Class 12: Resampling and Cross-Validation

MGSC 310

Prof. Jonathan Hersh

Class 12: Announcements

1. Quiz 5 posted tonight, due Friday @ midnight
2. Problem Set 3 posted, Due Oct 13
3. Data Analytics Week!
4. Feedback on course thus far
    1. https://chapmanu.co1.qualtrics.com/jfe/form/SV_5sD2pkboZkka3hr

Register on Handshake to get access to the following virtual events!

Data Analytics Accelerator Program Info Session
Monday, October 5 | 11 a.m. PST
Interested in pursuing a career in the growing field of data analytics? The Argyros School of Business is proud to present the new career skills-focused Analytics Accelerator Program. Learn more about what hard skills are needed to land a successful career in data analytics. Hear from Professor Toplansky and Dr. Hersh about how you can propel your success and prepare for 21st Century jobs that pay a premium.

Careers in Data Analytics
Tuesday, October 6 | 12 p.m. PST
Hear from the renowned authors of Build a Career in Data Science, Jacqueline Nolis and Emily Robinson about careers in data analytics.

Data Analytics Industry Panel
Thursday, October 8 | 4:30 p.m. PST
This data analytics panel will feature industry experts in analytics from entertainment, healthcare, technology, and more.

Entertainment Analytics: Turning Data Into Insights
Friday, October 9| 12 p.m. PST
Come see a live demo and learn about turning data into actionable insights in Entertainment Analytics with Andre Vargas Head of the data department at leading entertainment and sports agency, Creative Artists Agency (CAA).

CHAPMAN UNIVERSITY | Argyros School of Business and Economics

Class 12: Outline

1. Fun with R - rtweet
2. The Bootstrap!
3. Leave One Out Cross-Validation
4. K-Fold Cross-Validation

Scraping Twitter with rtweet Package

rtweet

build passing | CRAN 0.7.0 | codecov 65% | DOI 10.5281/zenodo.2528481 | Peer Reviewed
repo status Active | downloads 22K/month | downloads 264K | lifecycle maturing
JOSS 10.21105/joss.01829

R client for accessing Twitter's REST and stream APIs. Check out the rtweet package documentation website.

Package Functionality

There are several R packages for interacting with Twitter's APIs. See how {rtweet} compares to these others in the chart below.

Fun_rtweet.R | lab_class_12_resampling.R

Source on Save | Run

```r

library('tidyverse')
install.packages('rtweet')
library('rtweet')

# note you will need to go through Twitter Authorization to download tweets.
# follow this vignette for more info
vignette("auth", package = "rtweet")
# or go here: rtweet.info/articles/auth.html

# get_friends() finds all people a users has followed
nas_friends <- get_friends("LilNasX")
print(nas_friends)

# get_followers collects everyone who follows a user!
prof_followers <- get_followers("DogmaticPrior")
prof_followers


# Search for a keyword or phrase
rstats_tweets <- search_tweets(q = "rstats")
rstats_tweets

Chap_tweets <- search_tweets(q = "ChapmanU")
Chap_tweets
```

https://github.com/ropensci/rtweet

Scraping Twitter with rtweet Package

rtweet

build passing | CRAN 0.7.0 | codecov 65% | DOI 10.5281/zenodo.2528481 | Peer Reviewed
repo status Active | downloads 22K/month | downloads 264K | lifecycle maturing
JOSS 10.21105/joss.01829

R client for accessing Twitter's REST and stream APIs. Check out the rtweet package documentation website.

