is to understand that relationship so that we can tell whether lots A, B and C are the same or different. The best fit line to all the data is shown as well, fitted using the methods of section 13.1. On the* **right**, *a scatter plot of residual — the distance between each data point and the best fit line — against time for the devices from tables 13.1 and 13.1. Now you should notice a clear difference; some devices from lots B and C have positive and some negative residuals, but all lot A devices have negative residuals. This means that, when we account for loss of hormone over time, lot A devices still have less hormone in them. This is pretty good evidence that there is a problem with this lot.*

for a, b chosen to get the best fit (much more on this point later!). This means we can plot each data point on a scatter plot, together with the best fitting line. This plot allows us to ask whether any particular batch behaves differently from the overall model in any interesting way.

However, it is hard to evaluate the distances between data points and the best fitting line by eye. A sensible alternative is to subtract the amount of hormone predicted by the model from the amount that was measured. Doing so yields a **residual** — the difference between a measurement and a prediction. We can then plot those residuals (Figure 13.3). In this case, the plot suggests that lot A is special — all devices from this lot contain less hormone than our model predicts.

Definition: 13.1 *Regression*

Regression accepts a feature vector and produces a prediction, which is usually a number, but can sometimes have other forms. You can use these predictions as predictions, or to study trends in data. It is possible, but not usually particularly helpful, to see classification as a form of regression.

13.1 LINEAR REGRESSION AND LEAST SQUARES

Assume we have a dataset consisting of a set of N pairs $(\mathbf{x}_i, y_i)$. We want to use the examples we have — the training examples — to build a model of the dependence between y and $\mathbf{x}$. This model will be used to predict values of y for new values of $\mathbf{x}$, which are usually called test examples. The model needs to have some probabilistic component; we do not expect that y is a function of $\mathbf{x}$, and there is likely some error in evaluating y anyhow.

13.1.1 Linear Regression

We cannot expect that our model makes perfect predictions. Furthermore, y may not be a function of $\mathbf{x}$ — it is quite possible that the same value of $\mathbf{x}$ could lead to different y's. One way that this could occur is that y is a measurement (and so subject to some measurement noise). Another is that there is some randomness in y. For example, we expect that two houses with the same set of features (the $\mathbf{x}$) might still sell for different prices (the y's).

A good, simple model is to assume that the dependent variable (i.e. y) is obtained by evaluating a linear function of the explanatory variables (i.e. $\mathbf{x}$), then adding a zero-mean normal random variable. We can write this model as

$$y = \mathbf{x}^T\beta + \xi$$

where ξ represents random (or at least, unmodelled) effects. In this expression, β is a vector of weights, which we must estimate. We will always assume that ξ has zero mean, so that

$$\mathbb{E}[Y | \{X = \mathbf{x}_i\}] = \mathbf{x}_i\beta.$$

When we use this model to predict a value of y for a particular set of explanatory variables $\mathbf{x}^*$, we cannot predict the value that ξ will take. Our best available prediction is the mean value (which is zero). Notice that if $\mathbf{x} = 0$, the model predicts $y = 0$. This may seem like a problem to you — you might be concerned that we can fit only lines through the origin — but remember that $\mathbf{x}$ contains explanatory variables, and we can choose what appears in $\mathbf{x}$. The two examples show how a sensible choice of $\mathbf{x}$ allows us to fit a line with an arbitrary y-intercept.

Definition: 13.2 *Linear regression*

A linear regression takes the feature vector $\mathbf{x}$ and predicts $\mathbf{x}^T\beta$, for some vector of coefficients β. The coefficients are adjusted, using data, to produce the best predictions.

Example: 13.1 *A linear model fitted to a single explanatory variable*

Assume we fit a linear model to a single explanatory variable. Then the model has the form $y = x\beta + \xi$, where ξ is a zero mean random variable. For any value x^* of the explanatory variable, our best estimate of y is βx^*. In particular, if $x^* = 0$, the model predicts $y = 0$, which is unfortunate. We can draw the model by drawing a line through the origin with slope β in the x, y plane. The y-intercept of this line must be zero.

Example: 13.2 *A linear model with a non-zero y-intercept*

Assume we have a single explanatory variable, which we write u. We can then create a *vector* $\mathbf{x} = [u, 1]^T$ from the explanatory variable. We now fit a linear model to this vector. Then the model has the form $y = \mathbf{x}^T\beta + \xi$, where ξ is a zero mean random variable. For any value $\mathbf{x}^* = [u^*, 1]^T$ of the explanatory variable, our best estimate of y is $(\mathbf{x}^*)^T\beta$, which can be written as $y = \beta_1 u^* + \beta_2$. If $x^* = 0$, the model predicts $y = \beta_2$. We can draw the model by drawing a line through the origin with slope β_1 and y-intercept β_2 in the x, y plane.

13.1.2 Choosing β

We must determine β. We can proceed in two ways. I show both because different people find different lines of reasoning more compelling. Each will get us to the same solution. One is probabilistic, the other isn't. Generally, I'll proceed as if they're interchangeable, although at least in principle they're different.

Probabilistic approach: we could assume that ξ is a zero mean normal random variable with unknown variance. Then $P(y|x, \beta)$ is normal, with mean $\mathbf{x}^T\beta$, and so we can write out the log-likelihood of the data. Write σ^2 for the variance of ξ, which we don't know, but will not worry about right now. We have that

$$\begin{aligned}\log \mathcal{L}(\beta) &= -\sum_i \log P(y_i|\mathbf{x}_i, \beta) \\ &= \frac{1}{2\sigma^2}\sum_i (y_i - \mathbf{x}_i^T\beta)^2 + \text{term not depending on } \beta\end{aligned}$$

Maximizing the log-likelihood of the data is equivalent to minimizing the negative log-likelihood of the data. Furthermore, the term $\frac{1}{2\sigma^2}$ does not affect the location of the minimum, so we must have that β minimizes $\sum_i (y_i - \mathbf{x}_i^T\beta)^2$, or anything proportional to it. It is helpful to minimize an expression that is an average of squared errors, because (hopefully) this doesn't grow much when we add data. We

therefore minimize

$$\left(\frac{1}{N}\right)\left(\sum_i (y_i - \mathbf{x}_i^T\beta)^2\right).$$

Direct approach: notice that, if we have an estimate of β, we have an estimate of the values of the unmodelled effects ξ_i for each example. We just take $\xi_i = y_i - \mathbf{x}_i^T\beta$. It is quite natural to make the unmodelled effects "small". A good measure of size is the mean of the squared values, which means we want to minimize

$$\left(\frac{1}{N}\right)\left(\sum_i (y_i - \mathbf{x}_i^T\beta)^2\right).$$

13.1.3 Solving the Least Squares Problem

We can write all this more conveniently using vectors and matrices. Write $\mathbf{y}$ for the vector

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix}$$

and $\mathcal{X}$ for the matrix

$$\begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \dots \mathbf{x}_n^T \end{pmatrix}.$$

Then we want to minimize

$$\left(\frac{1}{N}\right)(\mathbf{y} - \mathcal{X}\beta)^T(\mathbf{y} - \mathcal{X}\beta)$$

which means that we must have

$$\mathcal{X}^T\mathcal{X}\beta - \mathcal{X}^T\mathbf{y} = 0.$$

For reasonable choices of features, we could expect that $\mathcal{X}^T\mathcal{X}$ — which should strike you as being a lot like a covariance matrix — has full rank. If it does, which is the usual case, this equation is easy to solve. If it does not, there is more to do, which we will do in section 13.2.4.

Remember this: *The vector of coefficients β for a linear regression is usually estimated using a least-squares procedure.*

13.1.4 Residuals

Assume we have produced a regression by solving

$$\mathcal{X}^T\mathcal{X}\hat{\beta} - \mathcal{X}^T\mathbf{y} = 0$$

for the value of $\hat{\beta}$. I write $\hat{\beta}$ because this is an *estimate*; we likely don't have the true value of the β that generated the data (the model might be wrong; etc.). We cannot expect that $\mathcal{X}\hat{\beta}$ is the same as $\mathbf{y}$. Instead, there is likely to be some error. The **residual** is the vector

$$\mathbf{e} = \mathbf{y} - \mathcal{X}\hat{\beta}$$

which gives the difference between the true value and the model's prediction at each point. Each component of the residual is an estimate of the unmodelled effects for that data point. The **mean square error** is

$$m = \frac{\mathbf{e}^T\mathbf{e}}{N}$$

and this gives the average of the squared error of prediction on the training examples.

Notice that the mean squared error is not a great measure of how good the regression is. This is because the value depends on the units in which the dependent variable is measured. So, for example, if you measure y in meters you will get a different mean squared error than if you measure y in kilometers *for the same dataset*. This is a serious nuisance, because it means that the value of the mean squared error cannot tell you how good a regression is. There is an alternative measure of the accuracy of a regression which does not depends on the units of y.

13.1.5 R-squared

Unless the dependent variable is a constant (which would make prediction easy), it has some variance. If our model is of any use, it should explain some aspects of the value of the dependent variable. This means that the variance of the residual should be smaller than the variance of the dependent variable. If the model made perfect predictions, then the variance of the residual should be zero.

We can formalize all this in a relatively straightforward way. We will ensure that $\mathcal{X}$ always has a column of ones in it, so that the regression can have a non-zero y-intercept. We now fit a model

$$\mathbf{y} = \mathcal{X}\beta + \mathbf{e}$$

(where $\mathbf{e}$ is the vector of residual values) by choosing β such that $\mathbf{e}^T\mathbf{e}$ is minimized. Then we get some useful technical results.

Useful Facts: 13.1 *Regression*

We write $\mathbf{y} = \mathcal{X}\hat{\beta} + \mathbf{e}$, where $\mathbf{e}$ is the residual. For a vector $\mathbf{v}$ of N components, we write $\overline{\mathbf{v}} = (1/N)\mathbf{1}^T\mathbf{v}$. Assume $\mathcal{X}$ has a column of ones, and $\hat{\beta}$ is chosen to minimize $\mathbf{e}^T\mathbf{e}$. Then we have

1. $\mathbf{e}^T\mathcal{X} = \mathbf{0}$, i.e. that $\mathbf{e}$ is orthogonal to any column of $\mathcal{X}$. If $\mathbf{e}$ is not orthogonal to some column of $\mathbf{e}$, we can increase or decrease the $\hat{\beta}$ term corresponding to that column to make the error smaller. Another way to see this is to notice that $\hat{\beta}$ is chosen to minimize $\frac{1}{N}\mathbf{e}^T\mathbf{e}$, which is $\frac{1}{N}(\mathbf{y} - \mathcal{X}\hat{\beta})^T(\mathbf{y} - \mathcal{X}\hat{\beta})$. Now because this is a minimum, the gradient with respect to $\hat{\beta}$ is zero, so $(\mathbf{y} - \mathcal{X}\hat{\beta})^T(-\mathcal{X}) = -\mathbf{e}^T\mathcal{X} = 0$.
2. $\mathbf{e}^T\mathbf{1} = 0$ (recall that $\mathcal{X}$ has a column of all ones, and apply the previous result).
3. $\mathbf{e}^T\mathcal{X}\hat{\beta} = 0$ (first result means that this is true).
4. $\mathbf{1}^T(\mathbf{y} - \mathcal{X}\hat{\beta}) = 0$ (same as previous result).
5. $\overline{\mathbf{y}} = \overline{\mathcal{X}\hat{\beta}}$ (same as previous result).

Now $\mathbf{y}$ is a one dimensional dataset arranged into a vector, so we can compute $\mathsf{mean}\,(\{y\})$ and $\mathsf{var}[y]$. Similarly, $\mathcal{X}\hat{\beta}$ is a one dimensional dataset arranged into a vector (its elements are $\mathbf{x}_i^T\hat{\beta}$), as is $\mathbf{e}$, so we know the meaning of mean and variance for each. We have a particularly important result:

$$\mathsf{var}[y] = \mathsf{var}\left[\mathcal{X}\hat{\beta}\right] + \mathsf{var}[e].$$

This is quite easy to show, with a little more notation. Write $\overline{\mathbf{y}} = (1/N)(\mathbf{1}^T\mathbf{y})\mathbf{1}$ for the vector whose entries are all $\mathsf{mean}\,(\{y\})$; similarly for $\overline{\mathbf{e}}$ and for $\overline{\mathcal{X}\hat{\beta}}$. We have

$$\mathsf{var}[y] = (1/N)(\mathbf{y} - \overline{\mathbf{y}})^T(\mathbf{y} - \overline{\mathbf{y}})$$

and so on for $\mathsf{var}[e_i]$, etc. Notice from the facts that $\overline{\mathbf{y}} = \overline{\mathcal{X}\hat{\beta}}$. Now

$$\begin{aligned}
\mathsf{var}[y] &= (1/N)\left(\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right] + [\mathbf{e} - \overline{\mathbf{e}}]\right)^T\left(\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right] + [\mathbf{e} - \overline{\mathbf{e}}]\right) \\
&= (1/N)\left(\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right]^T\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right] + 2\,[\mathbf{e} - \overline{\mathbf{e}}]^T\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right] + [\mathbf{e} - \overline{\mathbf{e}}]^T[\mathbf{e} - \overline{\mathbf{e}}]\right) \\
&= (1/N)\left(\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right]^T\left[\mathcal{X}\hat{\beta} - \overline{\mathcal{X}\hat{\beta}}\right] + [\mathbf{e} - \overline{\mathbf{e}}]^T[\mathbf{e} - \overline{\mathbf{e}}]\right) \\
&\quad \text{because } \overline{\mathbf{e}} = 0 \text{ and } \mathbf{e}^T\mathcal{X}\hat{\beta} = 0 \text{ and } \mathbf{e}^T\mathbf{1} = 0 \\
&= \mathsf{var}\left[\mathcal{X}\hat{\beta}\right] + \mathsf{var}[e].
\end{aligned}$$

This is extremely important, because us allows us to think about a regression as explaining variance in $\mathbf{y}$. As we are better at explaining $\mathbf{y}$, $\mathsf{var}[e]$ goes down. In turn, a natural measure of the goodness of a regression is what percentage of the variance of $\mathbf{y}$ it explains. This is known as R^2 (the r-squared measure). We have

$$R^2 = \frac{\mathsf{var}\left[\mathbf{x}_i^T \hat{\beta}\right]}{\mathsf{var}[y_i]}$$

which gives some sense of how well the regression explains the training data. Notice that the value of R^2 is not affected by the units of $\mathbf{y}$ (exercises)

Good predictions result in high values of R^2, and a perfect model will have $R^2 = 1$ (which doesn't usually happen). For example, the regression of figure 13.3 has an R^2 value of 0.87. Figures 13.1 and 13.2 show the R^2 values for the regressions plotted there; notice how better models yield larger values of R^2. Notice that if you look at the summary that R provides for a linear regression, it will offer you *two* estimates of the value for R^2. These estimates are obtained in ways that try to account for (a) the amount of data in the regression, and (b) the number of variables in the regression. For our purposes, the differences between these numbers and the R^2 I defined are not significant. For the figures, I computed R^2 as I described in the text above, but if you substitute one of R's numbers nothing terrible will happen.

