Chapter 3

DJM

2 February 2017

Let's review some stuff

REVIEW

Statistical models

We observe data $Z_1, Z_2, \ldots, Z_n$ generated by some probability distribution P. We want to use the data to learn about P.

A **statistical model** is a set of distributions $\mathbb{P}$.

Some examples:

1. $\mathbb{P} = \{0 < p < 1 : P(z = 1) = p,\ P(z = 0) = 1 - p\}$.
2. $\mathbb{P} = \{\beta \in \mathbb{R}^p, \sigma > 0 : Y \sim N(X^\top \beta, \sigma^2),\ X \text{ fixed}\}$.
3. $\mathbb{P} = \{\text{all CDF's } F\}$.
4. $\mathbb{P} = \{\text{all smooth functions } f : \mathbb{R}^p \to \mathbb{R}\}$

Statistical models 2

We observe data $Z_1, Z_2, \ldots, Z_n$ generated by some probability distribution P. We want to use the data to learn about P.

$$\mathbb{P} = \{P(z = 1) = p,\ P(z = 0) = 1 - p,\ 0 < p < 1\}$$

- To completely characterize P, I just need to estimate p.
- Need to assume that $P \in \mathbb{P}$.
- This assumption is mostly empty: **need independent, can't see** $z = 12$**.**

Statistical models 3

We observe data $Z_i = (Y_i, X_i)$ generated by some probability distribution P. We want to use the data to learn about P.

$$\mathbb{P} = \{\beta \in \mathbb{R}^p, \sigma > 0 : Y \sim N(X^\top \beta, \sigma^2),\ X \text{ fixed}\}.$$

- To completely characterize P, I just need to estimate β and σ.
- Need to assume that $P \in \mathbb{P}$.
- This time, I have to assume a lot more: **Linearity, independence, Gaussian noise, no ignored variables, no collinearity, etc.**

Convergence

Let $X_1, X_2, \ldots$ be a sequence of random variables, and let X be another random variable with distribution P. Let F_n be the cdf of X_n and let F be the cdf of X.

1. X_n converges **in probability** to X, $X_n \xrightarrow{P} X$, if for every $\epsilon > 0$,

$$\lim_{n\to\infty} P\left(|X_n - X| > \epsilon\right) = 0.$$

2. X_n converges **in distribution** to X, $X_n \xrightarrow{D} X$, if for all t,

$$\lim_{n\to\infty} F_n(t) = F(t)$$

Heuristics: the sequence is somehow getting "closer" to some limit. But things are random...

Convergence rules

Suppose $X_1, X_2, \ldots$ are independent random variables, each with mean μ and variance σ^2. Let $\bar{X}_n = \frac{1}{n}\sum_{i=1}^n X_i$.

1. **Weak law of large numbers**

$$\bar{X}_n \xrightarrow{P} \mu$$

The law of large numbers tell us that the probability mass of an average of random variables "piles up" near its expectation.

2. **Central limit theorem**

$$\frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma} \xrightarrow{D} N(0,1).$$

The CLT tells us about the shape of the "piling", when appropriately normalized.

Evaluation

Once I choose some way to "learn" a statistical model, I need to decide if I'm doing a good job.

How do I decide if I'm doing anything good?

Properties

Lots of ways to evaluate estimators, $\hat{\mu}$ of parameters μ.

(from last time)

- Consistency: $\hat{\mu} \xrightarrow{P} \mu$.
- Asymptotic Normality: $\hat{\mu} \xrightarrow{D} N(\mu, \Sigma)$
- Efficiency: how large is Σ
- Unbiased: $\mathbb{E}[\hat{\mu}] \stackrel{?}{=} \mu$
- etc.

None of these things make sense unless **your model is correct**.

Your model is wrong!

[unless you are flipping coins, gambling in a casino, or running randomized, controlled trials on cereal grains]

Mis-specified models

What happens when your model is wrong? And it **IS** wrong.

None of those evaluation criteria make any sense. The parameters no longer have any meaning.

[The criteria still hold in some sense: I can demand that I get close to the projection of the truth onto $\mathbb{P}$]

Prediction

Prediction is easier: your model may not actually represent the true state of nature, but it may still predict well.

> Over a 13-year period, [David Leinweber] found [that] annual **butter production** in Bangladesh "explained" 75% of the variation in the annual returns of the Standard & Poor's 500-stock index.
>
> By tossing in **U.S. cheese production** and the **total population of sheep** in both Bangladesh and the U.S., Mr. Leinweber was able to "predict" past U.S. stock returns with 99% accuracy.

This is why we don't use R^2 to measure prediction accuracy.

The setup

What do we mean by good predictions?

We make observations and then attempt to "predict" new, unobserved data.

Sometimes this is the same as estimating the mean.

Mostly, we observe $(y_1, x_1), \ldots, (y_n, x_n)$, and we want some way to predict Y from X.