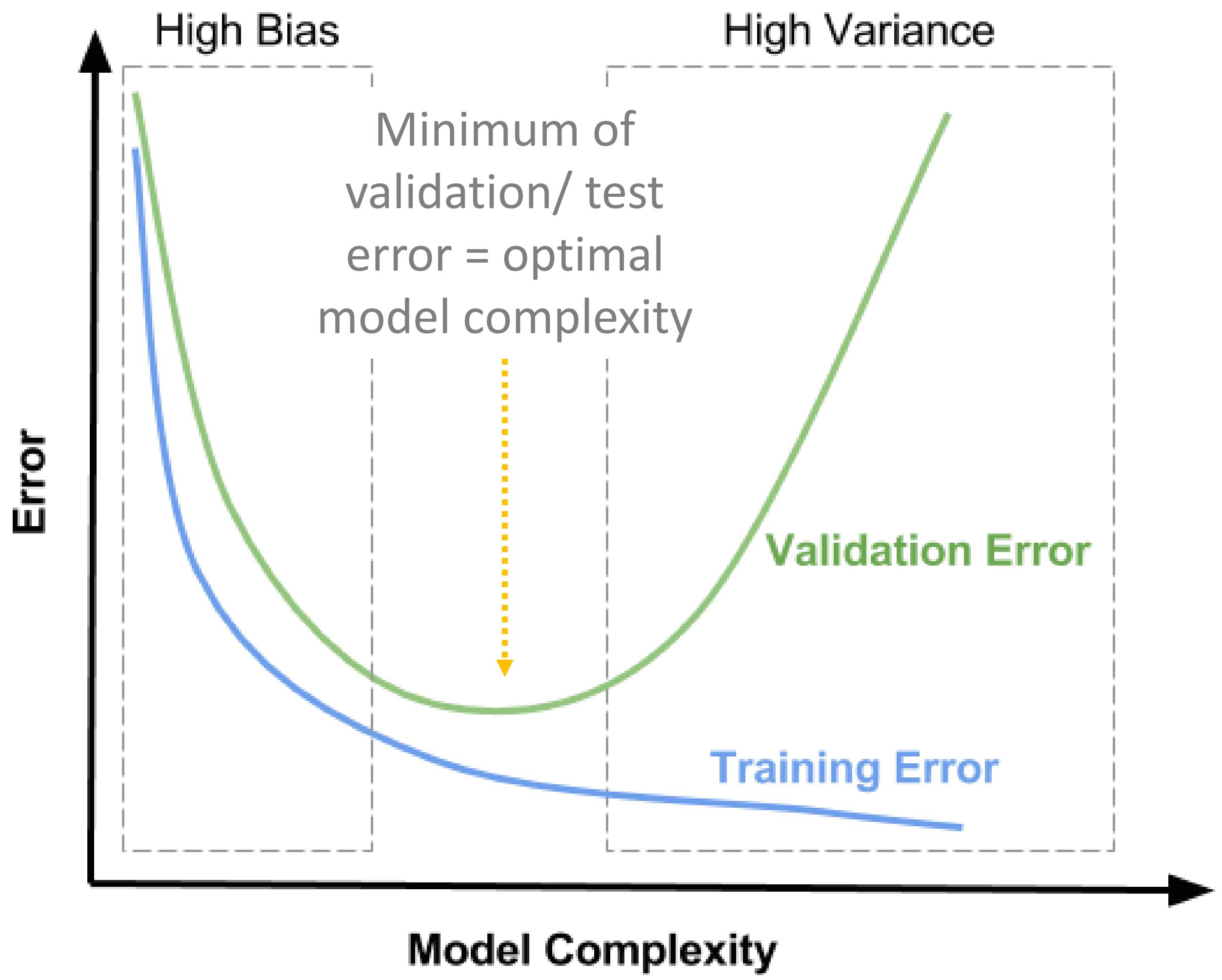
Package Functionality

There are several R packages for interacting with Twitter's APIs. See how {rtweet} compares to these others in the chart below.

```
# Exercises
# 1.  Go through the link below to get a Twitter API key
#     rtweet.info/articles/auth.html
# 2.  Use the get_friends() function to who follows one of your favorite account
# 3.  Use the get_followers() to find who follows one of your favorite accounts
# 4.  Use the get_timeline() function to see the timeline of tweets
#     for one of your favorite accounts
# 5.  Use the lubridate function to see the weekly or daily frequency of tweets
#     and plot this using geom_bar()
```

https://github.com/ropensci/rtweet

Recall Bias-Variance Tradeoff



- More model complexity test performance, but beyond a certain point it can increase test/validation error
- Note that training error always increases with model complexity!
- Key is determining optimal model complexity (in linear models, more complexity = more variables)

Resampling: Test-Validation Set Approach

- Recall: to approximate out of sample error we can set up a test and training split
- Problem: we only get one shot at building a test set. What if we select a weird test set (high variance to MSE_{tset})
- Can always build multiple test sets, but may not have enough observations for multiple test sets

