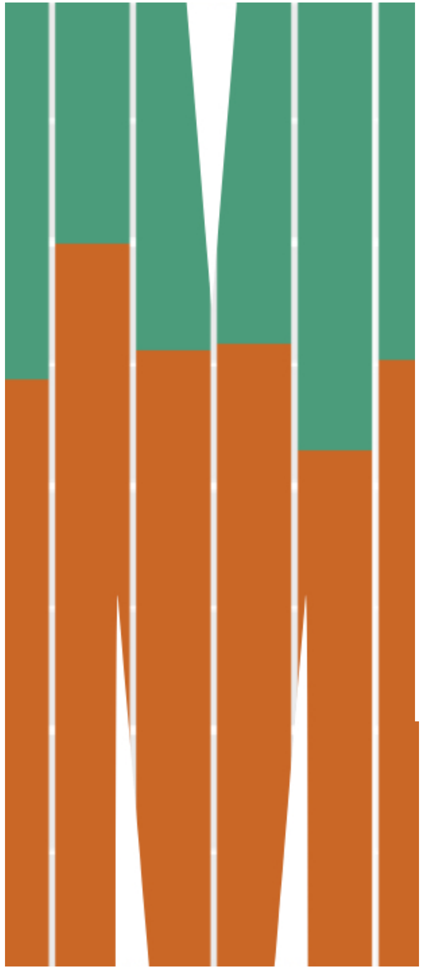


ETC3250: Resampling

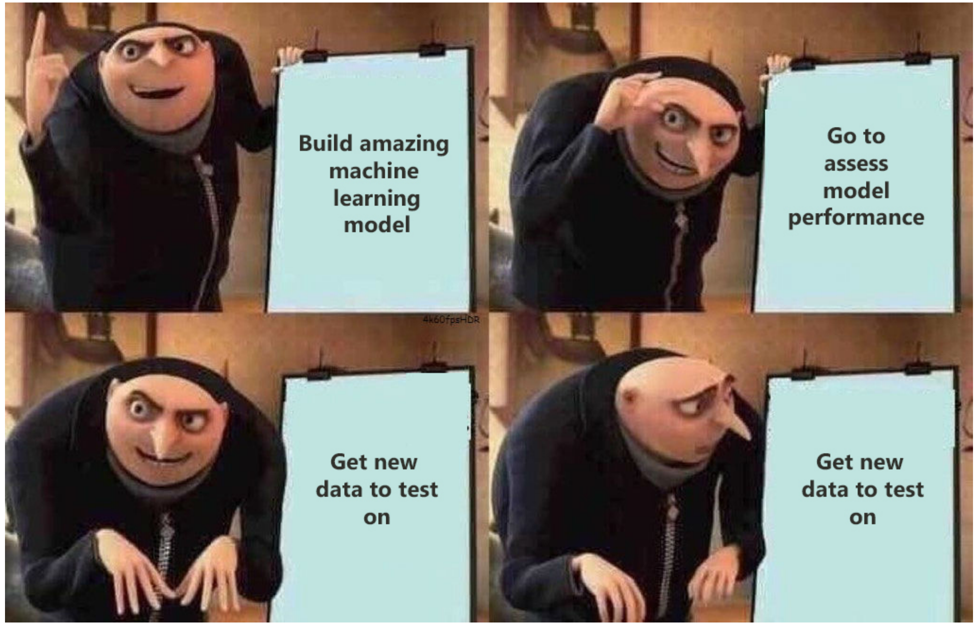
Semester 1, 2020

Professor Di Cook

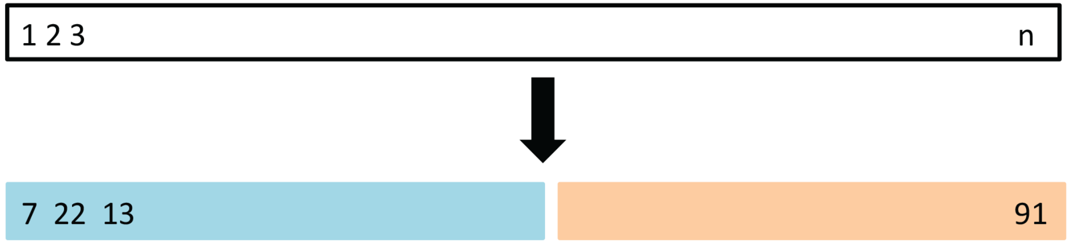
Econometrics and Business Statistics
Monash University

Week 3 (b)

Model assessment


Build amazing machine learning model
Go to assess model performance
Get new data to test on
Get new data to test on

Validating models



A set of n observations are randomly split into a training set (blue, containing observations 7, 22, 13, ...) and a validation set (yellow, all other observations not in training set).