Evaluating predictions

Of course, both Y and $\hat{Y}$ are **random**

I want to know how well I can predict **on average**

Let $\hat{f}$ be some way of making predictions $\hat{Y}$ of Y using covariates X

In fact, suppose I observe a dataset $\mathcal{D}_n = \{(Y_1, X_1,), \ldots, (Y_n, X_n)\}$.

Then I want to **choose** some $\hat{f}$ using $\mathcal{D}_n$.

Is $\hat{f}$ good on average?

Evaluating predictions

Choose some **loss function** that measures prediction quality: $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. We predict Y with $\hat{Y}$

Examples:

- **Squared-error:**

$$\ell(y, \hat{y}) = (y - \hat{y})^2$$

- **Absolute-error:**

$$\ell(y, \hat{y}) = |y - \hat{y}|$$

- **Zero-One:**

$$\ell(y, \hat{y}) = I(y \neq \hat{y}) = \begin{cases} 0 & y = \hat{y} \\ 1 & \text{else} \end{cases}$$

Can be generalized to Y in arbitrary spaces.

Prediction risk

Prediction risk

$$R_n(\hat{f}) = \mathbb{E}[\ell(Y, \hat{f}(X))]$$

where the expectation is taken over the new data point (Y, X) and $\mathcal{D}_n$ (everything that is random).

For **regression** applications, we will use squared-error loss:

$$R_n(\hat{f}) = \mathbb{E}[(Y - \hat{f}(X))^2]$$

For **classification** applications, we will use zero-one loss:

$$R_n(\hat{f}) = \mathbb{E}[I(Y \neq \hat{f}(X))]$$

Example 1: Estimating the mean

Suppose we know that we want to predict a quantity Y, where $\mathbb{E}[Y] = \mu \in \mathbb{R}$ and $\mathbb{V}[Y] = 1$.

That is, $Y \sim P \in \mathbb{P}$, where

$$\mathbb{P} = \{P : \mathbb{E}[Y] = \mu \text{ and } \mathbb{V}[Y] = 1\}.$$

Our data is $\mathcal{D}_n = \{Y_1, \ldots, Y_n\}$ such that $Y_i \overset{i.i.d.}{\sim} P$, and we want to estimate μ (and hence P).

Estimating the mean

- Let $\hat{Y} = \bar{Y}_n$ be the sample mean.
- We can ask about the **estimation risk** (since we're estimating μ):

$$\begin{aligned} R_n(\bar{Y}_n; \mu) &= \mathbb{E}[(\bar{Y}_n - \mu)^2] \\ &= \mathbb{E}[\bar{Y}_n^2] - 2\mu\mathbb{E}[\bar{Y}_n] + \mu^2 \\ &= \mu^2 + \frac{1}{n} - 2\mu^2 + \mu^2 \\ &= \frac{1}{n} \end{aligned}$$

Predicting new Y's

- Let $\hat{Y} = \bar{Y}_n$ be the sample mean.
- What is the **prediction risk** of $\bar{Y}$?

$$\begin{aligned} R_n(\bar{Y}_n) &= \mathbb{E}[(\bar{Y}_n - Y)^2] \\ &= \mathbb{E}[\bar{Y}_n^2] - 2\mathbb{E}[\bar{Y}_n Y] + \mathbb{E}[Y^2] \\ &= \mu^2 + \frac{1}{n} - 2\mu^2 + \mu^2 + 1 \\ &= 1 + \frac{1}{n} \end{aligned}$$

Predicting new Y's

- What is the prediction risk of guessing $Y = 0$?
- You can probably guess that this is a stupid idea.
- Let's show why it's stupid.

$$\begin{aligned} R_n(0) &= \mathbb{E}[(0 - Y)^2] \\ &= \mathbb{E}[(Y - \mu + \mu)^2] \\ &= \mathbb{E}[(Y - \mu)^2] + 2\mu\mathbb{E}[(Y - \mu)] + \mu^2 \\ &= 1 + \mu^2 \end{aligned}$$

Predicting new Y's

What is the prediction risk of guessing $Y = \mu$?

This is a great idea, but we don't know μ.

Let's see what happens anyway.

$$\begin{aligned} R_n(\mu) &= \mathbb{E}[(Y - \mu)^2] \\ &= 1 \end{aligned}$$

Estimating the mean

- Prediction risk: $R(\bar{Y}_n) = 1 + \frac{1}{n} = \sigma^2 + \frac{\sigma^2}{n}$
- Estimation risk: $R(\bar{Y}_n; \mu) = \frac{1}{n}$
- There is actually a nice interpretation here:
  1. The common $1/n$ term is $\mathbb{V}\left[\bar{Y}_n\right]$
  2. The extra factor of 1 in the prediction risk is **irreducible error**
     - Y is a random variable, and hence noisy.
     - We can never eliminate it's intrinsic variance.
     - In other words, even if we knew μ, we could never get closer than 1, on average.

- Intuitively, $\bar{Y}_n$ is the obvious thing to do.