$\hat{f}(X^{train})$ training

$\hat{y}^{test} = \hat{f}(X^{test})$ test

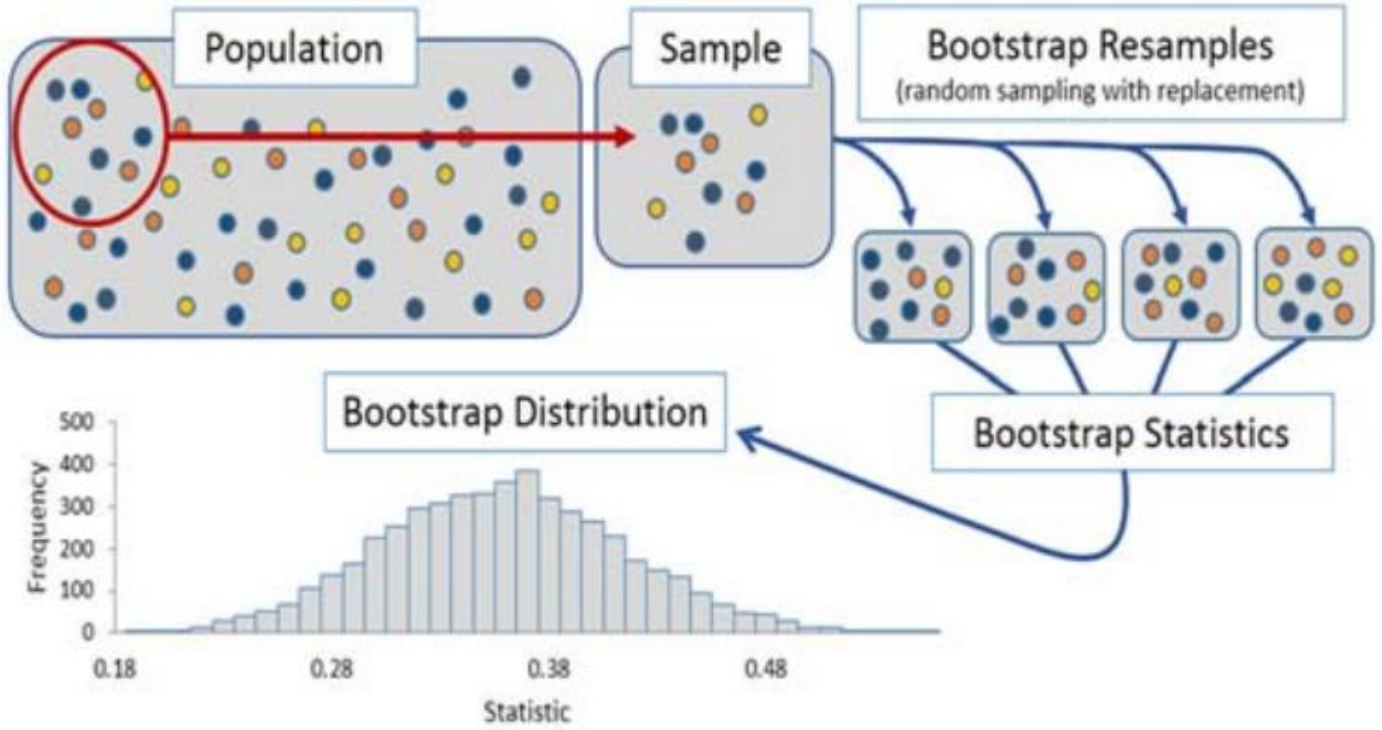
mpg	cyl	Displ
20	4	3
15	6	5
12	4	2.4
10	8	4.6
14	6	3
25	4	2

$$MSE_{tset} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left(y_i^{test} - \hat{y}^{test}\right)^2$$

Bootstrap

- The idea of the bootstrap is we take the original data (which is itself a sample from some population of possible data) and generate B bootstrap resamples.