Drawback: Only one split of data made, may not adequately estimate test error.

(Chapter5/5.1.pdf)

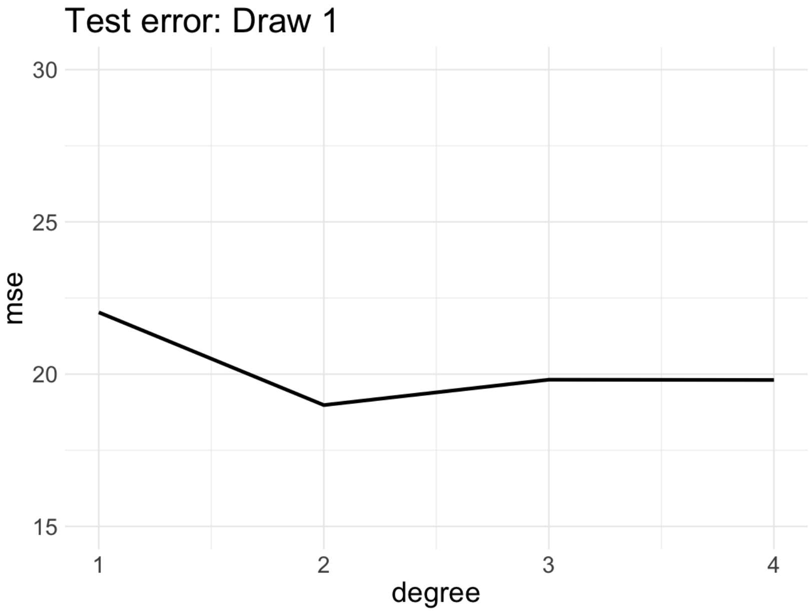
Validating models

Want to choose best degree of polynomial, for $\text{mpg} = \beta_0 + \beta_1 f(\text{horsepower}) + \varepsilon$

```
##  [1]  2  4  7  9 10 11 12 14 15 16 18 21 23
```

Remember:

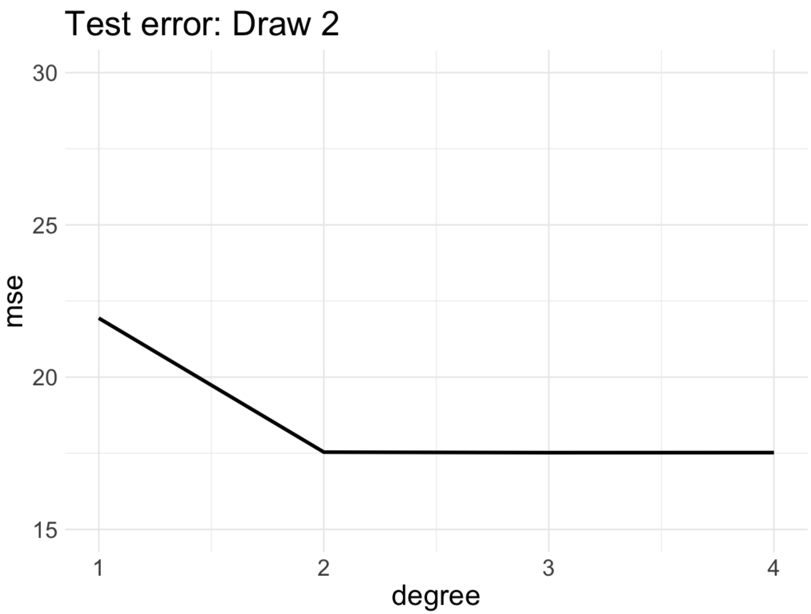
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$



Validating models

Want to choose best degree of polynomial, for $\text{mpg} = \beta_0 + \beta_1 f(\text{horsepower}) + \varepsilon$