Remember this: *The quality of predictions made by a regression can be evaluated by looking at the fraction of the variance in the dependent variable that is explained by the regression. This number is called R^2, and lies between zero and one; regressions with larger values make better predictions.*

Procedure: 13.1 *Linear Regression using Least Squares*

We have a dataset containing N pairs $(\mathbf{x}_i, y_i)$. Each x_i is a d-dimensional explanatory vector, and each y_i is a single dependent variable. We assume that each data point conforms to the model

$$y_i = \mathbf{x}_i^T \beta + \xi_i$$

where ξ_i represents unmodelled effects. We assume that ξ_i are samples of a random variable with 0 mean and unknown variance. Sometimes, we assume the random variable is normal. Write

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \dots \\ y_n \end{pmatrix} \text{ and } \mathcal{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \dots \\ \mathbf{x}_n^T \end{pmatrix}.$$

We estimate $\hat{\beta}$ (the value of β) by solving the linear system

$$\mathcal{X}^T \mathcal{X} \hat{\beta} - \mathcal{X}^T \mathbf{y} = 0.$$

For a data point $\mathbf{x}$, our model predicts $\mathbf{x}^T \hat{\beta}$. The residuals are

$$\mathbf{e} = \mathbf{y} - \mathcal{X}\hat{\beta}.$$

We have that $\mathbf{e}^T \mathbf{1} = 0$. The mean square error is given by

$$m = \frac{\mathbf{e}^T \mathbf{e}}{N}.$$

The R^2 is given by

$$\frac{\mathsf{var}\left(\left\{\mathbf{x}_i^T \hat{\beta}\right\}\right)}{\mathsf{var}\left(\{\mathbf{y}\}\right)}.$$

Values of R^2 range from 0 to 1; a larger value means the regression is better at explaining the data.

13.2 PRODUCING GOOD LINEAR REGRESSIONS

Linear regression is useful, but it isn't magic. Some regressions make poor predictions (recall the regressions of figure 13.2). As another example, regressing the first digit of your telephone number against the length of your foot won't work.

We have some straightforward tests to tell whether a regression is working. You can **look at a plot** for a dataset with one explanatory variable and one dependent variable. You plot the data on a scatter plot, then plot the model as a line on that scatterplot. Just looking at the picture can be informative (compare

Figure 13.1 and Figure 13.2).

You can check if the regression **predicts a constant**. This is usually a bad sign. You can check this by looking at the predictions for each of the training data items. If the variance of these predictions is small compared to the variance of the independent variable, the regression isn't working well. If you have only one explanatory variable, then you can plot the regression line. If the line is horizontal, or close, then the value of the explanatory variable makes very little contribution to the prediction. This suggests that there is no particular relationship between the explanatory variable and the independent variable.

You can also check, by eye, if **the residual isn't random**. If $y - \mathbf{x}^T\beta$ is a zero mean normal random variable, then the value of the residual vector should not depend on the corresponding y-value. Similarly, if $y - \mathbf{x}^T\beta$ is just a zero mean collection of unmodelled effects, we want the value of the residual vector to not depend on the corresponding y-value either. If it does, that means there is some phenomenon we are not modelling. Looking at a scatter plot of $\mathbf{e}$ against $\mathbf{y}$ will often reveal trouble in a regression (Figure 13.7). In the case of Figure 13.7, the trouble is caused by a few data points that are very different from the others severely affecting the regression. We will discuss such points in more detail below. Once they have been removed, the regression improves markedly (Figure 13.8).

Remember this: *Linear regressions can make bad predictions. You can check for trouble by: evaluating R^2; looking at a plot; looking to see if the regression makes a constant prediction; or checking whether the residual is random. Other strategies exist, but are beyond the scope of this book.*

13.2.1 Transforming Variables

Sometimes the data isn't in a form that leads to a good linear regression. In this case, transforming explanatory variables, the dependent variable, or both can lead to big improvements. Figure 13.4 shows one example, based on the idea of word frequencies. Some words are used very often in text; most are used seldom. The dataset for this figure consists of counts of the number of time a word occurred for the 100 most common words in Shakespeare's printed works. It was originally collected from a concordance, and has been used to attack a variety of interesting questions, including an attempt to assess how many words Shakespeare knew. This is hard, because he likely knew many words that he didn't use in his works, so one can't just count. If you look at the plot of Figure 13.4, you can see that a linear regression of count (the number of times a word is used) against rank (how common a word is, 1-100) is not really useful. The most common words are used very often, and the number of times a word is used falls off very sharply as one looks at less common words. You can see this effect in the scatter plot of residual against dependent variable in Figure 13.4 — the residual depends rather strongly

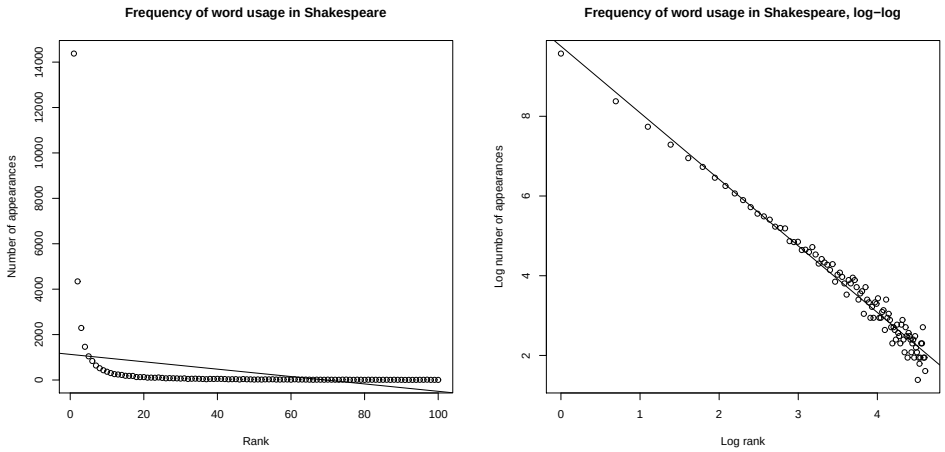

FIGURE 13.4: *On the* **left**, *word count plotted against rank for the 100 most common words in Shakespeare, using a dataset that comes with R (called "bard", and quite likely originating in an unpublished report by J. Gani and I. Saunders). I show a regression line too. This is a poor fit by eye, and the* R^2 *is poor, too (*$R^2 = 0.1$*). On the* **right**, *log word count plotted against log rank for the 100 most common words in Shakespeare, using a dataset that comes with R (called "bard", and quite likely originating in an unpublished report by J. Gani and I. Saunders). The regression line is very close to the data.*

on the dependent variable. This is an extreme example that illustrates how poor linear regressions can be.

However, if we regress log-count against log-rank, we get a very good fit indeed. This suggests that Shakespeare's word usage (at least for the 100 most common words) is consistent with **Zipf's law**. This gives the relation between frequency f and rank r for a word as

$$f \propto \frac{1}{r}^s$$

where s is a constant characterizing the distribution. Our linear regression suggests that s is approximately 1.67 for this data.

In some cases, the natural logic of the problem will suggest variable transformations that improve regression performance. For example, one could argue that humans have approximately the same density, and so that weight should scale as the cube of height; in turn, this suggests that one regress weight against the cube root of height. Figure 13.5 shows the result of this transformation on the fish data, where it appears to help a lot. Generally, shorter people tend not to be scaled versions of taller people, so the cube root might be too aggressive. The body mass index (BMI: a controversial but not completely pointless measure of the relationship between weight and height) uses the square root.

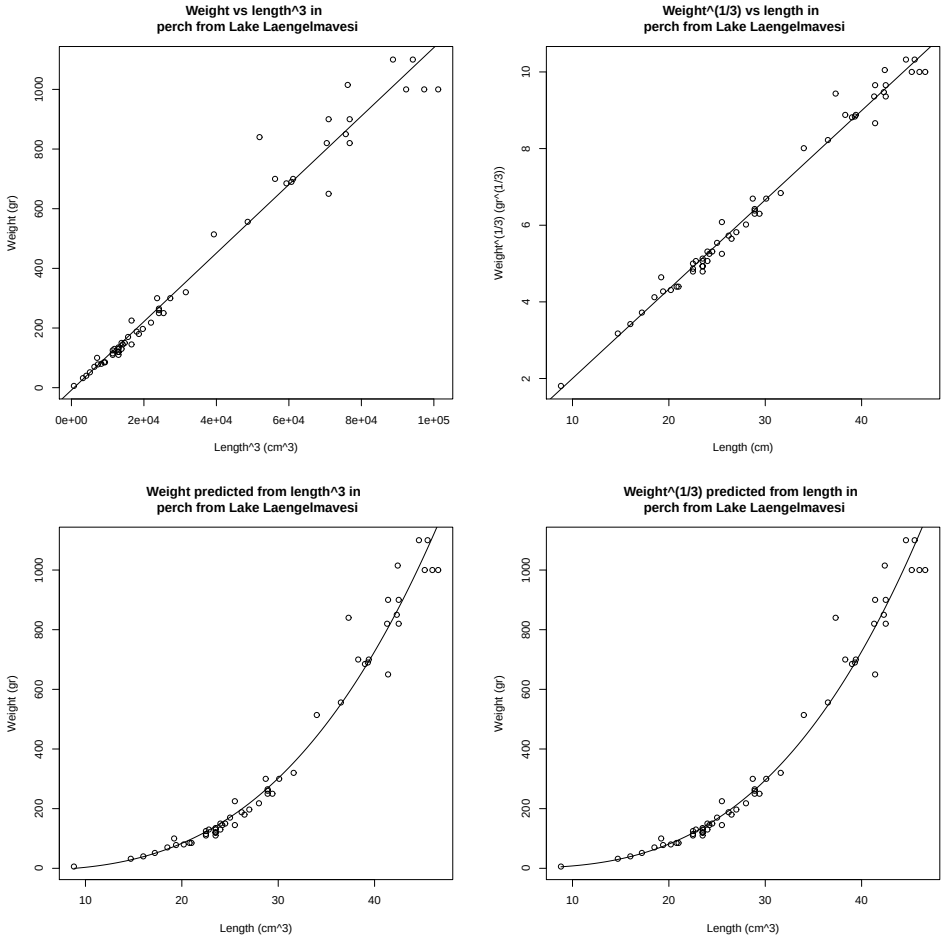

FIGURE 13.5: *Two variable transformations on the perch dataset. On the* **top left**, *weight predicted from length cubed; on the* **top right**, *cube root of weight predicted from length. On the* **bottom** *corresponding plots transformed to weight-length coordinates (you need to look very closely to see the differences). The non-linear transformation helps significantly.*

Remember this: *The performance of a regression can be improved by transforming variables. Transformations can follow from looking at plots, or thinking about the logic of the problem*

13.2.2 Problem Data Points have Significant Impact

Outlying data points can significantly weaken the usefulness of a regression. For some regression problems, we can identify data points that might be a problem, and then resolve how to deal with them. One possibility is that they are true outliers — someone recorded a data item wrong, or they represent an effect that just doesn't occur all that often. Another is that they are important data, and our linear model may not be good enough. If the data points really are outliers, we can drop them from the data set. If they aren't, we may be able to improve the regression by transforming features or by finding a new explanatory variable.

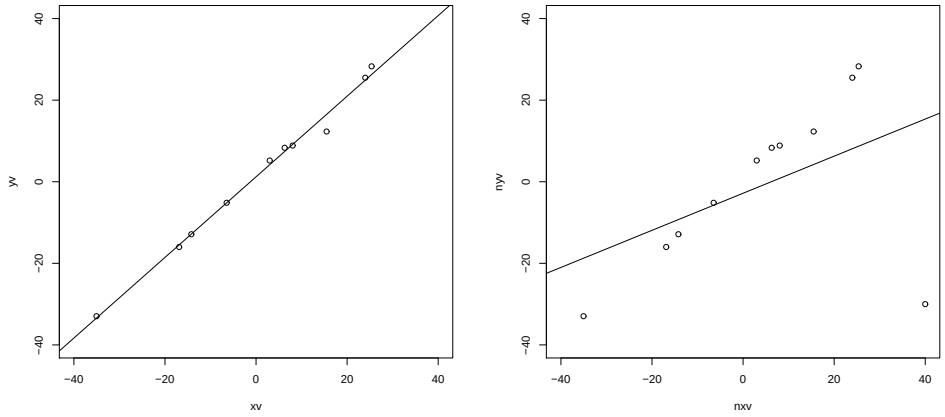

FIGURE 13.6: *On the* **left**, *a synthetic dataset with one independent and one explanatory variable, with the regression line plotted. Notice the line is close to the data points, and its predictions seem likely to be reliable. On the* **right**, *the result of adding a single outlying datapoint to that dataset. The regression line has changed significantly, because the regression line tries to minimize the sum of squared vertical distances between the data points and the line. Because the outlying datapoint is far from the line, the squared vertical distance to this point is enormous. The line has moved to reduce this distance, at the cost of making the other points further from the line.*

When we construct a regression, we are solving for the β that minimizes $\sum_i (y_i - \mathbf{x}_i^T \beta)^2$, equivalently for the β that produces the smallest value of $\sum_i e_i^2$. This means that residuals with large value can have a very strong influence on the outcome — we are squaring that large value, resulting in an enormous value. Generally, many residuals of medium size will have a smaller cost than one large residual and the rest tiny. As figure 13.6 illustrates, this means that a data point that lies far from the others can swing the regression line significantly.