- To do that we sample with replacement the original dataset until we have B bootstrap datasets, each of size n_b

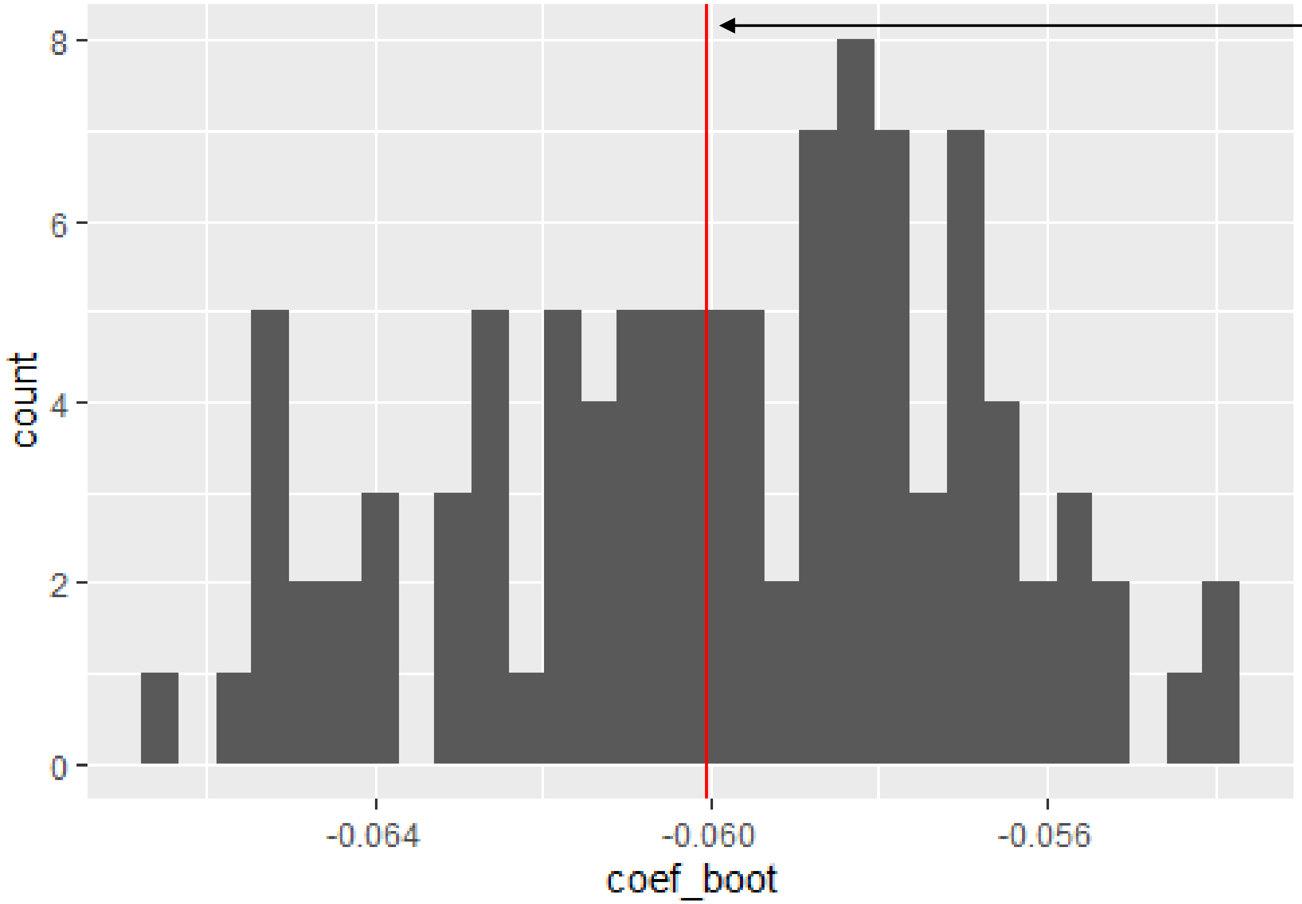


Bootstrapping in R

```r
B = 100 # number of bootstraped datasets
n_boot = 200 # size of each bootstrapped sample
coef_boot = NULL
for(b in 1:B){
  idx <- sample(1:nrow(Auto_sub),
                size = n_boot, replace = TRUE)
  mod <- lm(mpg ~ displacement,
            data = Auto_sub %>% slice(idx))
  coef_boot[b] <- mod$coefficients[2]
}
```

- Again, many ways to do it. First we do it by hand.