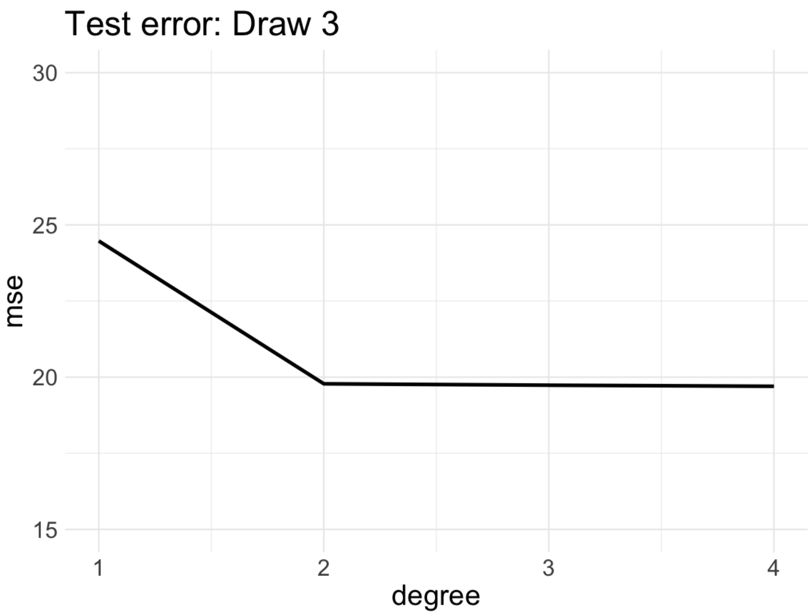
```
##  [1]  3  4  5  6  9 10 13 14 17 20 23 24 25
```



Validating models

Want to choose best degree of polynomial, for $\text{mpg} = \beta_0 + \beta_1 f(\text{horsepower}) + \varepsilon$

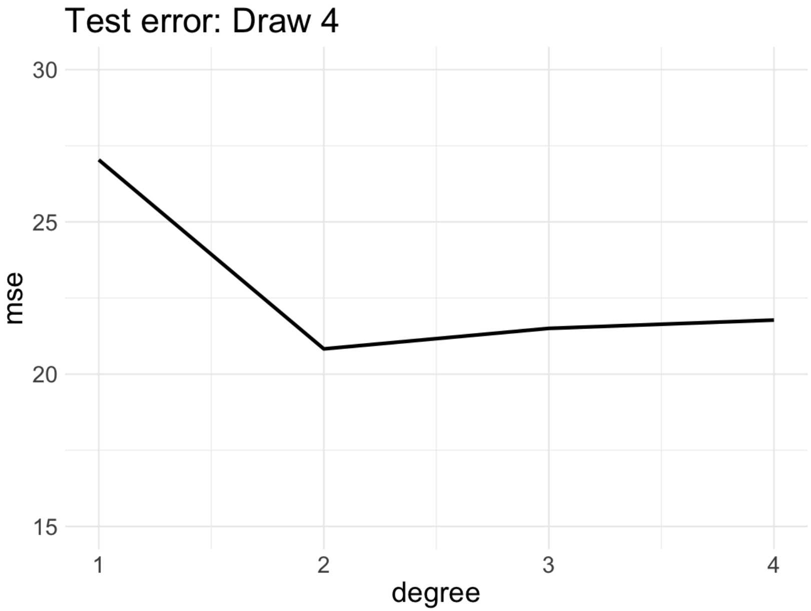
```
##   [1]  1  3  4  5  8  9 10 12 13 14 15 16 20
```


Test error: Draw 3

Validating models

Want to choose best degree of polynomial, for $\text{mpg} = \beta_0 + \beta_1 f(\text{horsepower}) + \varepsilon$