Predicting new Y's

- Let's try one more: $\hat{Y}_a = a\bar{Y}_n$ for some $a \in (0, 1]$.

$$R_n(\hat{Y}_a) = \mathbb{E}[(\hat{Y}_a - Y)^2] = (1-a)^2\mu^2 + \frac{a^2}{n} + 1$$

- We can minimize this in a to get the best possible prediction risk for an estimator of the form $\hat{Y}_a$:

$$\arg\min_a R_n(\hat{Y}_a) = \left(\frac{\mu^2}{\mu^2 + 1/n}\right)$$

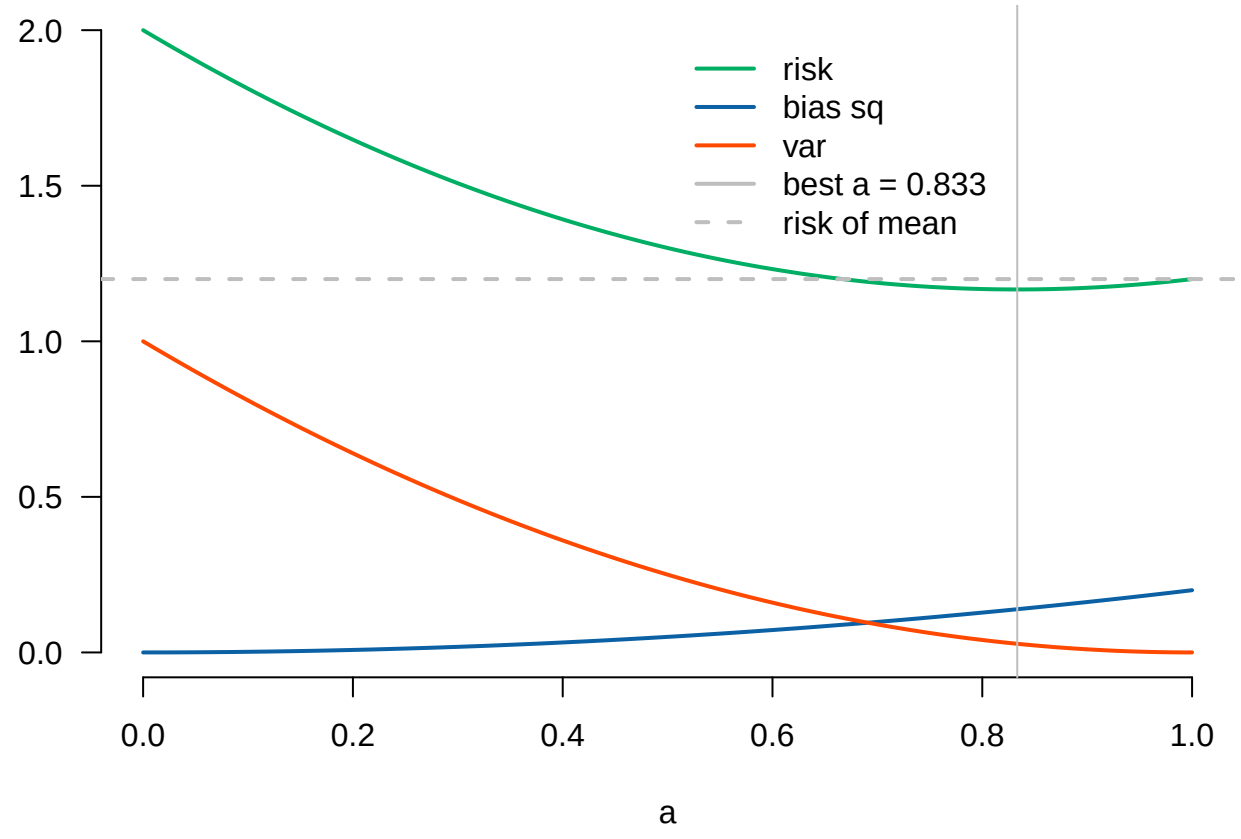
What happens if $\mu \ll 1$?

Wait a minute! You're saying there is a **better** estimator than $\bar{Y}_n$?

Bias-variance tradeoff: Estimating the mean

$$R(a) = R_n(\hat{Y}_a) = (a-1)^2\mu^2 + \frac{a^2}{n} + \sigma^2$$

```
mu=1; n=5; sig2=1
```



What?

Just to restate:

- If $\mu = 1$ and $n = 5$ then it is better to predict with 0.83 $\overline{Y}_n$ than with $\overline{Y}_n$ itself.
- In this case

1. $R(a) = R_1(a\overline{Y}_n) = 1.17$
2. $R(\overline{Y}_n) = 1.2$

Prediction risk

$$R_n(f) = \mathbb{E}[\ell(Y, f(X))]$$

Why care about $R_n(f)$?