Bootstrapping in R



Linear model coefficient on original sample

Each point shows a coefficient from a different bootstrapped sample

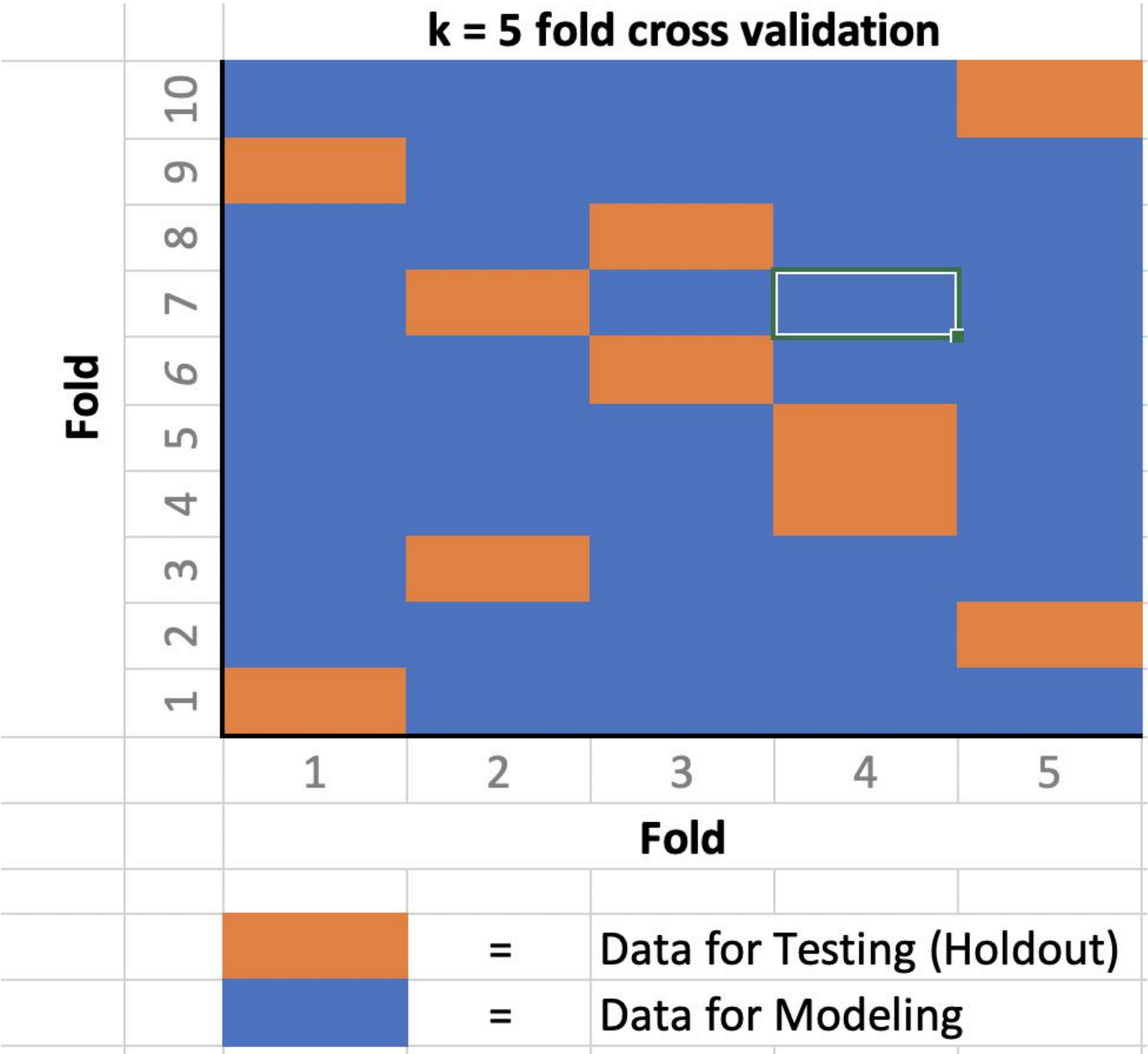
```
mod_lm <- lm(mpg ~ displacement,
             data = Auto_sub)

coef_boot <- data.frame(coef_boot =
                          coef_boot)

ggplot(coef_boot, aes(x = coef_boot)) +
  geom_histogram() +
  geom_vline(xintercept = mod_lm$coefficients[2],
             color = "red")
```