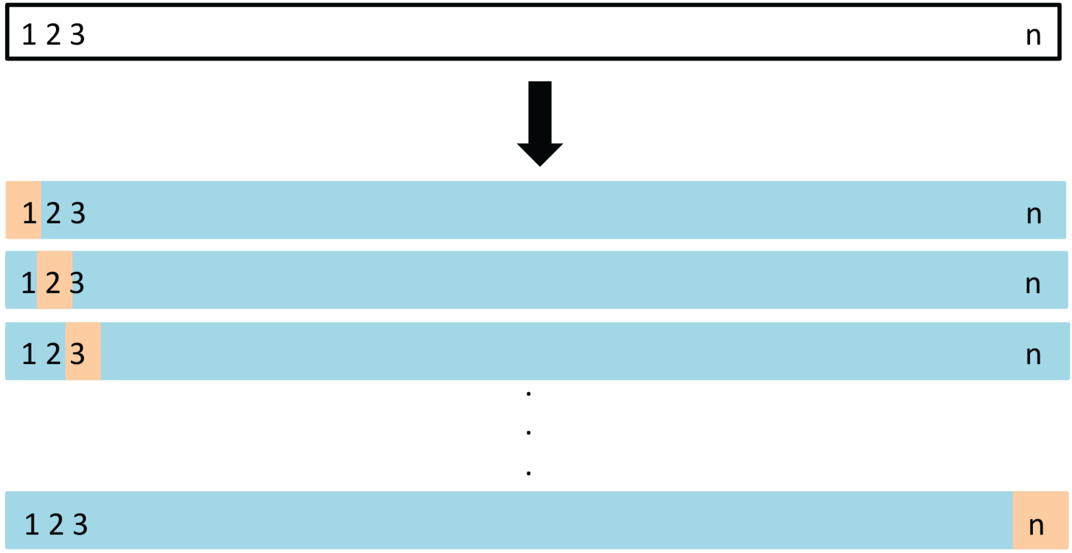
```
##  [1]  1  2  3  4  5 11 13 15 18 19 21 23 24
```



The variability between different draws of test sets can be large. This can provide poor choice of model, or misleading estimate of error.

LOOCV

Leave-one-out (LOOCV) cross-validation: n validation sets, each with ONE observation left out.



LOOCV

Leave-one-out (LOOCV) cross-validation: n validation sets, each with ONE observation left out. For each set, $i = 1, \ldots, n$, compute the MSE_i.

The LOOCV estimate for the test MSE is the average of these n test error estimates:

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} MSE_i$$

LOOCV

There is a computational shortcut, for linear or polynomial models, where not all n models need to be fitted.

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \frac{(y_i - \hat{y})^2}{1 - h_i}$$

where $h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i'}^{n} (x_{i'} - \bar{x})^2}$ (known as *leverage* from regression diagnostics).

Let's take a look at how this works

```
# This has the cv.glm function
library(boot)
glm.fit <- glm(mpg ~ horsepower,
               data=Auto)
# MSE on all observations
mse(glm.fit, Auto)
```

```
## [1] 23.94366
```

```
# LOOCV by default
cv.glm(Auto, glm.fit)$delta[1]
```

```
## [1] 24.23151
```

Compare with manual calculation

```
# Drop one observation our for fitting
m <- NULL
for (i in 1:nrow(Auto)) {
  fit <- glm(mpg ~ horsepower,
             data=Auto[-i,])
  m <- c(m,
         (Auto[i,]$mpg -
            predict(fit, Auto[i,]))^2)
}
head(m, 3)
```

```
##        1        2        3
## 2.020010 1.250924 3.068052
```

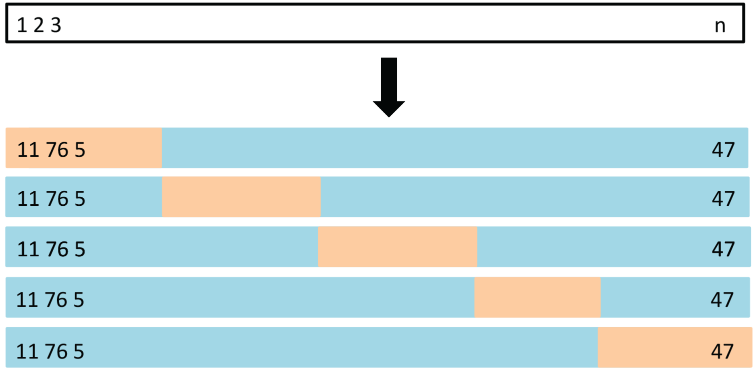
```
mean(m)
```

```
## [1] 24.23151
```

