Book: Hastie, Tibshirani + Friedman
- Material from Jordan's course notes
- Papers, etc. + Semi-supervised

Themes:
- Classification
- Regression
- Inference

Regression:
- Predict some value $y$ from vars $x$

  - Choose some loss function $l(y, f(x))$

  and obtain $f$ by minimizing

  $E[ l(y, f(x)) ]$ with respect to $f$
Important case: square error loss

\[ \text{loss}(y, f(x)) = (y - f(x))^2 \]

hence

\[ \min \mathbb{E}( (y - f(x))^2 ) \Rightarrow \text{EPE} = \text{expected prediction error} \]

\[ = \mathbb{E}_x \left[ \mathbb{E}_{y|x} \left[ (y - f(x))^2 \right] \right] \]

and so we can minimize

\[ \mathbb{E}_{y|x} \left[ (y - f(x))^2 \right] \]

\[ = \int (y - f(x))^2 \ p(y|x) \ dy \]

at each \( x \).

but the solution is

\[ f(x) = \mathbb{E}(y | x = x) \]

\[ = \int y \ p(y | x = x) \ dy \]
Example cases: linear regression

\[ f(x) = \mathbf{x}^T \beta \]

we usually ensure that one column of \( \mathbf{x} \) is a 1 to simplify notation

\[
\begin{align*}
\text{EPE} &= E \left[ (Y - \mathbf{x} \beta)^2 \right] \\
&= E[\mathbf{y}^2] - 2E[\mathbf{x} \mathbf{y}] \beta + \beta^T E[\mathbf{x} \mathbf{x}^T] \beta
\end{align*}
\]

so

\[
\beta = (\mathbf{x}^T \mathbf{x})^{-1} \mathbf{x}^T \mathbf{y}
\]

Alternatively:

What we're accustomed to is

\[
\begin{align*}
\mathbf{Y} &= [Y_1, \ldots, Y_n]^T \\
\mathbf{X} &= [x_1, \ldots, x_n]^T
\end{align*}
\]

\[
\min_\beta \| \mathbf{Y} - \mathbf{X} \beta \|^2
\]

\[
\beta = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}
\]

This replaces \( E_{\mathbf{x}_i \mathbf{y}} \) with \( \Sigma \) data
$f(x) = \frac{1}{k} \sum_{i \in k \text{ examples closest to } x} y_i$

Now this approximates $E(Y | x = x)$ by assuming that $x$ changes "slowly" and then replacing an expectation with a sum.

Notice:

For most $P(x, y)$ as $N, k \to \infty$ and $k \to 0$

$$f(x) \to \frac{1}{N} E(Y | x = x).$$
How does this relate to classification?

- Regression where y is a categorical var

- for now, \( y \in \{ 0, 1 \} \)

**Zero-One loss**

\[
L(y, f(x)) = \begin{cases} 
1 & \text{don't agree} \\
0 & \text{otherwise}
\end{cases}
\]

\[
E_{PE} = \mathbb{E}_x \left[ E_{y|x} \left[ L(c, y, f(x)) \right] \right]
\]

\[
= \mathbb{E}_x \left[ L(1, f(x)) P(1|x=x) + L(0, f(x)) P(0|x=x) \right]
\]

\[\rightarrow\] we can minimize this by choosing the class such that

\[
f(x) = \text{arg min}_y \mathbb{E}_x \left[ L(1, y, f(x)) P(1|x=x) + L(0, y) P(0|x=x) \right]
\]
for 0-1 loss, we get

\[ \begin{cases} 
1 & \text{if } P(1/x) < P(0/x) \\
0 & \text{(irrelevant)} 
\end{cases} \]

We could do this with K-NN easily

- but there are problems in high dimensions

The curse of dimension

1) where is volume in H.D.?

- on the skin

2) what % of data is nearby?