K-Fold Cross-Validation

- In K-Fold Cross Validation we partition (divide) data into K distinct groups
- Fit a model using data excluding group 1, use that model to predict into group 1.
- Fit a model using data excluding group 2, etc
- Proceed until we have yhats for every group



Resampling: K-Fold Cross-Validation

- We start by randomly assigning each data point to one of k folds
- Here we are setting k = 3
- We fit a model excluding data from fold 1
- That model is used to predict into fold 1

$$\hat{f}_{X^{-\{1\}}}(X^{-\{1\}})$$

$$\hat{y}^{KCV}_{\{1\}} = \hat{f}_{X^{\{1\}}}(X^{\{1\}})$$

$\hat{y}^{KCV}$	Fold	mpg	cyl	Displ
	3	20	4	3
	2	15	6	5
	3	12	4	2.4
11	1	10	8	4.6
	2	14	6	3
22	1	25	4	2

$X^{-\{1\}}$: X excluding fold 1

Resampling: K-Fold Cross-Validation

- Here we are setting k = 3
- Next we fit a model excluding observations in fold 2
- That model is used to predict into fold 2

$\hat{y}^{KCV}$	Fold	mpg	cyl	Displ
	3	20	4	3
18	2	15	6	5
	3	12	4	2.4
11	1	10	8	4.6
15	2	14	6	3
22	1	25	4	2

$\hat{f}_{X^{-\{2\}}}(X^{-\{2\}})$

$\hat{y}^{KCV}_{\{2\}} = \hat{f}_{X^{\{2\}}}(X^{\{2\}})$

$X^{-\{2\}}$: X excluding fold 2

Resampling: K-Fold Cross-Validation

- Here we are setting k = 3
- Next we fit a model excluding observations in fold 3
- That model is used to predict into fold 3

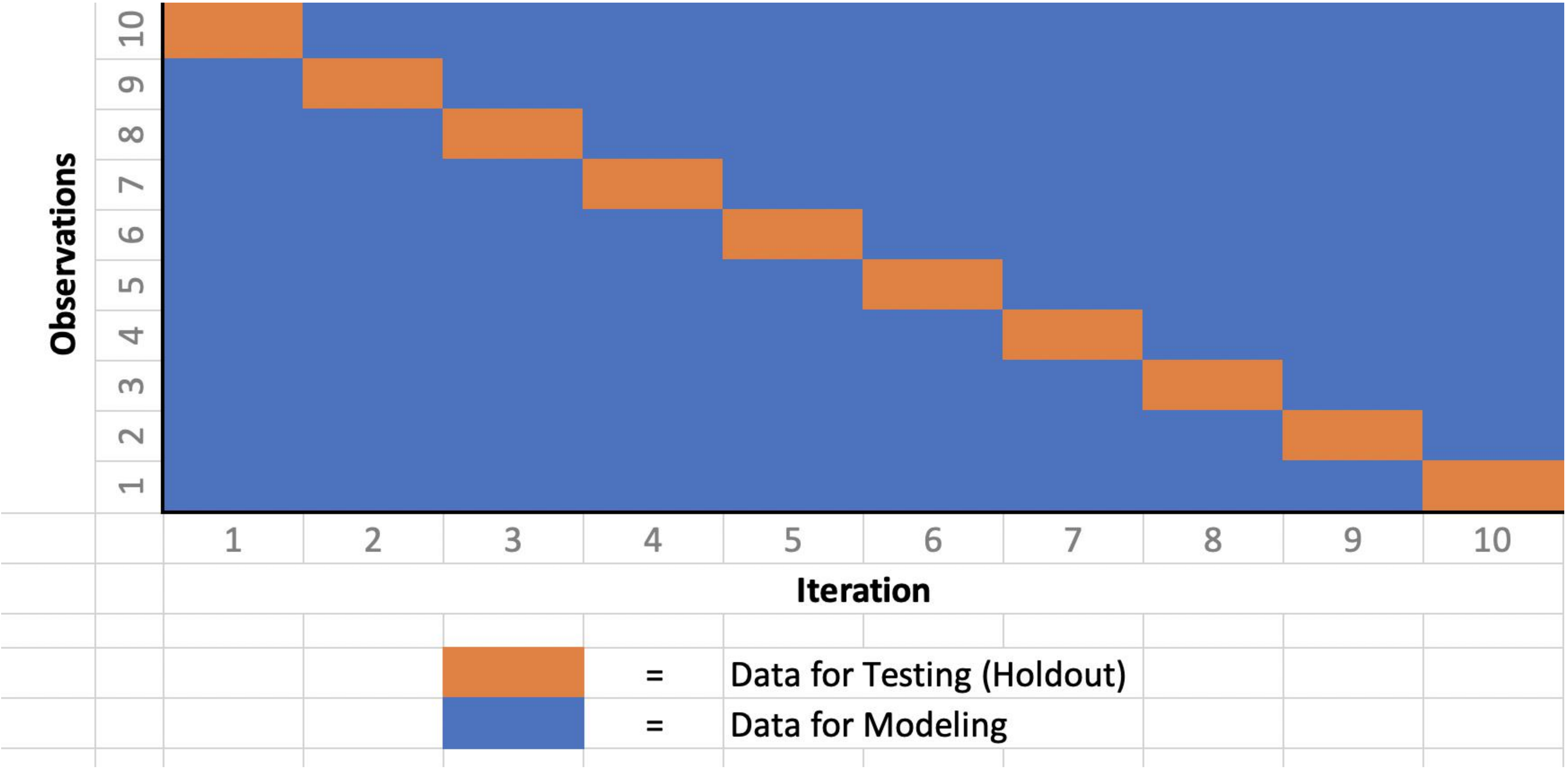
$\hat{y}^{KCV}$	Fold	mpg	cyl	Displ
22	3	20	4	3
18	2	15	6	5
12	3	12	4	2.4
11	1	10	8	4.6
15	2	14	6	3
22	1	25	4	2