👍

k-fold cross validation

1. Divide the data set into k different parts.
2. Remove one part, fit the model on the remaining $k-1$ parts, and compute the MSE on the omitted part.
3. Repeat k times taking out a different part each time



k-fold cross validation

1. Divide the data set into k different parts.
2. Remove one part, fit the model on the remaining $k-1$ parts, and compute the MSE on the omitted part.
3. Repeat k times taking out a different part each time

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{n} MSE_k$$

- LOOCV is a special case of k-fold cross-validation.
- Bias-variance trade-off:
  - one validation set overestimates test error, LOOCV approximately unbiased estimates, k-fold CV intermediate
  - LOOCV has higher variance than does k-fold CV
  - choice of $k = 5$ or 10 is a compromise

```r
glm.fit <- glm(mpg ~ horsepower,
               data=Auto)
# MSE on all observations
mse(glm.fit, Auto)
```

```
## [1] 23.94366
```

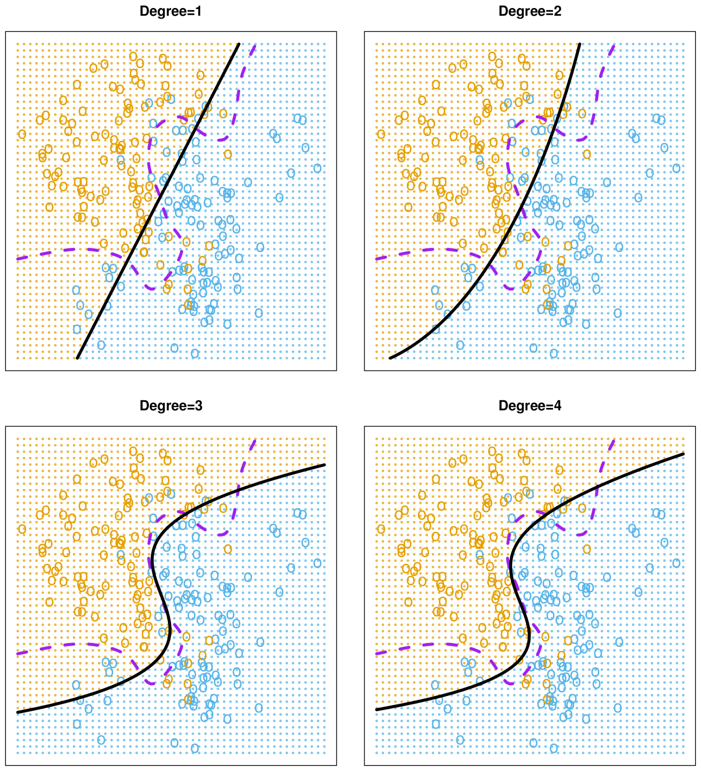
```r
# LOOCV by default
cv.glm(Auto, glm.fit)$delta[1]
```

```
## [1] 24.23151
```

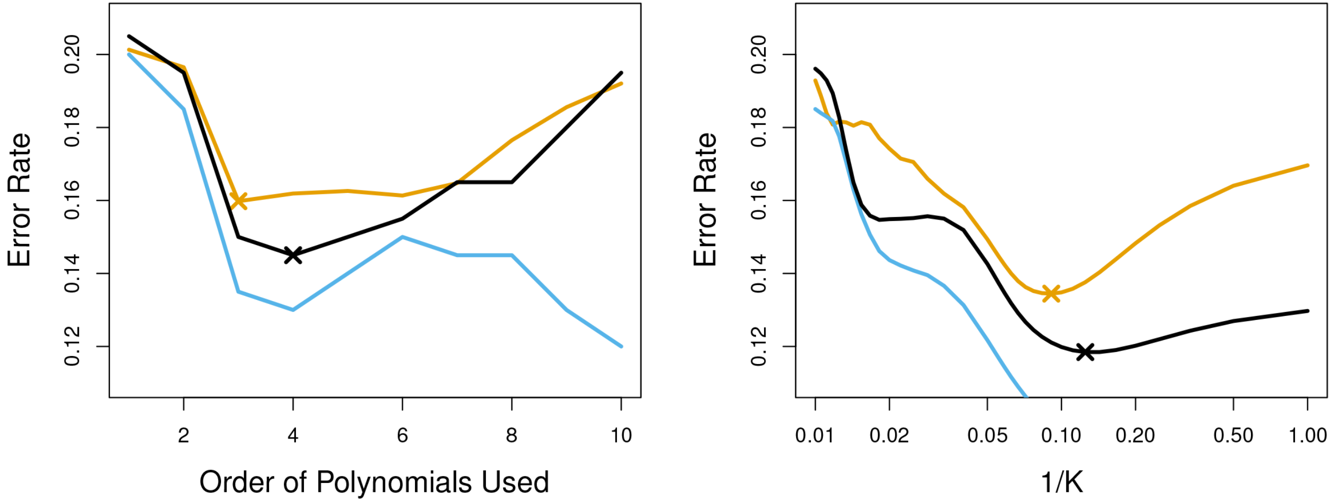
```r
# 5-fold CV
cv.glm(Auto, glm.fit, K=5)$delta[1]
```

```
## [1] 24.3191
```