This creates a problem, because data points that are very different from most others (sometimes called **outliers**) can also have the highest influence on the outcome of the regression. Figure 13.8 shows this effect for a simple case. When we have only one explanatory variable, there's an easy method to spot problem data

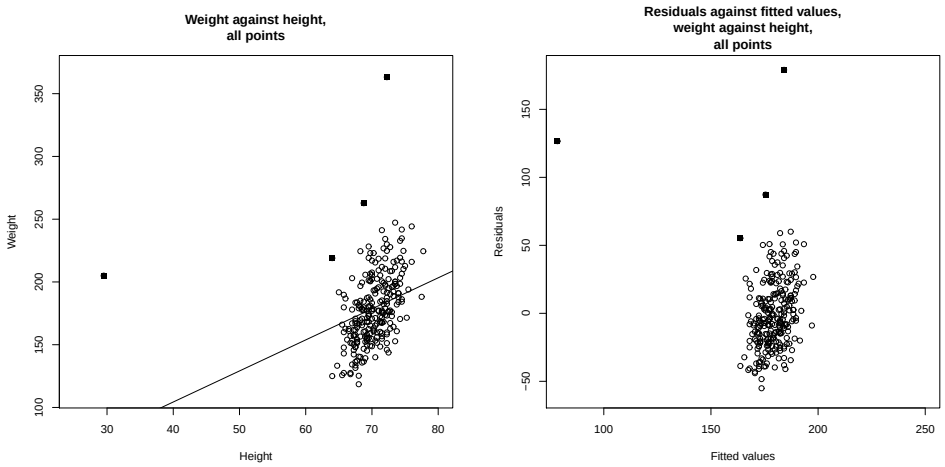

FIGURE 13.7: *On the* **left**, *weight regressed against height for the bodyfat dataset. The line doesn't describe the data particularly well, because it has been strongly affected by a few data points (filled-in markers). On the* **right**, *a scatter plot of the residual against the value predicted by the regression. This doesn't look like noise, which is a sign of trouble.*

points. We produce a scatter plot and a regression line, and the difficulty is usually obvious. In particularly tricky cases, printing the plot and using a see-through ruler to draw a line by eye can help (if you use an opaque ruler, you may not see some errors).

These data points can come from many sources. They may simply be errors. Failures of equipment, transcription errors, someone guessing a value to replace lost data, and so on are some methods that might produce outliers. Another possibility is your understanding of the problem is wrong. If there are some rare effects that are very different than the most common case, you might see outliers. Major scientific discoveries have resulted from investigators taking outliers seriously, and trying to find out what caused them (though you shouldn't see a Nobel prize lurking behind every outlier).

What to do about outliers is even more fraught. The simplest strategy is to find them, then remove them from the data. For low dimensional models, you can do this by plotting data and predictions, then looking for problems. There are other methods, but they are too complicated for us. You should be aware that this strategy can get dangerous fairly quickly, whether you use a simple or a sophisticated method. First, you might find that each time you remove a few problematic data points, some more data points look strange to you. This process is unlikely to end well. Second, you should be aware that throwing out outliers can *increase* your future prediction error, particularly if they're caused by real effects. An alternative strategy is to build methods that can either discount or model the effects of outliers.

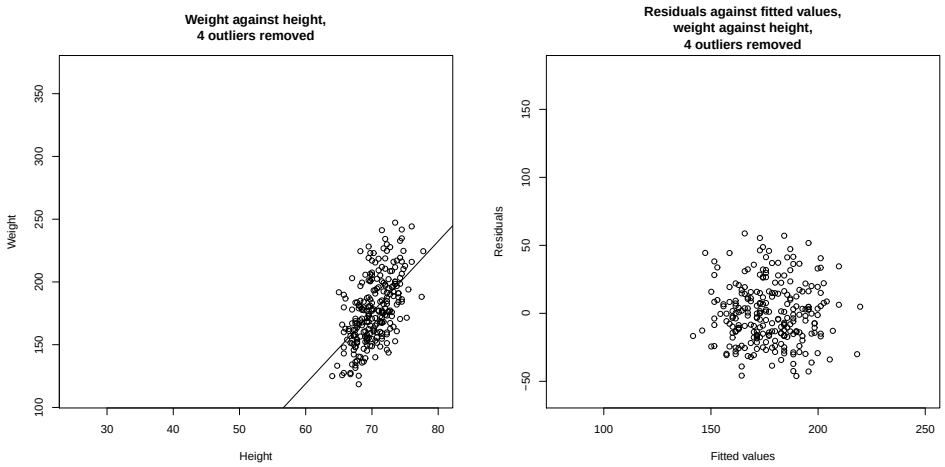

FIGURE 13.8: *On the* **left**, *weight regressed against height for the bodyfat dataset. I have now removed the four suspicious looking data points, identified in Figure 13.7 with filled-in markers; these seemed the most likely to be outliers. On the* **right**, *a scatter plot of the residual against the value predicted by the regression. Notice that the residual looks like noise. The residual seems to be uncorrelated to the predicted value; the mean of the residual seems to be zero; and the variance of the residual doesn't depend on the predicted value. All these are good signs, consistent with our model, and suggest the regression will yield good predictions.*

Remember this: *Outliers can affect linear regressions significantly. Usually, if you can plot the regression, you can look for outliers by eyeballing the plot. Other methods exist, but are beyond the scope of this text.*

13.2.3 Functions of One Explanatory Variable

Imagine we have only one measurement to form explanatory variables. For example, in the perch data of Figure 13.1, we have only the length of the fish. If we evaluate functions of that measurement, and insert them into the vector of explanatory variables, the resulting regression is still easy to plot. It may also offer better predictions. The fitted line of Figure 13.1 looks quite good, but the data points look as though they might be willing to follow a curve. We can get a curve quite easily. Our current model gives the weight as a linear function of the length with a noise term (which we wrote $y_i = \beta_1 x_i + \beta_0 + \xi_i$). But we could expand this model to incorporate other functions of the length. In fact, it's quite suprising that the weight of a fish should be predicted by its length. If the fish doubled in each direction, say, its weight should go up by a factor of eight. The success of our regression suggests that fish do not just scale in each direction as they grow. But

we might try the model $y_i = \beta_2 x_i^2 + \beta_1 x_i + \beta_0 + \xi_i$. This is easy to do. The i'th row of the matrix $\mathcal{X}$ currently looks like $[x_i, 1]$. We build a new matrix $\mathcal{X}^{(b)}$, where the i'th row is $[x_i^2, x_i, 1]$, and proceed as before. This gets us a new model. The nice thing about this model is that it is easy to plot – our predicted weight is still a function of the length, it's just not a linear function of the length. Several such models are plotted in Figure 13.9.

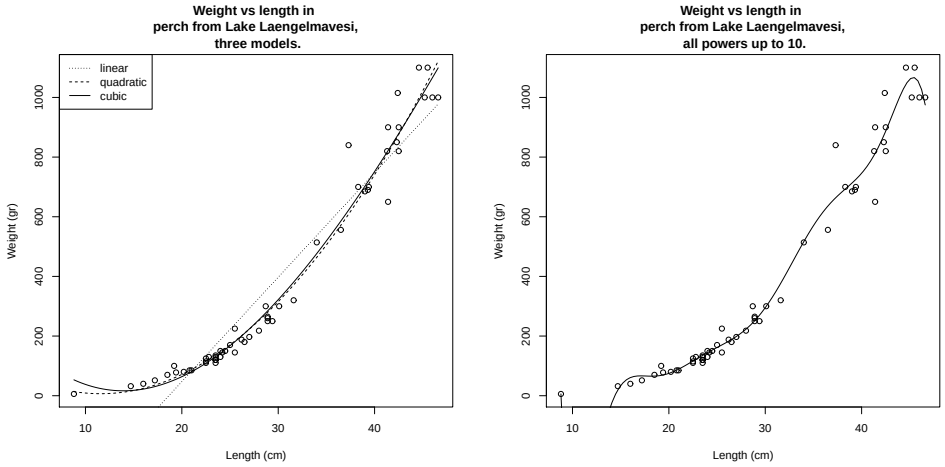

FIGURE 13.9: *On the* **left**, *several different models predicting fish weight from length. The line uses the explanatory variables* 1 *and* x_i*; and the curves use other monomials in* x_i *as well, as shown by the legend. This allows the models to predict curves that lie closer to the data. It is important to understand that, while you can make a curve go closer to the data by inserting monomials, that doesn't mean you necessarily have a better model. On the* **right**, *I have used monomials up to* x_i^{10}*. This curve lies very much closer to the data points than any on the other side, at the cost of some very odd looking wiggles inbetween data points (look at small lengths; the model goes quite strongly negative there, but I can't bring myself to change the axes and show predictions that are obvious nonsense). I can't think of any reason that these structures would come from true properties of fish, and it would be hard to trust predictions from this model.*

You should notice that it can be quite easy to add a lot of functions like this (in the case of the fish, I tried x_i^3 as well). However, it's hard to decide whether the regression has actually gotten better. The least-squares error *on the training data* will *never* go up when you add new explanatory variables, so the R^2 will *never* get worse. This is easy to see, because you could always use a coefficient of zero with the new variables and get back the previous regression. However, the models that you choose are likely to produce worse and worse predictions as you add explanatory variables. Knowing when to stop can be tough, though it's sometimes obvious that the model is untrustworthy (Figure 13.9).

Remember this: *If you have only one measurement, you can construct a high dimensional* $\mathbf{x}$ *by using functions of that measurement. This produces a regression that has many explanatory variables, but is still easy to plot. Knowing when to stop is hard. An understanding of the problem is helpful.*

13.2.4 Regularizing Linear Regressions

When we have many explanatory variables, some might be significantly correlated. This means that we can predict, quite accurately, the value of one explanatory variable using the values of the other variables. This means there must be a vector $\mathbf{w}$ so that $\mathcal{X}\mathbf{w}$ is small (exercises). In turn, that $\mathbf{w}^T\mathcal{X}^T\mathcal{X}\mathbf{w}$ must be small, so that $\mathcal{X}^T\mathcal{X}$ has some small eigenvalues. These small eigenvalues lead to bad predictions, as follows. The vector $\mathbf{w}$ has the property that $\mathcal{X}^T\mathcal{X}\mathbf{w}$ is small. This means that $\mathcal{X}^T\mathcal{X}(\hat{\beta} + \mathbf{w})$ is not much different from $\mathcal{X}^T\mathcal{X}\hat{\beta}$ (equivalently, the matrix can turn large vectors into small ones). All this means that $(\mathcal{X}^T\mathcal{X})^{-1}$ will turn some small vectors into big ones. A small change in $\mathcal{X}^T\mathbf{Y}$ can lead to a large change in the estimate of $\hat{\beta}$.

This is a problem, because we can expect that different samples from the same data will have somewhat different values of $\mathcal{X}^T\mathbf{Y}$. For example, imagine the person recording fish measurements in Lake Laengelmavesi recorded a different set of fish; we expect changes in $\mathcal{X}$ and $\mathbf{Y}$. But, if $\mathcal{X}^T\mathcal{X}$ has small eigenvalues, these changes could produce large changes in our model.

The problem is relatively easy to control. When there are small eigenvalues in $\mathcal{X}^T\mathcal{X}$, we expect that $\hat{\beta}$ will be large (because we can add components in the direction of $\mathbf{w}$ without changing all that much), and the largest components in $\hat{\beta}$ might be very inaccurately estimated. If we are trying to predict new y values, we expect that large components in $\hat{\beta}$ turn into large errors in prediction (exercises).

An important and useful way to suppress these errors is to try to find a $\hat{\beta}$ that isn't large, and also gives a low error. We can do this by regularizing, using the same trick we saw in the case of classification. Instead of choosing the value of β that minimizes

$$\left(\frac{1}{N}\right)(\mathbf{y} - \mathcal{X}\beta)^T(\mathbf{y} - \mathcal{X}\beta)$$

we minimize

$$\begin{array}{rcl}\left(\frac{1}{N}\right)(\mathbf{y} - \mathcal{X}\beta)^T(\mathbf{y} - \mathcal{X}\beta) & + & \lambda\beta^T\beta \\ \text{Error} & + & \text{Regularizer}\end{array}$$

Here $\lambda > 0$ is a constant (the **regularization weight**, though it's pretty widely known as λ) that weights the two requirements (small error; small $\hat{\beta}$) relative to one another. Notice also that dividing the total error by the number of data points means that our choice of λ shouldn't be affected by changes in the size of the data set.