- (+) Measures predictive accuracy on average.
- (+) How much confidence should you have in f's predictions.
- (+) Compare with other models.
- (-) **This is hard:**
  - Don't know P (if I knew the truth, this would be easy)

Risk for general models

We just saw that when you know the true model, and you have a nice estimator, the prediction risk has a nice decomposition

(this generalizes to much more complicated situations)

- Suppose we have a class of prediction functions $\mathcal{F}$,

$$\text{e.g. } \mathcal{F} = \left\{\beta : f(x) = x^\top \beta\right\}$$

- We use the data to choose some $\hat{f} \in \mathcal{F}$ and set $\hat{Y} = \hat{f}(X)$
- The **true** model is g (not necessarily in $\mathcal{F}$). Then:

$$R_n(\hat{f}) = \int \left[\text{bias}^2(\hat{f}(x)) + \text{var}(\hat{f}(x))\right] p(x)dx + \sigma^2$$

where $X \sim p$ and

$$\begin{aligned} \text{bias}(\hat{f}(x)) &= \mathbb{E}[\hat{f}(x)] - g(x) \\ \text{var}(\hat{f}(x)) &= \mathbb{E}[(\hat{f}(x) - \mathbb{E}\hat{f}(x))^2] \\ \sigma^2 &= \mathbb{E}[(Y - g(X))^2] \end{aligned}$$

Bias-variance decomposition

So,

1. prediction risk = bias^2 + variance + irreducible error
2. estimation risk = bias^2 + variance

What is $R(a)$ for our estimator $\hat{Y}_a = a\bar{Y}_n$?

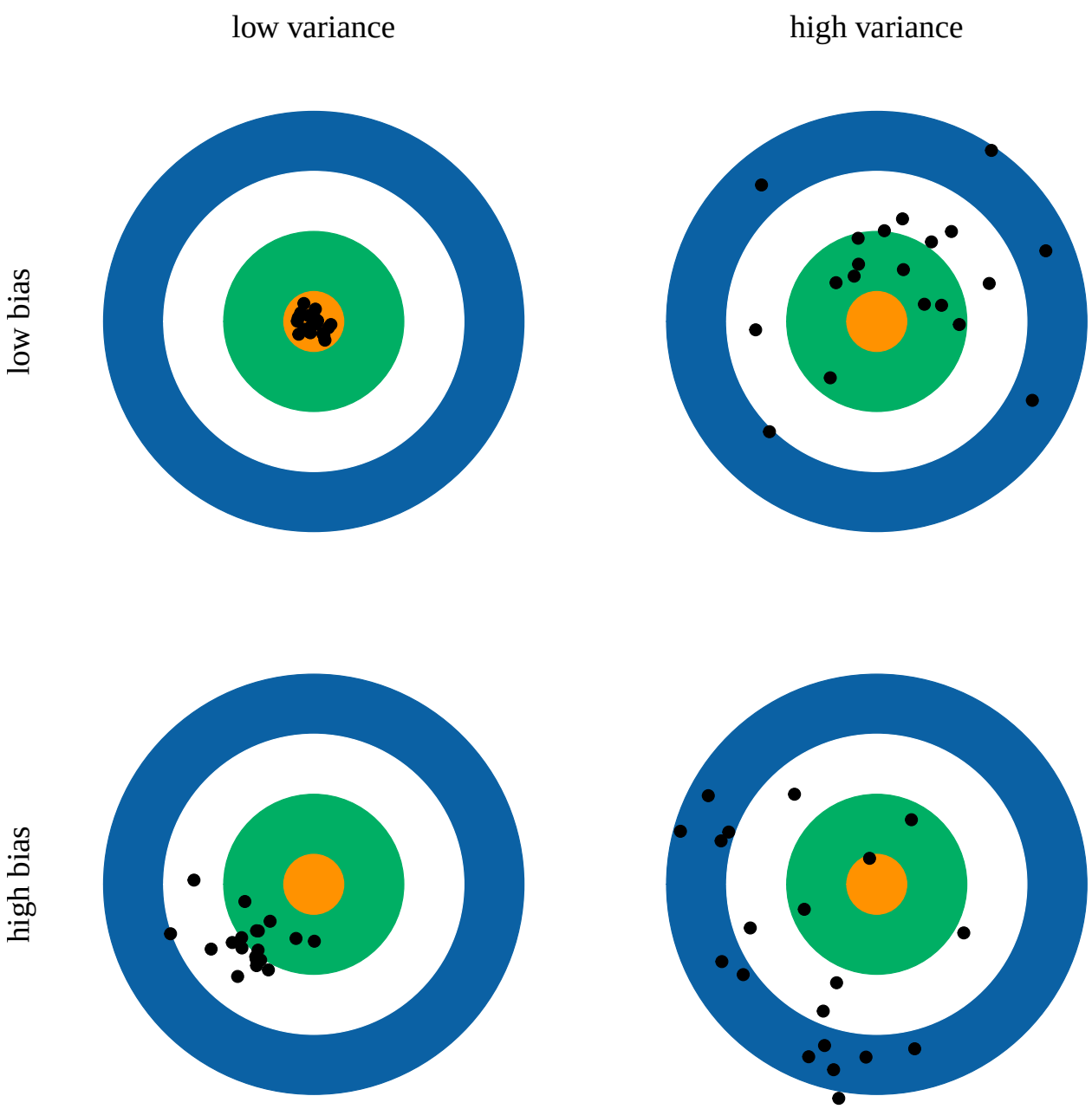
$$\begin{aligned}
\text{bias}(\hat{Y}_a) &= \mathbb{E}[a\bar{Y}_n] - \mu = (a-1)\mu \\
\text{var}(\hat{f}(x)) &= \mathbb{E}[(a\bar{Y}_n - \mathbb{E}[a\bar{Y}_n])^2] = a^2\mathbb{E}[(\bar{Y}_n - \mu)^2] = \frac{a^2}{n} \\
\sigma^2 &= \mathbb{E}[(Y-\mu)^2] = 1
\end{aligned}$$