Classification



Classification



Black line is 10-fold CV; training and TRUE test error for different choices of polynomial (left) and KNN classifier (right).

(Chapter 5/ 5.8)

Bootstrap procedure

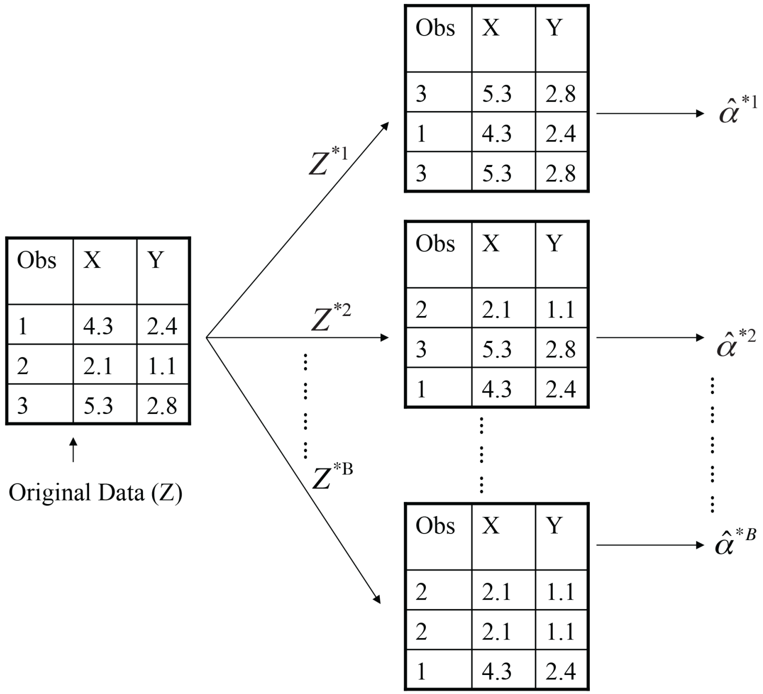
- Draw B independent bootstrap samples $X^{*(1)}, \dots, X^{*(B)}$ from $\hat{P}$:

$$X_1^{*(b)}, \dots, X_n^{*(b)} \sim \hat{P} \quad b = 1, \dots, B.$$

- Evaluate the bootstrap replications:

$$\hat{\theta}^{*(b)} = s(X^{*(b)}) \quad b = 1, \dots, B.$$

- Estimate the quantity of interest from the distribution of the $\hat{\theta}^{*(b)}$