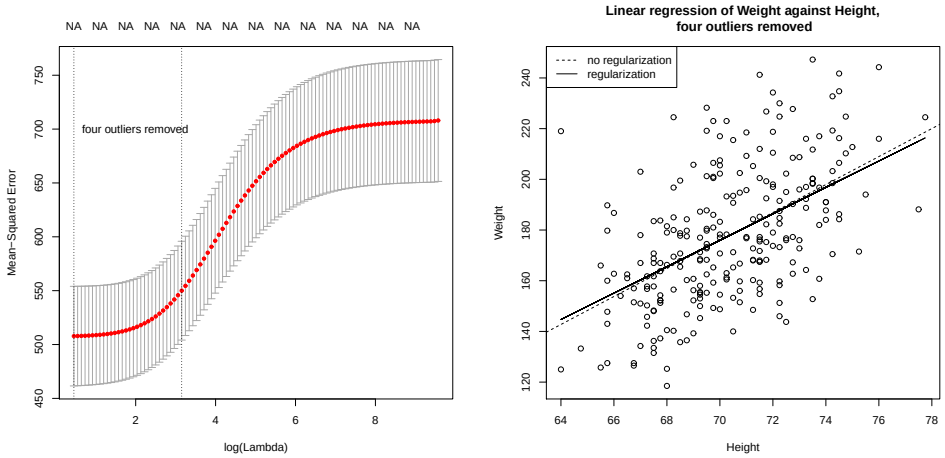

FIGURE 13.10: *On the* **left**, *cross-validated error estimated for different choices of regularization constant for a linear regression of weight against height for the bodyfat dataset, with four outliers removed. The horizontal axis is log regression constant; the vertical is cross-validated error. The mean of the error is shown as a spot, with vertical error bars. The vertical lines show a range of reasonable choices of regularization constant (***left** *yields the lowest observed error,* **right** *the error whose mean is within one standard error of the minimum). On the* **right**, *two regression lines on a scatter plot of this dataset; one is the line computed without regularization, the other is obtained using the regularization parameter that yields the lowest observed error. In this case, the regularizer doesn't change the line much, but may produce improved values on new data (notice how the cross-validated error is fairly flat with low values of the regularization constant).*

Regularization helps deal with the small eigenvalue, because to solve for β we must solve the equation

$$\left[\left(\frac{1}{N}\right)\mathcal{X}^T\mathcal{X} + \lambda\mathcal{I}\right]\hat{\beta} = \left(\frac{1}{N}\right)\mathcal{X}^T\mathbf{y}$$

(obtained by differentiating with respect to β and setting to zero) and the smallest eigenvalue of the matrix $\left(\left(\frac{1}{N}\right)\left(\mathcal{X}^T\mathcal{X} + \lambda\mathcal{I}\right)\right.$ will be at least λ (exercises). Penalizing a regression with the size of β in this way is sometimes known as **ridge regression**. The value of λ that is most helpful depends on the dataset. Typically, one sets up a range of values, then searches, using cross-validation to estimate the error.

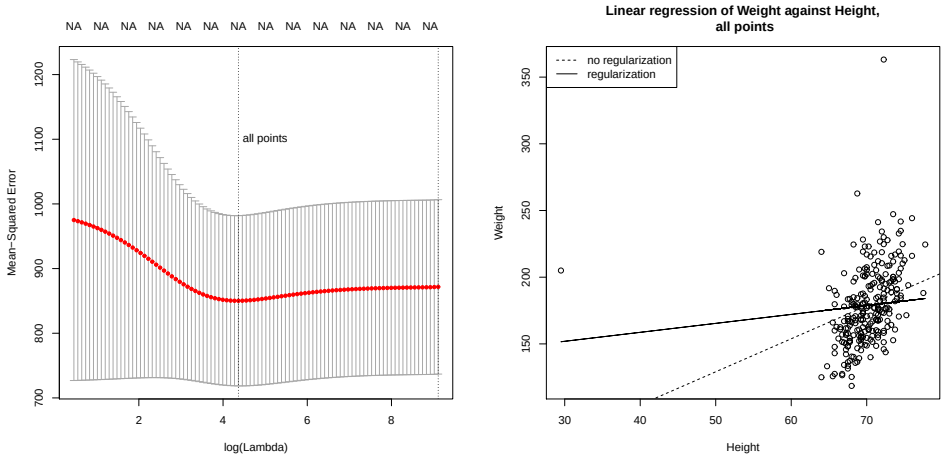

FIGURE 13.11: *Regularization doesn't make outliers go away. On the* **left**, *cross-validated error estimated for different choices of regularization constant for a linear regression of weight against height for the bodyfat dataset, with all points. The horizontal axis is log regression constant; the vertical is cross-validated error. The mean of the error is shown as a spot, with vertical error bars. The vertical lines show a range of reasonable choices of regularization constant (***left** *yields the lowest observed error,* **right** *the error whose mean is within one standard error of the minimum). On the* **right**, *two regression lines on a scatter plot of this dataset; one is the line computed without regularization, the other is obtained using the regularization parameter that yields the lowest observed error. In this case, the regularizer doesn't change the line much, but may produce improved values on new data (notice how the cross-validated error is fairly flat with low values of the regularization constant).*

Worked example 13.1 *Predicting the weight of a fish with regularized linear regression*

We have already seen how to predict the weight of a fish using different powers of its length (sections 13.2.1 and 13.2.3; figure 13.9). Section 13.2.3 showed that using too many powers would likely lead to poor predictions on test data. Show that regularization can be used to control this problem.

Solution: The main point of this example is how useful good statistical software can be. The package I use for regressions, `glmnet`, will choose a good range of regularization weights (λ's) and compute estimates of the mean and standard deviation of the squared cross-validated error for various values in that range. It then prepares a nice plot of this information, which makes the impact of the regularization clear. I've shown such a plot in figure 13.12. In this problem, quite a large value of the regularization constant produces the best result. I've also show a plot of the predictions made, with the coefficients of each power of length used in the regression for the best value of the regularization constant. You should notice that the coefficient of length and its square are fairly high, there's a small value of the coefficient for the cube of length, and for higher powers the coefficients are pretty tiny. If you're careful, you'll check that the coefficients are small compared to the scale of the numbers (because the 10'th power of 20, say, is big). The curve has no wiggles in it, because these coefficients mean that high powers make almost no contribution to its shape.

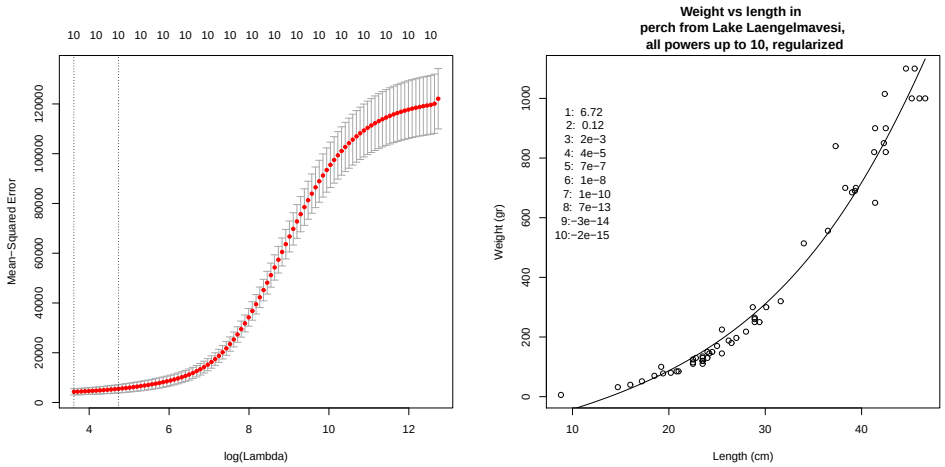

FIGURE 13.12: *Regularization can be a significant help when there are many predictors. On the* **left**, *the glmnet plot of cross-validated prediction error against log regularization coefficient for the perch data of Figure 13.9. The set of independent variables includes all powers of length up to 10 (as in the wiggly graph on the right of Figure 13.9). Notice that the regularization coefficient that yields the smallest error is quite large (the horizontal axis is on a logarithmic scale). On the* **right**, *the curve of predicted values. The cross-validated error chooses a regularization constant that discourages. wiggles; inspecting the coefficients, shown in the inset, shows that high powers of length are firmly suppressed.*

We choose λ in the same way we used for classification; split the training set into a training piece and a validation piece, train for different values of λ, and test the resulting regressions on the validation piece. The error is a random variable, random because of the random split. It is a fair model of the error that would occur on a randomly chosen test example (assuming that the training set is "like" the test set, in a way that I do not wish to make precise yet). We could use multiple splits, and average over the splits. Doing so yields both an average error for a value of λ and an estimate of the standard deviation of error.

Statistical software will do all the work for you. I used the `glmnet` package in R; this package is available in Matlab, too. There are likely other such packages. Figure 13.10 shows an example, for weight regressed against height. Notice the regularization doesn't change the model (plotted in the figure) all that much. For each value of λ (horizontal axis), the method has computed the mean error and standard deviation of error using cross-validation splits, and displays these with error bars. Notice that $\lambda = 0$ yields poorer predictions than a larger value; large $\hat{\beta}$ really are unreliable. Notice that now there is now no λ that yields the smallest validation error, because the value of error depends on the random splits used in cross-validation. A reasonable choice of λ lies between the one that yields the smallest error encountered (one vertical line in the plot) and the largest value whose mean error is within one standard deviation of the minimum (the other vertical line in the plot).

All this is quite similar to regularizing a classification problem. We started with a cost function that evaluated the errors caused by a choice of β, then added a term that penalized β for being "large". This term is the squared length of β, as a vector. It is sometimes known as the L_2 **norm** of the vector.

Remember this: *The performance of a regression can be improved by regularizing, particularly if some explanatory variables are correlated. The procedure is similar to that used for classification.*

13.3 EXPLOITING YOUR NEIGHBORS FOR REGRESSION

Nearest neighbors can clearly predict a number for a query example — you find the closest training example, and report its number. This would be one way to use nearest neighbors for regression, but it isn't terribly effective. One important difficulty is that the regression prediction is piecewise constant (Figure 13.13). If there is an immense amount of data, this may not present major problems, because the steps in the prediction will be small and close together. But it's not generally an effective use of data.

A more effective strategy is to find several nearby training examples, and use them to produce an estimate. This approach can produce very good regression estimates, because every prediction is made by training examples that are near to the query example. However, producing a regression estimate is expensive, because for every query one must find the nearby training examples.

Write $\mathbf{x}$ for the query point, and assume that we have already collected the N nearest neighbors, which we write $\mathbf{x}_i$. Write y_i for the value of the dependent variable for the i'th of these points. Notice that some of these neighbors could be quite far from the query point. We don't want distant points to make as much contribution to the model as nearby points. This suggests forming a weighted average of the predictions of each point. Write w_i for the weight at the i'th point. Then the estimate is

$$y_{pred} = \frac{\sum_i w_i y_i}{\sum_i w_i}.$$

A variety of weightings are reasonable choices. Write $d_i = \|(\mathbf{x} - \mathbf{x}_i)\|$ for the distance between the query point and the i'th nearest neighbor. Then inverse distance weighting uses $w_i = 1/d_i$. Alternatively, we could use an exponential function to strongly weight down more distant points, using

$$w_i = \exp\left(\frac{-d_i^2}{2\sigma^2}\right).$$

We will need to choose a scale σ, which can be done by cross-validation. Hold out some examples, make predictions at the held out examples using a variety of different scales, and choose the scale that gives the best held-out error. Alternatively,

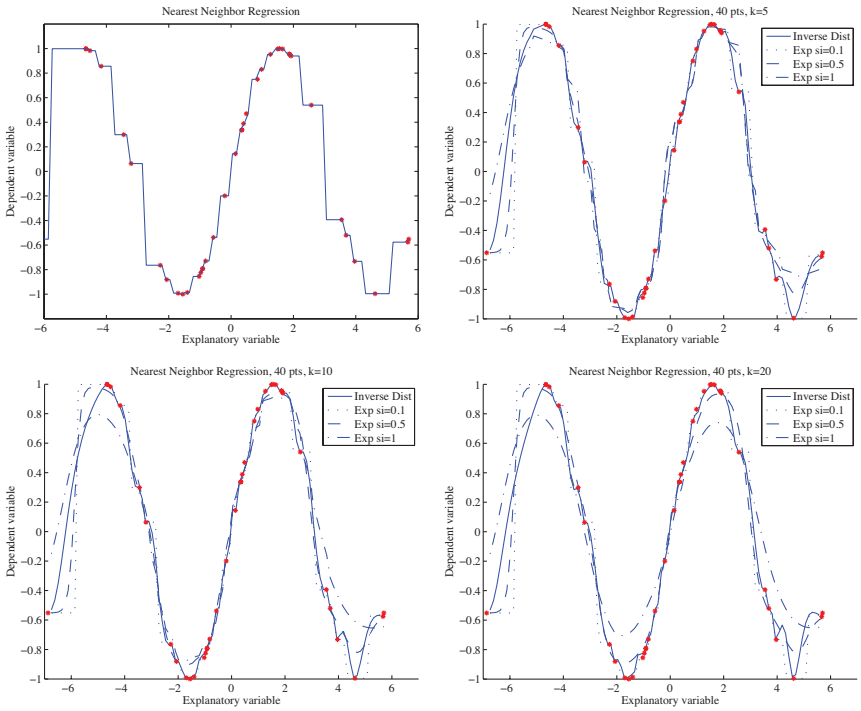

FIGURE 13.13: *Different forms of nearest neighbors regression, predicting y from a one-dimensional x, using a total of 40 training points.* **Top left:** *reporting the nearest neighbor leads to a piecewise constant function.* **Top right:** *improvements are available by forming a weighted average of the five nearest neighbors, using inverse distance weighting or exponential weighting with three different scales. Notice if the scale is small, then the regression looks a lot like nearest neighbors, and if it is too large, all the weights in the average are nearly the same (which leads to a piecewise constant structure in the regression).* **Bottom left** *and* **bottom right** *show that using more neighbors leads to a smoother regression.*

if there are enough nearest neighbors, we could form a distance weighted linear regression, then predict the value at the query point from that regression.

Each of these strategies presents some difficulties when $\mathbf{x}$ has high dimension. In that case, it is usual that the nearest neighbor is a lot closer than the second nearest neighbor. If this happens, then each of these weighted averages will boil down to evaluating the dependent variable at the nearest neighbor (because all the others will have very small weight in the average).

Remember this: *Nearest neighbors can be used for regression. In the simplest approach, you find the nearest neighbor to your feature vector, and take that neighbor's number as your prediction. More complex approaches smooth predictions over multiple neighbors.*

13.3.1 Using your Neighbors to Predict More than a Number

Linear regression takes some features and predicts a number. But in practice, one often wants to predict something more complex than a number. For example, I might want to predict a parse tree (which has combinatorial structure) from a sentence (the explanatory variables). As another example, I might want to predict a map of the shadows in an image (which has spatial structure) against an image (the explanatory variables). As yet another example, I might want to predict which direction to move the controls on a radio-controlled helicopter (which have to be moved together) against a path plan and the current state of the helicopter (the explanatory variables).