$$\left(\text{That is: } R_n(\hat{Y}_a) = (a-1)^2\mu^2 + \frac{a^2}{n} + 1\right)$$

Bias-variance decomposition

Important implication: prediction risk is proportional to estimation risk. However, defining estimation risk requires stronger assumptions.

In order to make good predictions, we want our prediction risk to be small. This means that we want to "balance" the bias and variance.



Bias-variance tradeoff: Overview

- **bias:** how well does $\hat{f}$ approximate the truth g
- more complicated $\mathcal{F}$, lower bias. Flexibility $\Rightarrow$ Parsimony
- more flexibility $\Rightarrow$ larger variance
- complicated models are hard to estimate precisely for fixed n
- irreducible error

Example 2: Normal means

Suppose we observe the following data:

$$Y_i = \beta_i + \epsilon_i, \qquad i = 1, \ldots, n$$

where $\epsilon_i \overset{iid}{\sim} \mathrm{N}(0, 1)$.

We want to estimate

$$\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n).$$

The usual estimator (MLE) is

$$\hat{\boldsymbol{\beta}}^{MLE} = (Y_1, \ldots, Y_n).$$

This estimator has lots of nice properties: **consistent, unbiased, UMVUE, (asymptotic) normality...**

Normal means

But, the standard estimator **STINKS!** It's a bad estimator.

It has no bias, but big variance.

$$R_n(\hat{\boldsymbol{\beta}}^{MLE}) = \text{bias}^2 + \text{var} = 0 + n \cdot 1 = n$$

What if we use a biased estimator?

Consider the following estimator instead:

$$\hat{\beta}_i^S = \begin{cases} Y_i & i \in S \\ 0 & \text{else.} \end{cases}$$

Here $S \subseteq \{1, \ldots, n\}$.

Normal means

What is the risk of this estimator?

$$R_n(\hat{\boldsymbol{\beta}}^S) = \sum_{i \notin S} \beta_i^2 + |S|.$$

In other words, if some $|\beta_i| < 1$, then don't bother estimating them!

In general, introduced parameters like S will be called **tuning parameters**.

Of course we don't know which $|\beta_i| < 1$.

But we could try to estimate $R_n(\hat{\boldsymbol{\beta}}^S)$, and choose S to minimize our estimate.

Estimating the risk

By definition, for any estimator $\hat{\boldsymbol{\beta}}$,

$$R_n(\hat{\boldsymbol{\beta}}) = \mathbb{E}\left[\sum_{i=1}^{n}(\hat{\beta}_i - \beta_i)^2\right]$$

An intuitive estimator of R_n is

$$\hat{R}_n(\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{n}(\hat{\beta}_i - Y_i)^2.$$

This is known as the **training error** and it can be shown that

$$\hat{R}_n(\hat{\boldsymbol{\beta}}) \approx R_n(\hat{\boldsymbol{\beta}}).$$

Also,

$$\hat{\boldsymbol{\beta}}^{MLE} = \arg\min_{\beta} \hat{R}_n(\hat{\boldsymbol{\beta}}^{MLE}).$$

What could possibly go wrong?

Dangers of using the training error

Although

$$\hat{R}_n(\hat{\boldsymbol{\beta}}) \approx R_n(\hat{\boldsymbol{\beta}}),$$

this approximation can be very bad. In fact:

Training Error: $\hat{R}_n(\hat{\boldsymbol{\beta}}^{MLE}) = 0$

Risk: $R_n(\hat{\boldsymbol{\beta}}^{MLE}) = n$

In this case, the **optimism** of the training error is n.

Normal means

What about $\hat{\boldsymbol{\beta}}^S$?

$$\hat{R}_n(\hat{\boldsymbol{\beta}}^S) = \sum_{i=1}^{n}(\hat{\beta}_i - Y_i)^2 = \sum_{i \notin S} Y_i^2$$

Well

$$\mathbb{E}\left[\hat{R}_n(\hat{\boldsymbol{\beta}}^S)\right] = R_n(\hat{\boldsymbol{\beta}}^S) - 2|S| + n.$$

So I can choose S by minimizing $\hat{R}_n(\hat{\boldsymbol{\beta}}^S) + 2|S|$.

$$\text{Estimate of Risk} = \text{training error} + \text{penalty}.$$

The penalty term corrects for the optimism.