Obs	X	Y
1	4.3	2.4
2	2.1	1.1
3	5.3	2.8

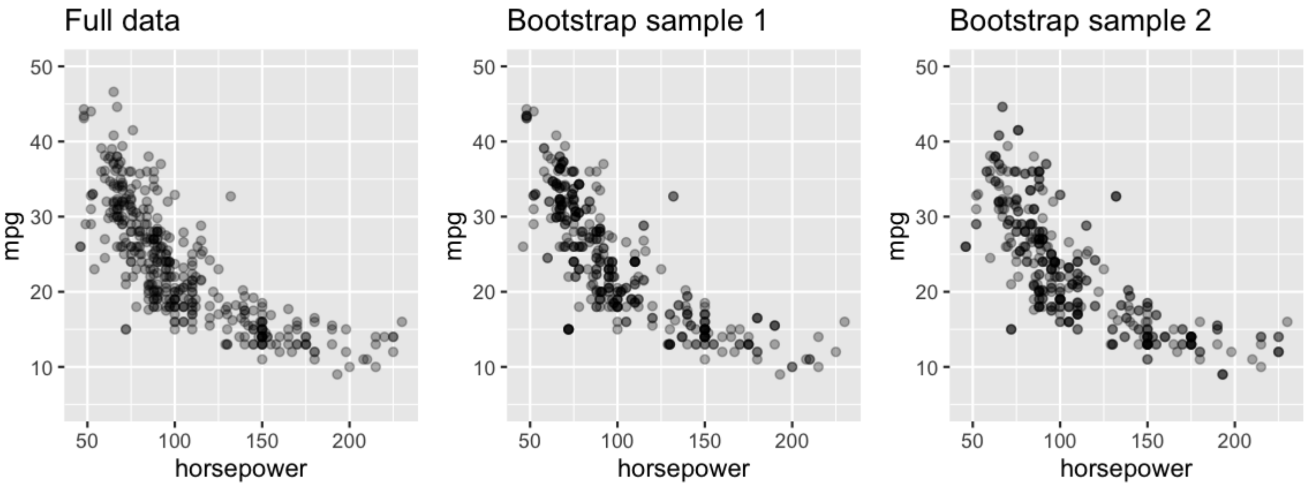
Original Data (Z)

Obs	X	Y
3	5.3	2.8
1	4.3	2.4
3	5.3	2.8

Obs	X	Y
2	2.1	1.1
3	5.3	2.8
1	4.3	2.4

Obs	X	Y
2	2.1	1.1
2	2.1	1.1
1	4.3	2.4

Bootstrap samples



Example - bootstrap model

- Fit the model on a set of bootstrap samples, and keep track of how well it predicts the original dataset

$$\mathrm{Err}_{\mathrm{boot}} = \frac{1}{B}\frac{1}{n}\sum_{b=1}^{B}\sum_{i=1}^{n} L(y_i, \hat{f}^{*b}(x_i))$$

- Each of these bootstrap data sets is created by sampling with replacement, and is the same size as our original dataset. As a result some observations may appear more than once in a given bootstrap data set and some not at all.

```r
glm.fit <- glm(mpg ~ horsepower,
               data=Auto)
# MSE on all observations
mse(glm.fit, Auto)
```

```
## [1] 23.94366
```

```r
# LOOCV by default
cv.glm(Auto, glm.fit)$delta[1]
```

```
## [1] 24.23151
```

```r
# 5-fold CV
cv.glm(Auto, glm.fit, K=5)$delta[1]
```

```
## [1] 24.10951
```

```r
mse.fn <- function(d, index) {
  glm.fit <- glm(mpg ~ horsepower,
                 data=d[index,])
  # MSE on all observations
  return(m = mse(glm.fit,
                 d[-unique(index),]) )
}
m <- NULL
for (i in 1:1000) {
  idx <- sample(nrow(Auto),
                replace = TRUE)
  m <- c(m, mse.fn(Auto, idx))
}
mean(m)
```

```
## [1] 24.39521
```

Bootstrap for error on a statistic

Overall median is 22.75

```r
# Bootstrap median, and standard error
med.fn <- function(d, index) {
  med <- median(d$mpg[index])
  med
}
bmed <- boot(Auto, med.fn, R=1000)
```

```r
bmed
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Auto, statistic = med.fn, R = 1
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1*    22.75 -0.10245   0.7635539
```

Uses and variants of the bootstrap

- Common uses:
  - Computing standard errors for complex statistics
  - Prediction error estimation
  - Bagging (Bootstrap aggregating) ML models
- Types of bootstrap based on different assumptions:
  - block bootstrap
  - sieve bootstrap
  - smooth bootstrap
  - residual bootstrap
  - wild bootstrap

👩‍💻 Made by a human with a computer

Slides at https://iml.numbat.space.

Code and data at https://github.com/numbats/iml.

Created using R Markdown with flair by **xaringan**, and **kunoichi** (female ninja) style.