Looking at neighbors is a very good way to solve such problems. The general strategy is relatively simple. We find a large collection of pairs of training data. Write $\mathbf{x}_i$ for the explanatory variables for the i'th example, and $\mathbf{y}_i$ for the dependent variable in the i'th example. This dependent variable could be anything — it doesn't need to be a single number. It might be a tree, or a shadow map, or a word, or anything at all. I wrote it as a vector because I needed to choose some notation.

In the simplest, and most general, approach, we obtain a prediction for a new set of explanatory variables $\mathbf{x}$ by (a) finding the nearest neighbor and then (b) producing the dependent variable for that neighbor. We might vary the strategy slightly by using an approximate nearest neighbor. If the dependent variables have enough structure that it is possible to summarize a collection of different dependent variables, then we might recover the k nearest neighbors and summarize their dependent variables. How we summarize rather depends on the dependent variables. For example, it is a bit difficult to imagine the average of a set of trees, but quite straightforward to average images. If the dependent variable was a word, we might not be able to average words, but we can vote and choose the most popular word. If the dependent variable is a vector, we can compute either distance weighted averages or a distance weighted linear regression.

13.3.2 Example: Filling Large Holes with Whole Images

Many different kinds of user want to remove things from images or from video. Art directors might like to remove unattractive telephone wires; restorers might want to remove scratches or marks; there's a long history of government officials removing people with embarrassing politics from publicity pictures (see the fascinating examples in **?**); and home users might wish to remove a relative they dislike from a family picture. All these users must then find something to put in place of the

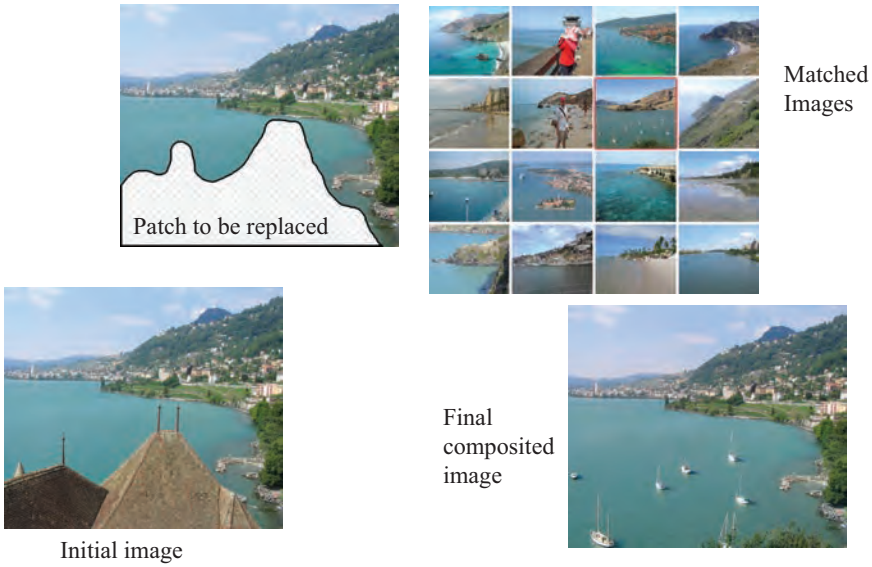

FIGURE 13.14: *We can fill large holes in images by matching the image to a collection, choosing one element of the collection, then cutting out an appropriate block of pixels and putting them into the hole in the query image. In this case, the hole has been made by an artist, who wishes to remove the roofline from the view. Notice how there are a range of objects (boats, water) that have been inserted into the hole. These objects are a reasonable choice, because the overall structures of the query and matched image are largely similar — getting an image that matches most of the query image supplies enough context to ensure that the rest "makes sense".*

pixels that were removed.

If one has a large hole in a large image, we may not be able to just extend a texture to fill the hole. Instead, entire objects might need to appear in the hole (Figure 13.14). There is a straightforward, and extremely effective, way to achieve this. We match the image to a large collection of images, to find the nearest neighbors (the details of the distance function are below). This yields a set of example images where all the pixels we didn't want to replace are close to those of the query image. From these, we choose one, and fill in the pixels from that image.

There are several ways to choose. If we wish to do so automatically, we could use the example with the smallest distance to the image. Very often, an artist is involved, and then we could prepare a series of alternatives — using, perhaps, the k closest examples — then show them to the artist, who will choose one. This method, which is very simple to describe, is extremely effective in practice.

It is straightforward to get a useful distance between images. We have an image with some missing pixels, and we wish to find nearby images. We will assume that all images are the same size. If this isn't in fact the case, we could either crop or resize the example images. A good measure of similarity between two images $\mathcal{A}$ and $\mathcal{B}$ can be measured by forming the **sum of squared differences** (or

SSD) of corresponding pixel values. You should think of an image as an array of pixels. If the images are grey-level images, then each pixel contains a single number, encoding the grey-level. If the images are color images, then each pixel (usually!) contains three values, one encoding the red-level, one encoding the green-level, and one encoding the blue-level. The SSD is computed as

$$\sum_{(i,j)} (\mathcal{A}_{ij} - \mathcal{B}_{ij})^2$$

where i and j range over all pixels. If the images are grey-level images, then by $(\mathcal{A}_{ij} - \mathcal{B}_{ij})^2$, I mean the squared difference between grey levels; if they are color images, then this means the sum of squared differences between red, green and blue values. This distance is small when the images are similar, and large when they are different (it is essentially the length of the difference vector).

Now we don't know some of the pixels in the query image. Write $\mathcal{K}$ for the set of pixels around a point whose values are known, and $\sharp\mathcal{K}$ for the size of this set. We can now use

$$\frac{1}{\sharp\mathcal{K}} \sum_{(i,j)\in\mathcal{K}} (\mathcal{A}_{ij} - \mathcal{B}_{ij})^2.$$

Filling in the pixels requires some care. One does not usually get the best results by just copying the missing pixels from the matched image into the hole. Instead, it is better to look for a good **seam**. We search for a curve enclosing the missing pixels which (a) is reasonably close to the boundary of the missing pixels and (b) gives a good boundary between the two images. A good boundary is one where the query image (on one side) is "similar to" the matched image (on the other side). A good sense of similarity requires that pixels match well, and that image gradients crossing the boundary tend to match too.

Remember this: *Nearest neighbors can be used to predict more than numbers.*

13.4 YOU SHOULD

13.4.1 remember these definitions:

13.4.2 remember these terms:

13.4.3 remember these facts:

13.4.4 remember these procedures:

APPENDIX: DATA

Batch A		Batch B		Batch C	
Amount of Hormone	Time in Service	Amount of Hormone	Time in Service	Amount of Hormone	Time in Service
25.8	99	16.3	376	28.8	119
20.5	152	11.6	385	22.0	188
14.3	293	11.8	402	29.7	115
23.2	155	32.5	29	28.9	88
20.6	196	32.0	76	32.8	58
31.1	53	18.0	296	32.5	49
20.9	184	24.1	151	25.4	150
20.9	171	26.5	177	31.7	107
30.4	52	25.8	209	28.5	125

TABLE 13.1: *A table showing the amount of hormone remaining and the time in service for devices from lot A, lot B and lot C. The numbering is arbitrary (i.e. there's no relationship between device 3 in lot A and device 3 in lot B). We expect that the amount of hormone goes down as the device spends more time in service, so cannot compare batches just by comparing numbers.*

PROBLEMS

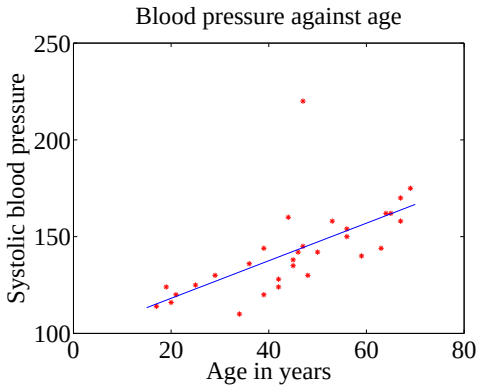

FIGURE 13.15: *A regression of blood pressure against age, for 30 data points.*

13.1. Figure 13.15 shows a linear regression of systolic blood pressure against age. There are 30 data points.

(a) Write $e_i = y_i - \mathbf{x}_i^T \beta$ for the residual. What is the $\mathsf{mean}\,(\{e\})$ for this regression?

(b) For this regression, $\mathsf{var}\,(\{y\}) = 509$ and the R^2 is 0.4324. What is $\mathsf{var}\,(\{e\})$ for this regression?

(c) How well does the regression explain the data?

(d) What could you do to produce better predictions of blood pressure (without actually measuring blood pressure)?

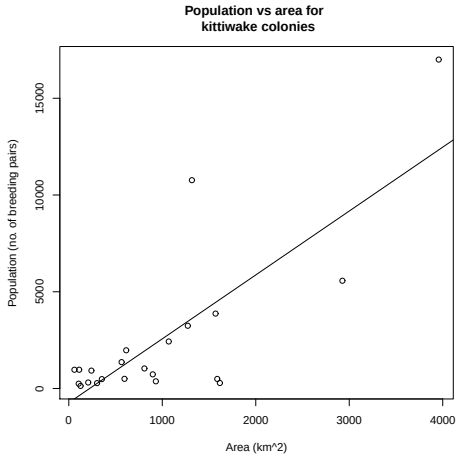

FIGURE 13.16: *A regression of the number of breeding pairs of kittiwakes against the area of an island, for 22 data points.*

13.2. At http://www.statsci.org/data/general/kittiwak.html, you can find a dataset collected by D.K. Cairns in 1988 measuring the area available for a seabird (black-legged kittiwake) colony and the number of breeding pairs for a variety of different colonies. Figure 13.16 shows a linear regression of the number of breeding pairs against the area. There are 22 data points.

(a) Write $e_i = y_i - \mathbf{x}_i^T \beta$ for the residual. What is the $\mathsf{mean}(\{e\})$ for this regression?

(b) For this regression, $\mathsf{var}(\{y\}) = 16491357$ and the R^2 is 0.62. What is $\mathsf{var}(\{e\})$ for this regression?

(c) How well does the regression explain the data? If you had a large island, to what extent would you trust the prediction for the number of kittiwakes produced by this regression? If you had a small island, would you trust the answer more?

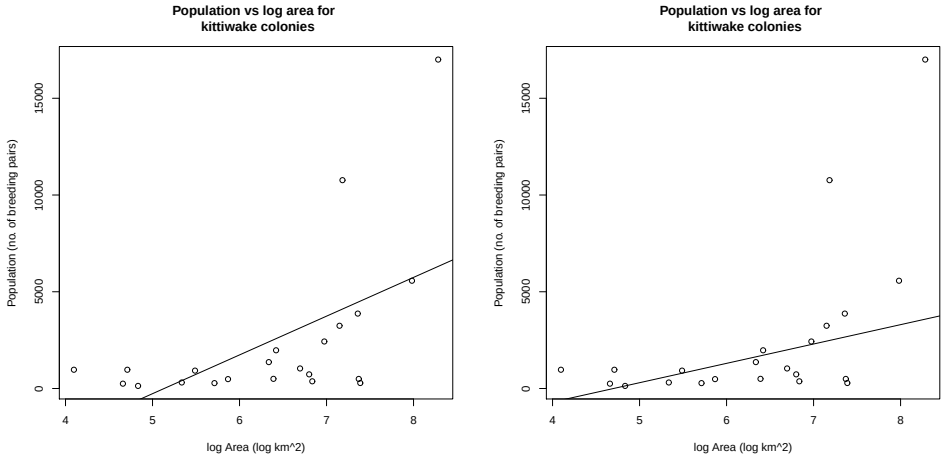

FIGURE 13.17: **Left:** *A regression of the number of breeding pairs of kittiwakes against the log of area of an island, for 22 data points.* **Right:** *A regression of the number of breeding pairs of kittiwakes against the log of area of an island, for 22 data points, using a method that ignores two likely outliers.*

13.3. At http://www.statsci.org/data/general/kittiwak.html, you can find a dataset collected by D.K. Cairns in 1988 measuring the area available for a seabird (black-legged kittiwake) colony and the number of breeding pairs for a variety of different colonies. Figure 13.17 shows a linear regression of the number of breeding pairs against the log of area. There are 22 data points.

(a) Write $e_i = y_i - \mathbf{x}_i^T \beta$ for the residual. What is the $\mathsf{mean}(\{e\})$ for this regression?

(b) For this regression, $\mathsf{var}(\{y\}) = 16491357$ and the R^2 is 0.31. What is $\mathsf{var}(\{e\})$ for this regression?

(c) How well does the regression explain the data? If you had a large island, to what extent would you trust the prediction for the number of kittiwakes produced by this regression? If you had a small island, would you trust the answer more? Why?

(d) Figure 13.17 shows the result of a linear regression that ignores two likely outliers. Would you trust the predictions of this regression more? Why?

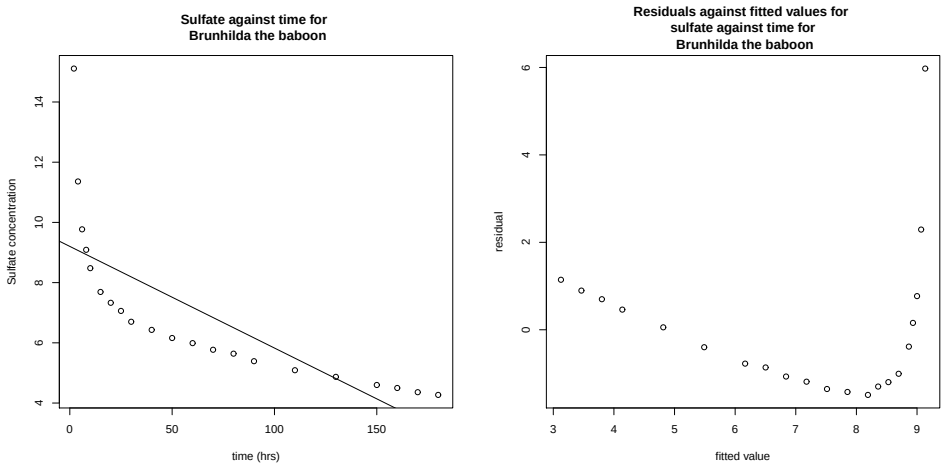

FIGURE 13.18: **Left:** *A regression of the concentration of sulfate in the blood of Brunhilda the baboon against time.* **Right:** *For this regression, a plot of residual against fitted value.*

13.4. At http://www.statsci.org/data/general/brunhild.html, you will find a dataset that measures the concentration of a sulfate in the blood of a baboon named Brunhilda as a function of time. Figure 13.18 plots this data, with a linear regression of the concentration against time. I have shown the data, and also a plot of the residual against the predicted value. The regression appears to be unsuccessful.

