Classification:
- Input: Features, output: one bit
- (more complicated models later)

Example:
- Input: height
- Output: gender

- For the moment, assume that $\sigma^\phi$ are evenly distributed.
- How would one classify?
  - Choose a height threshold
  - Mistakes are inevitable
  - We need to choose the least expensive.
Notice guaranteed error

we will get these $\theta$'s wrong.

we will get these $\sigma$'s wrong.
Strategy

\( h > t \uparrow \sigma \)

otherwise \( \varnothing \)

Cases:

- males, females equally common \( t \at x \)
- males very common, \( \sigma \)'s rare \( t \ll x \)
- \( \sigma \) rare, \( \varnothing \) common \( t \gg x \)

Q: reasonable way to set \( t \)

Choose \( t \) to produce the minimum expected cost of errors.

- two types of error
  
  \( (\sigma \rightarrow \varnothing) \)
  
  \( (\varnothing \rightarrow \sigma) \)

We assume a reward for right answer is 0.
we have a feature \( x \).

If we say \( \phi \), we get \( h(\phi \rightarrow \phi) \) wrong.

If we do this many times, we get \( \phi \) with frequency \( P(\phi | x) \).

and \( h(\phi \rightarrow \phi) \) with freq \( P(\phi | x) \).

So expected loss of \( \phi \) is \( 0 \cdot P(\phi | x) + h(\phi \rightarrow \phi) \cdot P(\phi | x) \).

Similarly, expected loss of \( \phi \) is \( 0 \cdot P(\phi | x) + h(\phi \rightarrow \phi) \cdot P(\phi | x) \).
so in principle we have a rule

\[
\text{cost of } \varphi \text{ is: } L(\varphi \rightarrow \varphi) p(\varphi | x)
\]

at \(x\),

\[
\text{cost of } \sigma^1 \text{ is: } L(\varphi \rightarrow \sigma^1) p(\sigma^1 | x)
\]

choose the least expensive, say that.

But where do we get \(p(\sigma^1 | x), p(\varphi | x)\)?

1) \(p(\varphi | x) = 1 - p(\sigma^1 | x)\)

(only 2 options)

2) \(p(\sigma^1 | x) = \frac{p(x | \sigma^1) p(\sigma^1)}{p(x)}\)

\[
= \frac{p(x | \sigma^1) p(\sigma^1)}{\left[ p(x | \sigma^1) p(\sigma^1) + p(x | \varphi) p(\varphi) \right]}
\]

We could read this off the histograms.
We can now build one useful form of classifier.

- measure \( p(x | \phi) \), \( p(\phi) = \pi \),
  
  (say, histogram)

- Say

\[ \phi \leftrightarrow o \] if \( L(\phi \rightarrow o) \cdot p(\phi | x) > L(o \rightarrow \phi) \cdot p(o | x) \).

\( \phi \leftrightarrow o \)
  
  doesn't matter
  
  now, consider

\[ L(\phi \rightarrow o) \cdot p(\phi | x) = L(o \rightarrow \phi) \cdot p(o | x) \]

all that matters is

\[ R = \frac{L(\phi \rightarrow o)}{L(o \rightarrow \phi)} \]
So we care about

$$R \cdot p(\varphi | x) = p(\sigma^\top | x)$$

i.e.

$$R \cdot \frac{p(x | \varphi)}{p(x | \sigma^\top)} \frac{p(\sigma^\top)}{p(x)} = \frac{p(x | \sigma^\top)}{p(x)} \frac{(1-\Pi)}{\Pi} \cdot \frac{1}{R}$$

i.e.

$$\frac{p(x | \varphi)}{p(x | \sigma^\top)} = \frac{(1-\Pi)}{\Pi} \cdot \frac{1}{R} \Rightarrow \text{likelihood ratio}$$

i.e. if

$$\frac{p(x | \varphi)}{p(x | \sigma^\top)} > q(\Pi, R) \Rightarrow \varphi$$

doesn't matter say \varphi

say \sigma
Now equations give a way to determine \( Q \) but we can manage without.

**Rule:**

\[
p(x|q) > t, \text{ say } q
\]

\[
\frac{p(x|q)}{p(x|0^q)}
\]

Plot errors \( v \) varying \( t \)

Receiver operating curve.