Model selection

- Broadly, Chapter 3 is about model selection
- Choosing S is model selection
- To do this, we need to estimate the Risk (MSE)
- CV can be used for this purpose
- The training error **cannot**
- Also, you should not use `anova` or the p-values from the `lm` output for this purpose.
- Why? These things are to determine whether those **parameters** are different from zero if you were to repeat the experiment many times, if the model were true, etc. etc.
- This is not the same as "are they useful for prediction = do they help me get smaller MSE"

What is Cross Validation

- Cross validation (CV, not coefficient of variation).
- This is another way or estimating the prediction risk.
- Why?

To recap:

$$R_n(\hat{f}) = \mathbb{E}[\ell(Y, \hat{f}(X))]$$

where the expectation is taken over the new data point (Y, X) and $\mathcal{D}_n$ (everything that is random).

We saw one estimator of R_n:

$$\hat{R}_n(\hat{f}) = \sum_{i=1}^{n} \ell(Y_i, \hat{f}(X_i)).$$

This is the training error. It is a **BAD** estimator because it is often optimistic.

Intuition for CV

- One reason that $\hat{R}_n(\hat{f})$ is bad is that we are using the same data to pick $\hat{f}$ **AND** to estimate R_n.
- Notice that R_n is an expected value over a **NEW** observation (Y, X).
- We don't have new data.

Wait a minute...

...or do we?

- What if we set aside one observation, say the first one (Y_1, X_1).
- We estimate $\hat{f}^{(1)}$ without using the first observation.
- Then we test our prediction:

$$\tilde{R}_1(\hat{f}^{(1)}) = \ell(Y_1, \hat{f}^{(1)}(X_1)).$$

- But that was only one data point (Y_1, X_1). Why stop there?
- Do the same with (Y_2, X_2)! Get an estimate $\hat{f}^{(2)}$ without using it, then

$$\tilde{R}_2(\hat{f}^{(2)}) = \ell(Y_2, \hat{f}^{(2)}(X_2)).$$

Keep going

- We can keep doing this until we try it for every data point.
- And then average them! (Averages are good)
- In the end we get

$$\text{LOO-CV} = \sum_{i=1}^{n} \tilde{R}_i(\hat{f}^{(i)}) = \sum_{i=1}^{n} \ell(Y_i - \hat{f}^{(i)}(X_i))$$

- This is leave-one-out cross validation

Problems with LOO-CV

1. Each held out set is small ($n = 1$). Therefore, the variance of my predictions is high.
2. Since each held out set is small, the training sets overlap. This is bad.
   - Usually, averaging reduces variance:

$$\mathbb{V}\left[\overline{X}\right] = \frac{1}{n^2}\sum_{i=1}^{n}\mathbb{V}\left[X_i\right] = \frac{1}{n}\mathbb{V}\left[X_1\right].$$

   - But only if the variables are independent. If not, then

$$\begin{aligned}\mathbb{V}\left[\overline{X}\right] &= \frac{1}{n^2}\mathbb{V}\left[\sum_{i=1}^{n} X_i\right] \\ &= \frac{1}{n}\mathbb{V}\left[X_1\right] + \frac{1}{n^2}\sum_{i\neq j}\text{Cov}\left[X_i,\ X_j\right].\end{aligned}$$

   - Since the training sets overlap a lot, that covariance can be pretty big.
3. We have to estimate this model n times.
   - There is an exception to this one. More on that in a minute.

K-fold CV

- To aleviate some of these problems, people usually use K-fold cross validation.
- The idea of K-fold is

1. Divide the data into K groups.
2. Leave a group out and estimate with the rest.
3. Test on the held-out group. Calculate an average risk over these $\sim n/K$ data.
4. Repeat for all K groups.
5. Average the average risks.