(a) What suggests the regression has problems?

(b) What is the cause of the problem, and why?

(c) What could you do to improve the problems?

13.5. Assume we have a dataset where $\mathbf{Y} = \mathcal{X}\beta + \xi$, for some unknown β and ξ. The term ξ is a normal random variable with zero mean, and covariance $\sigma^2\mathcal{I}$ (i.e. this data really does follow our model).

(a) Write $\hat{\beta}$ for the estimate of β recovered by least squares, and $\hat{\mathbf{Y}}$ for the values predicted by our model for the training data points. Show that

$$\hat{\mathbf{Y}} = \mathcal{X}\left(\mathcal{X}^T\mathcal{X}\right)^{-1}\mathcal{X}^T\mathbf{Y}$$

(b) Show that

$$\mathbb{E}[\hat{y}_i - y_i] = 0$$

for each training data point y_i, where the expectation is over the probability distribution of ξ.

(c) Show that

$$\mathbb{E}\left[(\hat{\beta} - \beta)\right] = 0$$

where the expectation is over the probability distribution of ξ.

13.6. In this exercise, I will show that the prediction process of chapter 3(see page 65) is a linear regression with two independent variables. Assume we have N data items which are 2-vectors $(x_1, y_1), \ldots, (x_N, y_N)$, where $N > 1$. These could be obtained, for example, by extracting components from larger vectors. As usual, we will write $\hat{x}_i$ for x_i in normalized coordinates, and so on. The correlation coefficient is r (this is an important, traditional notation).

(a) Assume that we have an x_o, for which we wish to predict a y value. Show that the value of the prediction obtained using the method of page 66 is

$$\begin{aligned} y_{\text{pred}} &= \frac{\mathsf{std}\,(y)}{\mathsf{std}\,(x)} r(x_0 - \mathsf{mean}\,(\{x\})) + \mathsf{mean}\,(\{y\}) \\ &= \left(\frac{\mathsf{std}\,(y)}{\mathsf{std}\,(x)} r\right) x_0 + \left(\mathsf{mean}\,(\{y\}) - \frac{\mathsf{std}\,(x)}{\mathsf{std}\,(y)}\mathsf{mean}\,(\{x\})\right). \end{aligned}$$

(b) Show that

$$\begin{aligned} r &= \frac{\mathsf{mean}\,(\{(x - \mathsf{mean}\,(\{x\}))(y - \mathsf{mean}\,(\{y\}))\})}{\mathsf{std}\,(x)\mathsf{std}\,(y)} \\ &= \frac{\mathsf{mean}\,(\{xy\}) - \mathsf{mean}\,(\{x\})\mathsf{mean}\,(\{y\})}{\mathsf{std}\,(x)\mathsf{std}\,(y)}. \end{aligned}$$

(c) Now write

$$\mathcal{X} = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \ldots & \ldots \\ x_n & 1 \end{pmatrix} \text{ and } \mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \ldots \\ y_n \end{pmatrix}.$$

The coefficients of the linear regression will be $\hat{\beta}$, where $\mathcal{X}^T\mathcal{X}\hat{\beta} = \mathcal{X}^T\mathbf{Y}$. Show that

$$\begin{aligned} \mathcal{X}^T\mathcal{X} &= N\begin{pmatrix} \mathsf{mean}\left(\left\{x^2\right\}\right) & \mathsf{mean}\,(\{x\}) \\ \mathsf{mean}\,(\{x\}) & 1 \end{pmatrix} \\ &= N\begin{pmatrix} \mathsf{std}\,(x)^2 + \mathsf{mean}\,(\{x\})^2 & \mathsf{mean}\,(\{x\}) \\ \mathsf{mean}\,(\{x\}) & 1 \end{pmatrix} \end{aligned}$$

(d) Now show that

$$\begin{aligned}\mathcal{X}^T\mathbf{Y} &= N\begin{pmatrix} \mathsf{mean}\left(\{xy\}\right) \\ \mathsf{mean}\left(\{y\}\right)\end{pmatrix} \\ &= N\begin{pmatrix} \mathsf{std}\left(x\right)\mathsf{std}\left(y\right)r + \mathsf{mean}\left(\{x\}\right)\mathsf{mean}\left(\{y\}\right) \\ \mathsf{mean}\left(\{y\}\right)\end{pmatrix}.\end{aligned}$$

(e) Now show that

$$\left(\mathcal{X}^T\mathcal{X}\right)^{-1} = \frac{1}{N}\frac{1}{\mathsf{std}\left(x\right)^2}\begin{pmatrix} 1 & -\mathsf{mean}\left(\{x\}\right) \\ -\mathsf{mean}\left(\{x\}\right) & \mathsf{std}\left(x\right)^2 + \mathsf{mean}\left(\{x\}\right)^2\end{pmatrix}$$

(f) Now (finally!) show that if $\hat{\beta}$ is the solution to $\mathcal{X}^T\mathcal{X}\hat{\beta} - \mathcal{X}^T\mathbf{Y} = 0$, then

$$\hat{\beta} = \begin{pmatrix} r\frac{\mathsf{std}(y)}{\mathsf{std}(x)} \\ \mathsf{mean}\left(\{y\}\right) - \left(r\frac{\mathsf{std}(y)}{\mathsf{std}(x)}\right)\mathsf{mean}\left(\{x\}\right)\end{pmatrix}$$

and use this to argue that the process of page 65 is a linear regression with two independent variables.

13.7. This exercise investigates the effect of correlation on a regression. Assume we have N data items, $(\mathbf{x}_i, y_i)$. We will investigate what happens when the data have the property that the first component is relatively accurately predicted by the other components. Write x_{i1} for the first component of $\mathbf{x}_i$, and $\mathbf{x}_{i,\hat{1}}$ for the vector obtained by deleting the first component of $\mathbf{x}_i$. Choose $\mathbf{u}$ to predict the first component of the data from the rest *with minimum error*, so that $x_{i1} = \mathbf{x}_{i\hat{1}}^T\mathbf{u} + w_i$. The error of prediction is w_i. Write $\mathbf{w}$ for the vector of errors (i.e. the i'th component of $\mathbf{w}$ is w_i). Because $\mathbf{w}^T\mathbf{w}$ is minimized by choice of $\mathbf{u}$, we have $\mathbf{w}^T\mathbf{1} = 0$ (i.e. the average of the w_i's is zero). Assume that these predictions are very good, so that there is some small positive number ϵ so that $\mathbf{w}^T\mathbf{w} \leq \epsilon$.

(a) Write $\mathbf{a} = [-1, \mathbf{u}]^T$. Show that

$$\mathbf{a}^T\mathcal{X}^T\mathcal{X}\mathbf{a} \leq \epsilon.$$

(b) Now show that the smallest eigenvalue of $\mathcal{X}^T\mathcal{X}$ is less than or equal to ϵ.

(c) Write $s_k = \sum_u x_{uk}^2$, and $s\mathrm{max}$ for $\max(s_1, \ldots, s_d)$. Show that the largest eigenvalue of $\mathcal{X}^T\mathcal{X}$ is greater than or equal to $s\mathrm{max}$.

(d) The **condition number** of a matrix is the ratio of largest to smallest eigenvalue of a matrix. Use the information above to bound the **condition number** of $\mathcal{X}^T\mathcal{X}$.

(e) Assume that $\hat{\beta}$ is the solution to $\mathcal{X}^T\mathcal{X}\hat{\beta} = \mathcal{X}^T\mathbf{Y}$. Show that the

$$(\mathcal{X}^T\mathbf{Y} - \mathcal{X}^T\mathcal{X}(\hat{\beta} + \mathbf{a}))^T(\mathcal{X}^T\mathbf{Y} - \mathcal{X}^T\mathcal{X}(\hat{\beta} + \mathbf{a}))$$

(for $\mathbf{a}$ as above) is bounded above by

$$\epsilon^2(1 + \mathbf{u}^T\mathbf{u})$$

(f) Use the last sub exercises to explain why correlated data will lead to a poor estimate of $\hat{\beta}$.

13.8. This exercise explores the effect of regularization on a regression. Assume we have N data items, $(\mathbf{x}_i, y_i)$. We will investigate what happens when the data have the property that the first component is relatively accurately predicted by the other components. Write x_{i1} for the first component of $\mathbf{x}_i$, and $\mathbf{x}_{i,\hat{1}}$ for the vector obtained by deleting the first component of $\mathbf{x}_i$. Choose $\mathbf{u}$ to predict the first component of the data from the rest *with minimum error*, so that $x_{i1} = \mathbf{x}_{i\hat{1}}^T\mathbf{u} + w_i$. The error of prediction is w_i. Write $\mathbf{w}$ for the vector of errors (i.e. the i'th component of $\mathbf{w}$ is w_i). Because $\mathbf{w}^T\mathbf{w}$ is minimized by choice of $\mathbf{u}$, we have $\mathbf{w}^T\mathbf{1} = 0$ (i.e. the average of the w_i's is zero). Assume that these predictions are very good, so that there is some small positive number ϵ so that $\mathbf{w}^T\mathbf{w} \leq \epsilon$.

(a) Show that, *for any vector* $\mathbf{v}$,

$$\mathbf{v}^T\left(\mathcal{X}^T\mathcal{X} + \lambda\mathcal{I}\right)\mathbf{v} \geq \lambda\mathbf{v}^T\mathbf{v}$$

and use this to argue that the smallest eigenvalue of $\left(\mathcal{X}^T\mathcal{X} + \lambda\mathcal{I}\right)$ is greater than λ.

(b) Write $\mathbf{b}$ for an eigenvector of $\mathcal{X}^T\mathcal{X}$ with eigenvalue $\lambda_{\mathbf{b}}$. Show that $\mathbf{b}$ is an eigenvector of $\left(\mathcal{X}^T\mathcal{X} + \lambda\mathcal{I}\right)$ with eigenvalue $\lambda_{\mathbf{b}} + \lambda$.

(c) Recall $\mathcal{X}^T\mathcal{X}$ is a $d \times d$ matrix which is symmetric, and so has d orthonormal eigenvectors. Write $\mathbf{b}_i$ for the i'th such vector, and $\lambda_{\mathbf{b}_i}$ for the corresponding eigenvalue. Show that

$$\mathcal{X}^T\mathcal{X}\beta - \mathcal{X}^T\mathbf{Y} = 0$$

is solved by

$$\beta = \sum_{i=1}^{d} \frac{\mathbf{Y}^T\mathcal{X}\mathbf{b}_i}{\lambda_{\mathbf{b}_i}}.$$

(d) Using the notation of the previous sub exercise, show that

$$(\mathcal{X}^T\mathcal{X} + \lambda\mathcal{I})\beta - \mathcal{X}^T\mathbf{Y} = 0$$

is solved by

$$\beta = \sum_{i=1}^{d} \frac{\mathbf{Y}^T\mathcal{X}\mathbf{b}_i}{\lambda_{\mathbf{b}_i} + \lambda}.$$

Use this expression to explain why a regularized regression may produce better results on test data than an unregularized regression.

PROGRAMMING EXERCISES

13.9. At http://www.statsci.org/data/general/brunhild.html, you will find a dataset that measures the concentration of a sulfate in the blood of a baboon named Brunhilda as a function of time. Build a linear regression of the log of the concentration against the log of time.

(a) Prepare a plot showing (a) the data points and (b) the regression line in log-log coordinates.

(b) Prepare a plot showing (a) the data points and (b) the regression curve in the original coordinates.

(c) Plot the residual against the fitted values in log-log and in original coordinates.

(d) Use your plots to explain whether your regression is good or bad and why.

13.10. At http://www.statsci.org/data/oz/physical.html, you will find a dataset of measurements by M. Larner, made in 1996. These measurements include body mass, and various diameters. Build a linear regression of predicting the body mass from these diameters.

(a) Plot the residual against the fitted values for your regression.

(b) Now regress the cube root of mass against these diameters. Plot the residual against the fitted values in both these cube root coordinates and in the original coordinates.

(c) Use your plots to explain which regression is better.

13.11. At https://archive.ics.uci.edu/ml/datasets/Abalone, you will find a dataset of measurements by W. J. Nash, T. L. Sellers, S. R. Talbot, A. J. Cawthorn and W. B. Ford, made in 1992. These are a variety of measurements of blacklip abalone (*Haliotis rubra*; delicious by repute) of various ages and genders.

(a) Build a linear regression predicting the age from the measurements, ignoring gender. Plot the residual against the fitted values.

(b) Build a linear regression predicting the age from the measurements, including gender. There are three levels for gender; I'm not sure whether this has to do with abalone biology or difficulty in determining gender. You can represent gender numerically by choosing 1 for one level, 0 for another, and -1 for the third. Plot the residual against the fitted values.

(c) Now build a linear regression predicting the log of age from the measurements, ignoring gender. Plot the residual against the fitted values.

(d) Now build a linear regression predicting the log age from the measurements, including gender, represented as above. Plot the residual against the fitted values.

(e) It turns out that determining the age of an abalone is possible, but difficult (you section the shell, and count rings). Use your plots to explain which regression you would use to replace this procedure, and why.

(f) Can you improve these regressions by using a regularizer? Use glmnet to obtain plots of the cross-validated prediction error.