---

Plot for different \( t \).
Model: We are trying to detect $0^\dagger$'s in a population of $1^\dagger$'s (or vice versa).

$$P\text{ (false det.)} = \frac{\# \text{ of } 1^\dagger\text{'s we called } 0^\dagger\text{'s}}{\# \text{ of times we classified}}$$

$$P\text{ (true det.)} = \frac{\# \text{ of } 0^\dagger\text{'s we called } 0^\dagger\text{'s}}{\# \text{ of } 0^\dagger\text{'s in population}}$$

We evaluate this on test data.

Training:

- Form histograms $p(x|\theta)$ etc.
- Can be hard to do with high dimensional $x$. 
Major Problem:
- a 1-D histogram w/ n cells in each dir has n^2 cells
- 2D
- 3D
- dD

We cannot build such histograms:

Strategies:
- Simplify the model
- Search for decision boundary directly
- Search for decision boundary directly
Simplify model

model

\[ p(x_1, x_2, \ldots, x_n \mid \theta) = p(x_1 \mid \theta) p(x_2 \mid \theta) \ldots p(x_n \mid \theta) \]

This model is usually wrong. But it's convenient and works surprisingly well.

Naive Bayes:

Method: one histogram each in each direction.

- form likelihood ratio
- test against threshold.
B) model $p(q|x)$ directly.
- parametric models, perhaps later

C) Find decision boundary:
- by continuity reasoning (Nearest neighbours)
- by search

Nearest neighbours:

alg:
- find $x_i \in$ examples such that
  \[ \|x_i - x\| \text{ is smallest} \]
- class of $x$ is class of $x_i$
**k-Nearest Neighbors:**

**Algorithm:**
- Find the $k$ examples that are closest to $x$.
- Find the most common class in these neighbors.
- If there are $k$ in this class, classify $x$ with that class; otherwise, don't know.

**Properties:**
- With enough examples, error rate is no more than $2^x$ best possible.
- We must worry about scaling dimension.
- Algorithmically complex.
model \( P(\theta|x) \) directly

(we'll talk about this later)

Find decision boundary directly

- recall

- eg in 2D

- Hard to search for a curve in 2D or more-D
Easy, highly successful strategy.

The decision boundary is a flat hyperplane:

\[
\begin{align*}
(w^T x + b) > 0 & \text{ class 1} \\
= 0 & \\
< 0 & \text{ class 2}
\end{align*}
\]
clearly, this won't work always

\[ (x, y) \rightarrow (x^2, y^2) \]
again

\[ (x, y) \rightarrow (x^2, xy, y^2, x, y) \]

But (recall general ellipse is)

\[ ax^2 + bxy + cy^2 + dx + ey + f = 0 \]

and we get linear boundary.

**General principle here:**

With enough features a linear classifier will behave well.
How to choose a linear classifier

Q: What is $w, b$?

A: Minimize loss of using classifier.

$x$-side

$T$-side

$w^T x + b = 0$

might become bad with small change.
Good choice, closest examples should be as far away as possible.

We need to deal with easy cases.
\[ \text{dist} = \frac{|w^T x + b|}{||w||} \]

- hence, if loss is zero, we would like to minimize \( ||w||^2 \)
- if loss is non-zero, small \( ||w||^2 \) is a good idea
- Hence minimize

\[ \text{loss} + \theta ||w||^2 \]

↑ weighting parameter choose later
Logistic loss:

\[ \text{log} \left[ \frac{1}{1 + \exp(-y \cdot \hat{y})} \right] \]

\[ y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \]

\[ \text{log} \left[ \frac{1}{1 + \exp(-y \cdot \hat{y})} \right] \]

leads to

\[ w \approx \text{argmin} \sum_i \left[ y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i) \right] \]
loss:
- Consider an example of class 1
- We want $w^T x + b > 0$

\[ w^T x + b \]

- For $w^T x + b > 0$, loss = 0
- For $w^T x + b < 0$, loss is big
- For $w^T x + b \leq 0$, bigger

- Loss should not grow too fast, otherwise one example dominates
- Loss should be non-zero for small $w^T x + b$
Hinge loss:

- Example has label \( y_i \in \{1, -1\} \)
- We predict \( y_i^p = \mathbf{w}^T \mathbf{x}_i + b \)
- Loss is \( \max \{0, 1 - y_i \cdot y_i^p\} \)
- Plotted above for \( y_i = 1 \)
- Leads to
  \[
  \min_{\mathbf{w}, b} \|\mathbf{w}\|^2 + \sum_i \max\{0, 1 - y_i \cdot y_i^p\}
  \]
  which is hard.