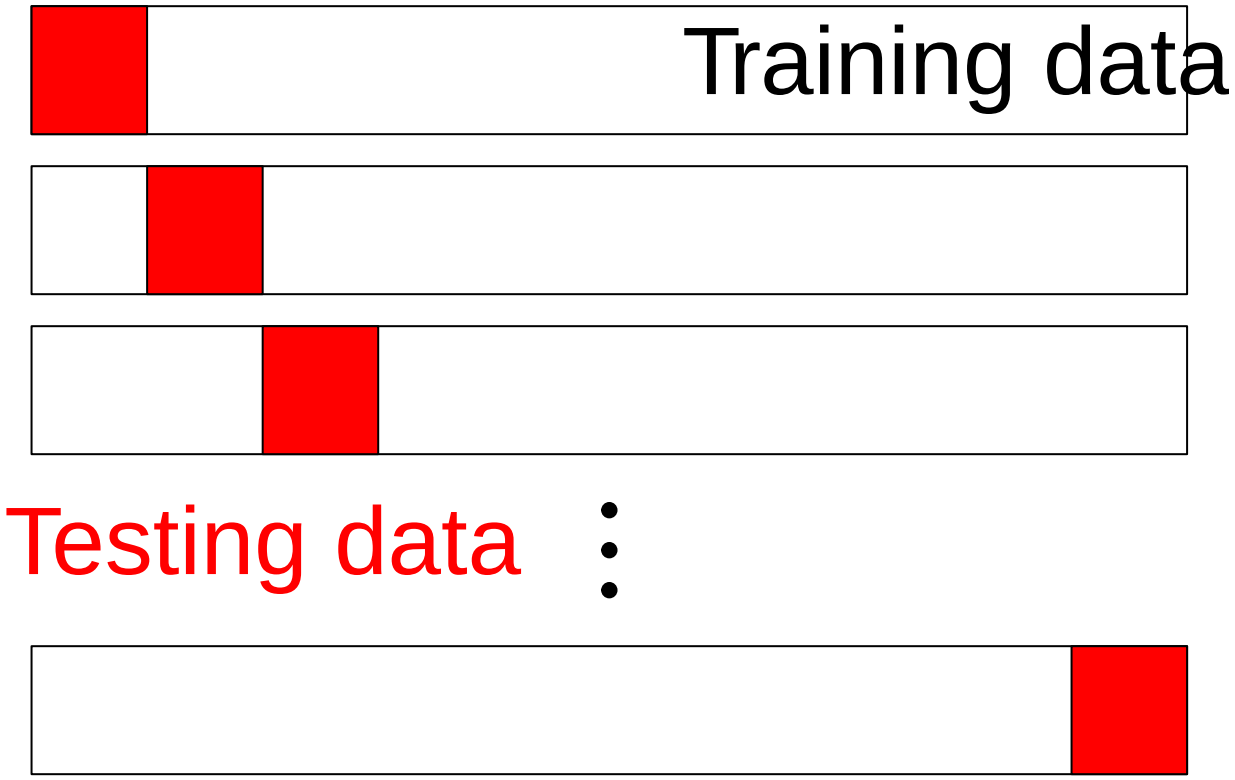
- Why is this better?

1. Less overlap, smaller covariance.
2. Larger hold-out sets, smaller variance.
3. Less computations (only need to estimate K times)

- Why might this be worse?

1. LOO-CV is (nearly) unbiased. The risk depends on how much data you use to estimate the model. LOO-CV uses almost the same amount of data.

A picture



CV code

```
cv.lm <- function(data, formulae, nfolds = 5) {
  data <- na.omit(data)
  formulae <- sapply(formulae, as.formula)
  responses <- sapply(formulae, response.name)
  names(responses) <- as.character(formulae)
  n <- nrow(data)
  fold.labels <- sample(rep(1:nfolds, length.out = n))
  mses <- matrix(NA, nrow = nfolds, ncol = length(formulae))
  colnames <- as.character(formulae)
  for (fold in 1:nfolds) {
    test.rows <- which(fold.labels == fold)
    train <- data[-test.rows, ]
    test <- data[test.rows, ]
    for (form in 1:length(formulae)) {
      current.model <- lm(formula = formulae[[form]], data = train)
      predictions <- predict(current.model, newdata = test)
      test.responses <- test[, responses[form]]
      test.errors <- test.responses - predictions
      mses[fold, form] <- mean(test.errors^2)
    }
  }
  return(colMeans(mses))
}
```

```
response.name <- function(formula) {
    var.names <- all.vars(formula)
    return(var.names[1])
}
```

Over-fitting vs. Under-fitting

Over-fitting means estimating a really complicated function when you don't have enough data.