- very little

Eg ->
we want to use uniform data.
- $r$ is the fraction of data.
- $r = \frac{1}{V}$, where $V$ is the volume of the cube.
- to capture 10% of data, we need edge length of 0.8
- hardly local

Bias and Variance:

- consider $y = f(x) = e^{-11x^2}$

(Deterministic $f(x)$)

- we will draw a bunch of $(x, y)$ samples $T$
  and approximate $f$ using $1$-NN
- All error is due to choice of samples
- consider $\hat{y}_o = 1$-NN est of $y(x)$
\[ \text{MSE} (\theta_0) = E_T \left[ (f(x_0) - y_o)^2 \right] \]

\[ = f(x_0)^2 - 2 E_T [y_0] f(x_0) + E_T [y_0^2] \]

\[ = f(x_0)^2 - 2 E_T [y_0] f(x_0) + \left( E_T [y_0^2] \right)^2 \]

\[ + \left( E_T [y_0] \right)^2 - 2 \left( E_T [y_0] \right) + E_T [y_0^2] \]

\[ = \left( f(x_0) - E_T [y_0] \right)^2 \]

\[ \vdots \]

\[ + E_T \left[ (y_0 - E_T [y_0])^2 \right] \]

This sort of decomposition is universal.

In our example, as dimension goes up,

\[ E_T [y_0] \]

goes down fast (why?)

so bias goes up fast.
linear in Stabilizing linear regression

- wish to reduce prediction error
  - by shrinking variables

- disadvantages:
  - variance could go up

- penalizing large coeffs
  (in the hope they shrink to zero)

\[
\text{ridge } L(b) = \| Y - \beta_0 I - X \beta \|^2 + \lambda \beta' \beta
\]

- we don't want to penalize the constant because that would result in fits that aren't covariant under translation in Y — doesn't make sense.

- for this discussion, center Y so

\[
(\text{ie. } \sum_{i} y_i = 0) \implies \beta_0 = 0
\]
Then
\[(y - X\beta)'(y - X\beta) + \lambda \beta'\beta\]

or
\[
\text{ridge}\quad \beta = \left(X'X + \lambda I\right)^{-1}X'y.
\]

Deals with possible rank problems.

Notice that
\[
\min \quad (y - X\beta)'(y - X\beta) + \lambda \beta'\beta
\]

\[\lambda > 0\]

\[\text{II} \]
\[
\min \quad (y - X\beta)'(y - X\beta)
\]

\[s + \beta'\beta \leq s\]
Notice that:
- ridge regression isn't covar under scaling of inputs (why?)
- qualitative arg:
  - $\beta$ can be poorly determined
    for presence of correlated vars

\[ \text{Ridge regression + SVD} \rightarrow \text{diagonal.} \]
write \[ X = U D V^T \]
\[ \begin{bmatrix} X^T X \end{bmatrix} \begin{bmatrix} \beta \end{bmatrix} = \begin{bmatrix} X^T \end{bmatrix} \begin{bmatrix} y \end{bmatrix} \]
\[ \begin{bmatrix} X \beta \end{bmatrix} = X \left( X^T X \right)^{-1} X^T y \]
\[ = U U^T y \]
\[ \begin{array}{l}
\Rightarrow \text{these are coefs of } y \text{ in } U \text{ basis.}
\end{array} \]
\[ \hat{X}_\beta = X(X^TX + \lambda I)^{-1}X^Ty \]

\[ = UD(D^2 + \lambda I)^{-1}DU^Ty \]

\[ = \sum \frac{u_j}{d_j^2 + \lambda} \left[ \frac{d_j^2}{d_j^2 + \lambda} \right] u_j^Ty \]

- we shrink coefs by \( \frac{d_j^2}{d_j^2 + \lambda} \)

- shrinkage is most pronounced for

\[ d_j^2 \text{ is small} \]

\[ \Rightarrow \text{components of } x \text{ with low variance.} \]

\[ \Rightarrow \text{poor estimates of gradient of } y \text{ in this linear.} \]
Lasso:

\[
\text{Lasso} \quad \underset{\beta}{\text{arg min}} \quad \sum_{i=1}^{N} \left( y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2
\]

\text{st} \quad \sum_{j} |\beta_j| \leq t

...can no longer use linear alg; this is a quadratic programming problem

...sufficiently small \( t \) forces some

\[ \beta_j = 0 \]