PART FIVE

SOME MATHEMATICAL BACKGROUND

CHAPTER 14

Resources

14.1 USEFUL MATERIAL ABOUT MATRICES

Terminology:

- A matrix $\mathcal{M}$ is **symmetric** if $\mathcal{M} = \mathcal{M}^T$. A symmetric matrix is necessarily square.
- We write $\mathcal{I}$ for the identity matrix.
- A matrix is **diagonal** if the only non-zero elements appear on the diagonal. A diagonal matrix is necessarily symmetric.
- A symmetric matrix is **positive semidefinite** if, for any $\mathbf{x}$ such that $\mathbf{x}^T\mathbf{x} > 0$ (i.e. this vector has at least one non-zero component), we have $\mathbf{x}^T\mathcal{M}\mathbf{x} \geq 0$.
- A symmetric matrix is **positive definite** if, for any $\mathbf{x}$ such that $\mathbf{x}^T\mathbf{x} > 0$, we have $\mathbf{x}^T\mathcal{M}\mathbf{x} > 0$.
- A matrix $\mathcal{R}$ is **orthonormal** if $\mathcal{R}^T\mathcal{R} = \mathcal{I} = \mathcal{I}^T = \mathcal{R}\mathcal{R}^T$. Orthonormal matrices are necessarily square.

Orthonormal matrices: You should think of orthonormal matrices as rotations, because they do not change lengths or angles. For $\mathbf{x}$ a vector, $\mathcal{R}$ an orthonormal matrix, and $\mathbf{u} = \mathcal{R}\mathbf{x}$, we have $\mathbf{u}^T\mathbf{u} = \mathbf{x}^T\mathcal{R}^T\mathcal{R}\mathbf{x} = \mathbf{x}^T\mathcal{I}\mathbf{x} = \mathbf{x}^T\mathbf{x}$. This means that $\mathcal{R}$ doesn't change lengths. For $\mathbf{y}$, $\mathbf{z}$ both unit vectors, we have that the cosine of the angle between them is $\mathbf{y}^T\mathbf{x}$; but, by the same argument as above, the inner product of $\mathcal{R}\mathbf{y}$ and $\mathcal{R}\mathbf{x}$ is the same as $\mathbf{y}^T\mathbf{x}$. This means that $\mathcal{R}$ doesn't change angles, either.

Eigenvectors and Eigenvalues: Assume $\mathcal{S}$ is a $d \times d$ symmetric matrix, $\mathbf{v}$ is a $d \times 1$ vector, and λ is a scalar. If we have

$$\mathcal{S}\mathbf{v} = \lambda\mathbf{v}$$

then $\mathbf{v}$ is referred to as an **eigenvector** of $\mathcal{S}$ and λ is the corresponding **eigenvalue**. Matrices don't have to be symmetric to have eigenvectors and eigenvalues, but the symmetric case is the only one of interest to us.

In the case of a symmetric matrix, the eigenvalues are real numbers, and there are d distinct eigenvectors that are normal to one another, and can be scaled to have unit length. They can be stacked into a matrix $\mathcal{U} = [\mathbf{v}_1, \ldots, \mathbf{v}_d]$. This matrix is orthonormal, meaning that $\mathcal{U}^T\mathcal{U} = \mathcal{I}$. This means that there is a diagonal matrix Λ such that

$$\mathcal{S}\mathcal{U} = \mathcal{U}\Lambda.$$

In fact, there is a large number of such matrices, because we can reorder the eigenvectors in the matrix $\mathcal{U}$, and the equation still holds with a new Λ, obtained by reordering the diagonal elements of the original Λ. There is no reason to keep track of this complexity. Instead, we adopt the convention that the elements of $\mathcal{U}$ are always ordered so that the elements of Λ are sorted along the diagonal, with the largest value coming first.

Diagonalizing a symmetric matrix: This gives us a particularly important procedure. We can convert any symmetric matrix $\mathcal{S}$ to a diagonal form by computing

$$\mathcal{U}^T \mathcal{S} \mathcal{U} = \Lambda.$$

This procedure is referred to as **diagonalizing** a matrix. Again, we assume that the elements of $\mathcal{U}$ are always ordered so that the elements of Λ are sorted along the diagonal, with the largest value coming first. Diagonalization allows us to show that positive definiteness is equivalent to having all positive eigenvalues, and positive semidefiniteness is equivalent to having all non-negative eigenvalues.

Factoring a matrix: Assume that $\mathcal{S}$ is symmetric and positive semidefinite. We have that

$$\mathcal{S} = \mathcal{U} \Lambda \mathcal{U}^T$$

and all the diagonal elements of Λ are non-negative. Now construct a diagonal matrix whose diagonal entries are the positive square roots of the diagonal elements of Λ; call this matrix $\Lambda^{(1/2)}$. We have $\Lambda^{(1/2)}\Lambda^{(1/2)} = \Lambda$ and $(\Lambda^{(1/2)})^T = \Lambda^{(1/2)}$. Then we have that

$$\mathcal{S} = (\mathcal{U}\Lambda^{(1/2)})(\Lambda^{(1/2)}\mathcal{U}^T) = (\mathcal{U}\Lambda^{(1/2)})(\mathcal{U}\Lambda^{(1/2)})^T$$

so we can factor $\mathcal{S}$ into the form $\mathcal{X}\mathcal{X}^T$ by computing the eigenvectors and eigenvalues.

14.1.1 The Singular Value Decomposition

For any $m \times p$ matrix $\mathcal{X}$, it is possible to obtain a decomposition

$$\mathcal{X} = \mathcal{U} \Sigma \mathcal{V}^T$$

where $\mathcal{U}$ is $m \times m$, $\mathcal{V}$ is $p \times p$, and Σ is $m \times p$ and is diagonal. If you don't recall what a diagonal matrix looks like when the matrix *isn't* square, it's simple. All entries are zero, except the i, i entries for i in the range 1 to $\min(m, p)$. So if Σ is tall and thin, the top square is diagonal and everything else is zero; if Σ is short and wide, the left square is diagonal and everything else is zero. Both $\mathcal{U}$ and $\mathcal{V}$ are orthonormal (i.e. $\mathcal{U}\mathcal{U}^T = \mathcal{I}$ and $\mathcal{V}\mathcal{V}^T = \mathcal{I}$).

Notice that there is a relationship between forming an SVD and diagonalizing a matrix. In particular, $\mathcal{X}^T\mathcal{X}$ is symmetric, and it can be diagonalized as

$$\mathcal{X}^T\mathcal{X} = \mathcal{V}\Sigma^T\Sigma\mathcal{V}^T.$$

Similarly, $\mathcal{X}\mathcal{X}^T$ is symmetric, and it can be diagonalized as

$$\mathcal{X}\mathcal{X}^T = \mathcal{U}\Sigma\Sigma^T\mathcal{U}.$$

14.1.2 Approximating A Symmetric Matrix

Assume we have a $k \times k$ symmetric matrix $\mathcal{T}$, and we wish to construct a matrix $\mathcal{A}$ that approximates it. We require that (a) the rank of $\mathcal{A}$ is precisely $r < k$ and (b) the approximation should minimize the **Frobenius norm**, that is,

$$\| (\mathcal{T} - \mathcal{A}) \|_F^2 = \sum_{ij} (T_{ij} - A_{ij})^2.$$

It turns out that there is a straightforward construction that yields $\mathcal{A}$.

The first step is to notice that if $\mathcal{U}$ is orthonormal and $\mathcal{M}$ is any matrix, then

$$\| \mathcal{U}\mathcal{M} \|_F = \| \mathcal{M}\mathcal{U} \|_F = \| \mathcal{M} \|_F.$$

This is true because $\mathcal{U}$ is a rotation (as is $\mathcal{U}^T = \mathcal{U}^{-1}$), and rotations do not change the length of vectors. So, for example, if we write $\mathcal{M}$ as a table of row vectors $\mathcal{M} = [\mathbf{m}_1, \mathbf{m}_2, ...\mathbf{m}_k]$, then $\mathcal{U}\mathcal{M} = [\mathcal{U}\mathbf{m}_1, \mathcal{U}\mathbf{m}_2, ...\mathcal{U}\mathbf{m}_k]$. Now $\| \mathcal{M} \|_F^2 = \sum_{j=1}^{k} \| \mathbf{m}_j \|^2$, so $\| \mathcal{U}\mathcal{M} \|_F^2 = \sum_{i=1}^{k} \| \mathcal{U}\mathbf{m}_k \|^2$. But rotations do not change lengths, so $\| \mathcal{U}\mathbf{m}_k \|^2 = \| \mathbf{m}_k \|^2$, and so $\| \mathcal{U}\mathcal{M} \|_F = \| \mathcal{M} \|_F$. To see the result for the case of $\mathcal{M}\mathcal{U}$, just think of $\mathcal{M}$ as a table of row vectors.

Notice that, if $\mathcal{U}$ is the orthonormal matrix whose columns are eigenvectors of $\mathcal{T}$, then we have

$$\| (\mathcal{T} - \mathcal{A}) \|_F^2 = \| \mathcal{U}^T (\mathcal{T} - \mathcal{A}) \mathcal{U} \|_F^2.$$

Now write Λ_r for $\mathcal{U}^T \mathcal{A} \mathcal{U}$, and Λ for the diagonal matrix of eigenvalues of $\mathcal{T}$. Then we have

$$\| (\mathcal{T} - \mathcal{A}) \|_F^2 = \| \Lambda - \Lambda_A \|_F^2,$$

an expression that is easy to solve for Λ_A. We know that Λ is diagonal, so the best Λ_A is diagonal, too. The rank of $\mathcal{A}$ must be r, so the rank of Λ_A must be r as well. To get the best Λ_A, we keep the r largest diagonal values of Λ, and set the rest to zero; Λ_A has rank r because it has only r non-zero entries on the diagonal, and every other entry is zero.

Now to recover $\mathcal{A}$ from Λ_A, we know that $\mathcal{U}^T\mathcal{U} = \mathcal{U}\mathcal{U}^T = \mathcal{I}$ (remember, $\mathcal{I}$ is the identity). We have $\Lambda_A = \mathcal{U}^T \mathcal{A} \mathcal{U}$, so

$$\mathcal{A} = \mathcal{U} \Lambda_A \mathcal{U}^T.$$

We can clean up this representation in a useful way. Notice that only the first r columns of $\mathcal{U}$ (and the corresponding rows of $\mathcal{U}^T$) contribute to $\mathcal{A}$. The remaining $k - r$ are each multiplied by one of the zeros on the diagonal of Λ_A. Remember that, by convention, Λ was sorted so that the diagonal values are in descending order (i.e. the largest value is in the top left corner). We now keep only the top left $r \times r$ block of Λ_A, which we write Λ_r. We then write $\mathcal{U}_r$ for the $k \times r$ matrix consisting of the first r columns of $\mathcal{U}$. Then

$$\mathcal{A} = \mathcal{U}_r \Lambda_r \mathcal{U}^T$$

This is so useful a result, I have displayed it in a box; you should remember it.

Procedure: 14.1 *Approximating a symmetric matrix with a low rank matrix*

Assume we have a symmetric $k \times k$ matrix $\mathcal{T}$. We wish to approximate $\mathcal{T}$ with a matrix $\mathcal{A}$ that has rank $r < k$. Write $\mathcal{U}$ for the matrix whose columns are eigenvectors of $\mathcal{T}$, and Λ for the diagonal matrix of eigenvalues of $\mathcal{A}$ (so $\mathcal{A}\mathcal{U} = \mathcal{U}\Lambda$). Remember that, by convention, Λ was sorted so that the diagonal values are in descending order (i.e. the largest value is in the top left corner).
Now construct Λ_r from Λ by setting the $k - r$ smallest values of Λ to zero, and keeping only the top left $r \times r$ block. Construct $\mathcal{U}_r$, the $k \times r$ matrix consisting of the first r columns of $\mathcal{U}$. Then

$$\mathcal{A} = \mathcal{U}_r \Lambda_r \mathcal{U}_r^T$$

is the best possible rank r approximation to $\mathcal{T}$ in the Frobenius norm.

Now if $\mathcal{A}$ is positive semidefinite (i.e. if at least the r largest eigenvalues of $\mathcal{T}$ are non-negative), then we can factor $\mathcal{A}$ as in the previous section. This yields a procedure to approximate a symmetric matrix by factors. This is so useful a result, I have displayed it in a box; you should remember it.

Procedure: 14.2 *Approximating a symmetric matrix with low dimensional factors*

Assume we have a symmetric $k \times k$ matrix $\mathcal{T}$. We wish to approximate $\mathcal{T}$ with a matrix $\mathcal{A}$ that has rank $r < k$. We assume that at least the r largest eigenvalues of $\mathcal{T}$ are non-negative. Write $\mathcal{U}$ for the matrix whose columns are eigenvectors of $\mathcal{T}$, and Λ for the diagonal matrix of eigenvalues of $\mathcal{A}$ (so $\mathcal{A}\mathcal{U} = \mathcal{U}\Lambda$). Remember that, by convention, Λ was sorted so that the diagonal values are in descending order (i.e. the largest value is in the top left corner).
Now construct Λ_r from Λ by setting the $k - r$ smallest values of Λ to zero and keeping only the top left $r \times r$ block. Construct $\Lambda_r^{(1/2)}$ by replacing each diagonal element of Λ with its positive square root. Construct $\mathcal{U}_r$, the $k \times r$ matrix consisting of the first r columns of $\mathcal{U}$. Then write $\mathcal{V} = (\mathcal{U}_r \Lambda_r^{(1/2)})$

$$\mathcal{A} = \mathcal{V}\mathcal{V}^T$$

is the best possible rank r approximation to $\mathcal{T}$ in the Frobenius norm.