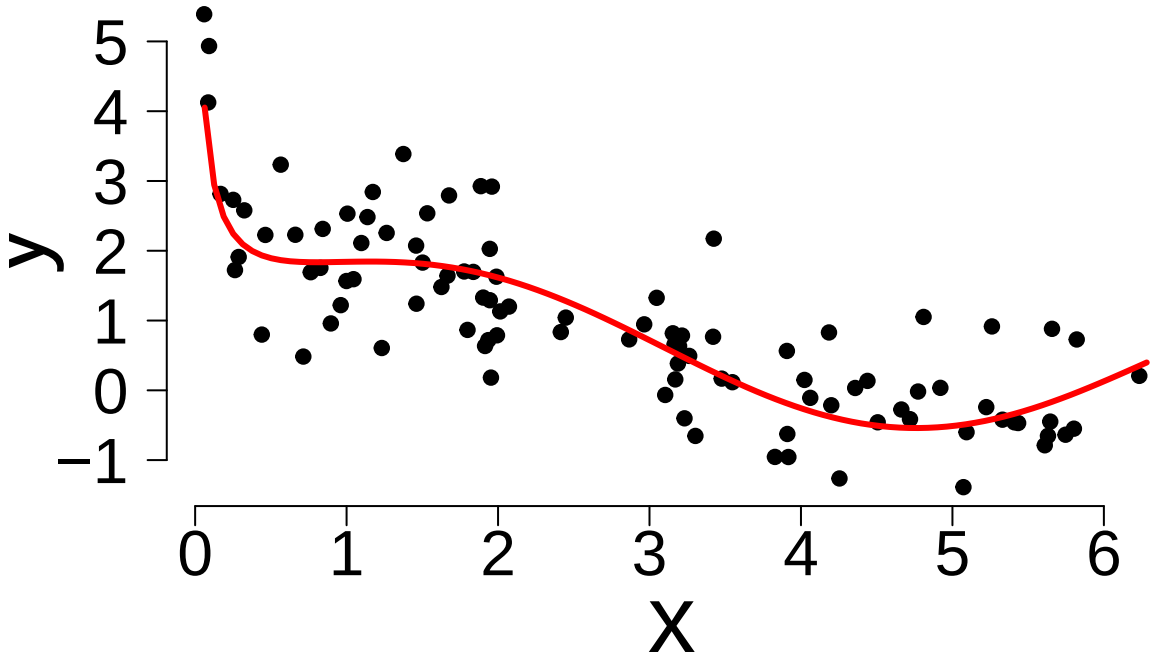
- This is likely a low-bias/high-variance situation.

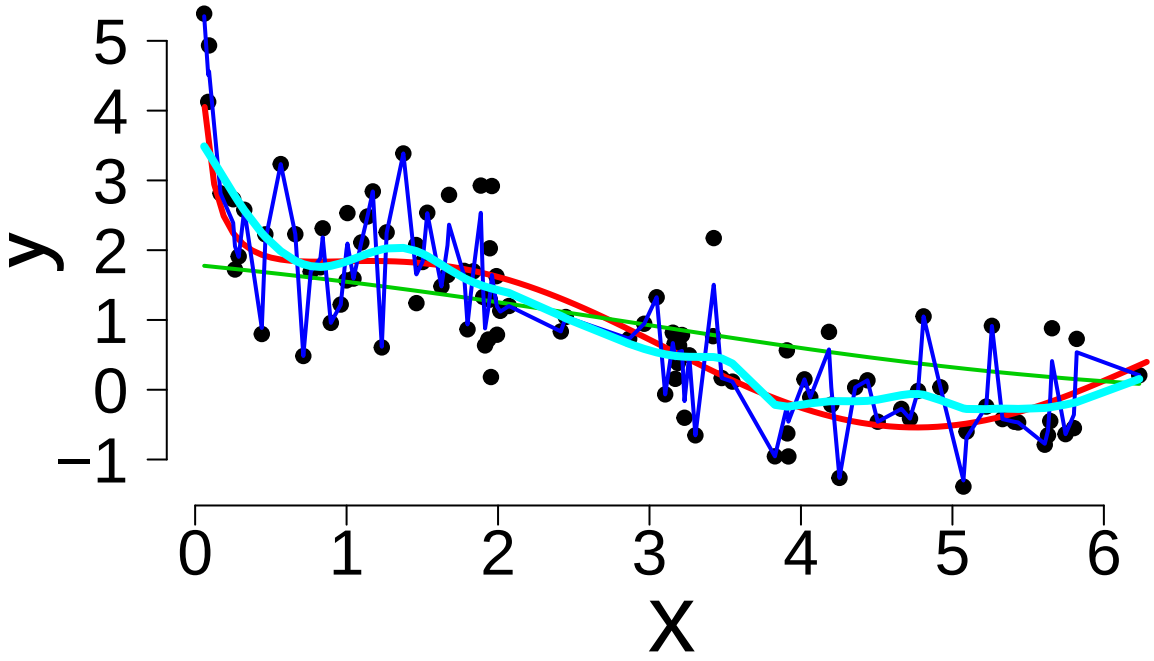
Under-fitting means estimating a really simple function when you have lots of data.

- This is likely a high-bias/low-variance situation.
- Both of these outcomes are bad (they have high risk).
- The best way to avoid them is to use a reasonable estimate of **prediction risk** to choose how complicated your model should be.

Example

```
trueFunction <- function(x) sin(x) + 1/sqrt(x)
set.seed(1234)
x = runif(100, 0, 2*pi)
y = trueFunction(x) + rnorm(100, 0, .75)
plot(x, y, pch=19, bty='n', las=1, cex.lab=3, cex.axis=2)
curve(trueFunction(x), 0, 2*pi, col=2, lwd=3, add=TRUE)
```






```
library(np)
fitAndPlot <- function(bw, col=4, lwd=2){
  fit = npreg(y~x, bws=bw)
  lines(x[order(x)], fitted(fit)[order(x)], col=col, lwd=lwd)
}
fitAndPlot(2, 'green') #under fit
fitAndPlot(.01, 'dark blue') #over fit
fitAndPlot(.2, 'cyan') #not too bad
```

More Questions about CV?