14.2 SOME SPECIAL FUNCTIONS

Error functions and Gaussians: The **error function** is defined by

$$\mathsf{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

and programming environments can typically evaluate the error function. This fact is made useful to us by a simple change of variables. We get

$$\frac{1}{\sqrt{2\pi}} \int_0^x e^{\frac{-u^2}{2}} du = \frac{1}{\sqrt{\pi}} \int_0^{\frac{x}{\sqrt{2}}} e^{-t^2} dt = \frac{1}{2}\mathsf{erf}\left(\frac{x}{\sqrt{2}}\right).$$

A particularly useful manifestation of this fact comes by noticing that

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^0 e^{\frac{-t^2}{2}} dt = 1/2$$

(because $\frac{1}{\sqrt{2\pi}} e^{\frac{-u^2}{2}}$ is a probability density function, and is symmetric about 0). As a result, we get

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{\frac{-t^2}{2}} dt = 1/2(1 + \mathsf{erf}\left(\frac{x}{\sqrt{2}}\right)).$$

Inverse error functions: We sometimes wish to know the value of x such that

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{\frac{-t^2}{2}} dt = p$$

for some given p. The relevant function of p is known as the **probit function** or the **normal quantile function**. We write

$$x = \Phi(p).$$

The probit function Φ can be expressed in terms of the **inverse error function**. Most programming environments can evaluate the inverse error function (which is the inverse of the error function). We have that

$$\Phi(p) = \sqrt{2}\mathsf{erf}^{-1}(2p - 1).$$

One problem we solve with some regularity is: choose u such that

$$\int_{-u}^u \frac{1}{\sqrt{2\pi}} \exp(-x^2/2) dx = p.$$

Notice that

$$\begin{aligned} \frac{p}{2} &= \frac{1}{\sqrt{2\pi}} \int_0^u e^{\frac{-t^2}{2}} dt \\ &= \frac{1}{2}\mathsf{erf}\left(\frac{u}{\sqrt{2}}\right) \end{aligned}$$

so that

$$u = \sqrt{2}\mathsf{erf}^{-1}(p).$$

Gamma functions:

The gamma function $\Gamma(x)$ is defined by a series of steps. First, we have that for n an integer,

$$\Gamma(n) = (n-1)!$$

and then for z a complex number with positive real part (which includes positive real numbers), we have

$$\Gamma(z) = \int_0^\infty t^z \frac{e^{-t}}{t} dt.$$

By doing this, we get a function on positive real numbers that is a smooth interpolate of the factorial function. We won't do any real work with this function, so won't expand on this definition. In practice, we'll either look up a value in tables or require a software environment to produce it.

14.3 FINDING NEAREST NEIGHBORS

To build a nearest neighbors classifier, we need to find the members of a set of high dimensional vectors that are closest to some query vector. It turns out this is a general, and quite difficult, problem. A linear search through the dataset is fine for a small set of data items, but we will operate at scales where we need something more efficient. Surprisingly, exact solutions turn out to be only very slightly more efficient than a linear search through the dataset. Approximate solutions are much more efficient. They are approximate in the sense that, with high probability, they return a point that is almost as close to the query as the closest point. The main trick to obtaining a good approximate solution is to carve the space into cells, then look at items that lie in cells near the query vector; there are two methods that are worth discussing in detail here.

In **locality sensitive hashing**, we build a set of hash tables containing the data items, using different hashing functions for each table. For a query item, we recover whatever is in each hash table at the location corresponding to the hash code computed for the query item. We search this set, keeping any data items from this set that are sufficiently close to the query. There are many choices of hash function; the most widely used in vision is **random projection**. Write $\mathbf{v}$ for a vector, representing either a query or a data item. We now obtain a single bit of a hash code by choosing a random vector $\mathbf{r}$ and then computing $sign(\mathbf{v} \cdot \mathbf{r})$. Computing a k-bit hash code involves choosing k such random vectors, then computing one bit for each. There is a set of k such random vectors associated with each hash table. Geometrically, choosing an $\mathbf{r}$ corresponds to choosing a hyperplane in the data space, and the hashing bit corresponds to which side of the hyperplane $\mathbf{v}$ lies on. A k-bit hash code identifies a cell in an arrangement of k hyperplanes in which $\mathbf{v}$ lies. k will be small compared to the dimension, and so we are cutting the space into 2^k cells. This means that there will be relatively few data items that lie in the same cell as a query. Some nearby data items may not lie in the same cell, because they could be on the other side of a hyperplane, but these items should lie in the same cell in another hash table.

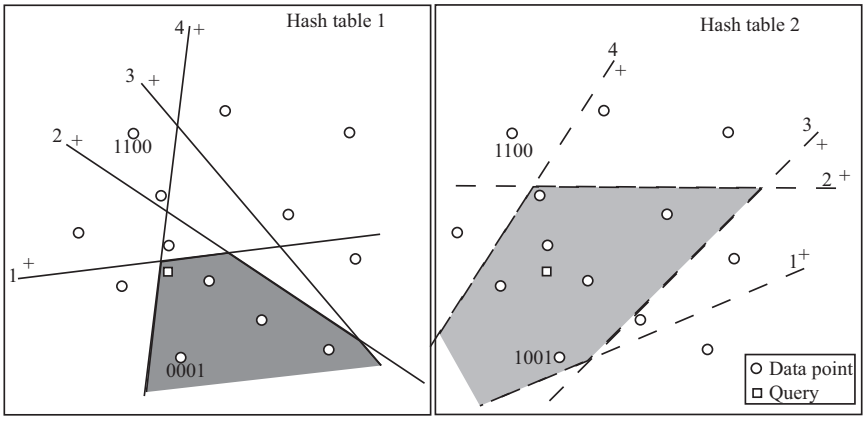

FIGURE 14.1: *In locality sensitive hashing using a random projection hash function, the hash function is equivalent to a hyperplane in the data space. Items that lie on one side of the hyperplane corresponding to the nth bit have that bit set to one; otherwise, it is zero. These hyperplanes cut the space of data into a set of cells. All the data items in a cell get a binary hash code (shown for two points in each figure; we have marked the order of the bits by labeling the hyperplanes, and the +s show which side of the hyperplane gets a one). To query, we find all data items in the same hash table entry as the query (the filled polygons in the figure), and then find the closest. However, the nearest neighbor might not be in this cell (for example, the case on the* **left***). To reduce the probability of error from this cause, we use more than one hash table and search the union of the sets of points lying in the query cell. In the case illustrated, the nearest neighbor of the query lies in the query cell for the second hash table, on the* **right***. The hash tables reduce the set of points we need to search, with high probability of finding a point that is almost as close as the nearest neighbor.*

All these assertions can be made precise, resulting in a guarantee that: (a) a data item that is almost as close as the nearest neighbor will be found with high probability; and (b) all data items closer to the query than some threshold will be found with high probability, whereas data items that are significantly more distant will be found with low probability. Straightforward geometric intuition suggests that this approach will work best when the data items have zero mean, which is easy to arrange. Notice that using n k-bit hash tables is not the same as using one nk-bit hash table. In the first case, the list of points returned from a particular query is a union of the lists returned from each of the n hash tables. This means that points that are near the query but just happen to lie outside the query's cell for one hash table, have a good chance of being found in another hash table. In the second case, the list we must handle is much shorter (because there are more cells), but there is a better chance of missing nearby points. The choice of n and k will

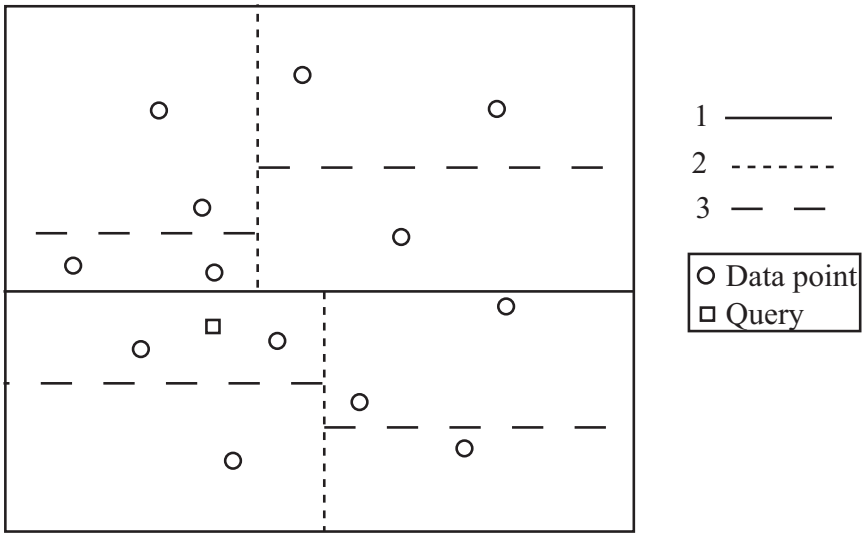

FIGURE 14.2: *A k-d tree is built by recursively splitting cells along dimensions. The order in which cells are split for this tree is shown by the dashes on the lines. The nearest neighbor for the query point is found by (a) finding the closest item in the query point's cell, then (b) backtracking, and looking at cells that could contain closer items. Notice that in this example one will need to go right up to the root of the tree and down the other side to find the nearest neighbor. In high dimensions, this backtracking becomes intractable, but if an approximate nearest neighbor is sufficient, the amount of backtracking can be controlled successfully.*

depend on dimension and on the nature of the probabilistic guarantee one wants. There are a variety of other possible choices of hash function. Details of other choices, and precise statements of the relevant guarantees, can be found in (**?**).

Random projection methods build a cell structure that is independent of the distribution of the data. This means trouble if data is heavily concentrated in some regions, because queries that land in a heavily populated cell of the hash table will need to search a long list. An alternative method is to use a **k-d tree** to build the cell structure. A k-d tree is built by recursively splitting cells. The root will be the whole space. To generate the children of a cell, select one dimension d, perhaps at random, and select some threshold value t_d. Write the dth component of $\mathbf{v}$ as v_d. Now all data items in a cell with $v_d \leq t_d$ are placed in the left child, and all others in the right. We now apply this splitting procedure recursively to the root, until the children are sufficiently small. If we choose the threshold value appropriately (for example, the median of the data in the cell), we can ensure that cells are small in dense components of the space and large in sparse components.

The nearest neighbor to a query can then be found by walking the tree to find the cell containing the query point. We then check any data items in that cell. Write the distance from the query to the closest as d_c. We now backtrack, investigating cells that could contain points closer than d_c and updating d_c when

we find a better point. We can prune any branch of the tree that represents a volume that is further from the query than d_c. This procedure works well for low dimensions, but becomes unattractive in high dimensions because we will need to explore too many cells (the number of neighbors of a cell goes up exponentially with dimension).

This difficulty can be avoided if an approximate nearest neighbor is sufficient. In the **best bin first** approach, we look at a fixed number N_c of cells, then report the best point found so far. Promising cells will tend to have some points that are close to the query, and we define the distance between a cell and the query to be the shortest distance from the query to any point on the cell's boundary. Whenever we investigate the child of a cell, we insert the other child into a priority queue, ordered by distance to the query. Once we have checked a cell, we retrieve the next cell from the priority queue. We do this until we have looked at N_c cells. We will look mainly at cells that are close to the query, and so the point we report is a good approximate nearest neighbor.

Good performance of a particular method depends somewhat on the dataset. For most applications, the choice of method can be made offline using the dataset, or a subset of it. **?** describe a software package that can choose an approximate nearest neighbors method that is fastest for a particular dataset. Generally, they find that using multiple randomized k-d trees is usually the best; at the time of writing, software could be found at http://www.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN.

14.4 ENTROPY AND INFORMATION GAIN

It turns out to be straightforward to keep track of information, in simple cases. We will start with an example. Assume I have 4 classes. There are 8 examples in class 1, 4 in class 2, 2 in class 3, and 2 in class 4. How much information *on average* will you need to send me to tell me the class of a given example? Clearly, this depends on how you communicate the information. You could send me the complete works of Edward Gibbon to communicate class 1; the Encyclopaedia for class 2; and so on. But this would be redundant. The question is how little can you send me. Keeping track of the amount of information is easier if we encode it with bits (i.e. you can send me sequences of '0's and '1's).

Imagine the following scheme. If an example is in class 1, you send me a '1'. If it is in class 2, you send me '01'; if it is in class 3, you send me '001'; and in class 4, you send me '101'. Then the expected number of bits you will send me is

$$p(\text{class} = 1)1 + p(2)2 + p(3)3 + p(4)3 = \frac{1}{2}1 + \frac{1}{4}2 + \frac{1}{8}3 + \frac{1}{8}3$$

which is 1.75 bits. This number doesn't have to be an integer, because it's an expectation.

Notice that for the i'th class, you have sent me $-\log_2 p(i)$ bits. We can write the expected number of bits you need to send me as

$$-\sum_i p(i) \log_2 p(i).$$

This expression handles other simple cases correctly, too. You should notice that it isn't really important *how many* objects appear in each class. Instead, the *fraction* of all examples that appear in the class is what matters. This fraction is the prior probability that an item will belong to the class. You should try what happens if you have two classes, with an even number of examples in each; 256 classes, with an even number of examples in each; and 5 classes, with $p(1) = 1/2$, $p(2) = 1/4$, $p(3) = 1/8$, $p(4) = 1/16$ and $p(5) = 1/16$. If you try other examples, you may find it hard to construct a scheme where you can send as few bits *on average* as this expression predicts. It turns out that, in general, the smallest number of bits you will need to send me is given by the expression

$$-\sum_i p(i) \log_2 p(i)$$

under all conditions, though it may be hard or impossible to determine what representation is required to achieve this number.

The **entropy** of a probability distribution is a number that scores how many bits, on average, would need to be known to identify an item sampled from that probability distribution. For a discrete probability distribution, the entropy is computed as

$$-\sum_i p(i) \log_2 p(i)$$

where i ranges over all the numbers where $p(i)$ is not zero. For example, if we have two classes and $p(1) = 0.99$, then the entropy is 0.0808, meaning you need very little information to tell which class an object belongs to. This makes sense, because there is a very high probability it belongs to class 1; you need very little information to tell you when it is in class 2. If you are worried by the prospect of having to send 0.0808 bits, remember this is an average, so you can interpret the number as meaning that, if you want to tell which class each of 10^4 independent objects belong to, you could do so in principle with only 808 bits.

Generally, the entropy is larger if the class of an item is more uncertain. Imagine we have two classes and $p(1) = 0.5$, then the entropy is 1, and this is the largest possible value for a probability distribution on two classes. You can always tell which of two classes an object belongs to with just one bit (though you might be able to tell with even less than one bit).