$\hat{f}_{X^{-\{3\}}}(X^{-\{3\}})$

$\hat{y}^{KCV}_{\{3\}} = \hat{f}_{X^{\{3\}}}(X^{\{3\}})$

$X^{-\{3\}}$: X excluding fold 3

Leave-One-Out Cross-Validation



Resampling: Leave-One-Out Cross-Validation

- Idea of LOOCV: Let's approximate a bunch of test sets, each of size 1
- We start with estimating a model using every observation except 1.
- Use that model to predict into observation 1.

$\hat{y}^{LOOCV} = \hat{f}_{X^1}(X^1)$

$\hat{f}_{X^{-1}}(X^{-1})$ training

$\hat{y}^{LOOCV}$	mpg	cyl	Displ
18	20	4	3
	15	6	5
	12	4	2.4
	10	8	4.6
	14	6	3
	25	4	2

X^{-1}: X excluding observation 1

Resampling: Leave-One-Out Cross-Validation

- We then exclude observation 2 use observations 1,3,..,n to fit a model.
- We use the estimates from that model to predict into observation 2

training

$\hat{y}^{LOOCV} = \hat{f}_{X^{-2}}(X^{-2})$

$\hat{f}_{X^{-2}}(X^{-2})$ training

$\hat{y}^{LOOCV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
	12	4	2.4
	10	8	4.6
	14	6	3
	25	4	2

X^{-2}: X excluding observation 2

Resampling: Leave-One-Out Cross-Validation

- We then exclude observation 2 use observations 1,3,..,n to fit a model.
- We use the estimates from that model to predict into observation 2
- And we proceed in that manner until we have predictions for every observation

training

$\hat{y}^{LOOCV} = \hat{f}_{X^{-3}}(X^{-3})$

$\hat{f}_{X^{-3}}(X^{-3})$ training

$\hat{y}^{CV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
12	12	4	2.4
	10	8	4.6
	14	6	3
	25	4	2

X^{-3}: X excluding observation 3

Resampling: Leave-One-Out Cross-Validation

- We then exclude observation 2 use observations 1,3,..,n to fit a model.
- We use the estimates from that model to predict into observation 2
- And we proceed in that manner until we have predictions for every observation

$\hat{f}_{X^{\{-4\}}}(X^{\{-4\}})$ training

$\hat{y}^{LOOCV} = \hat{f}_{X^{\{-4\}}}(X^{\{4\}})$

training

$\hat{y}^{CV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
12	12	4	2.4
11	10	8	4.6
	14	6	3
	25	4	2

$X^{\{-4\}}$: X excluding observation 4

Resampling: Leave-One-Out Cross-Validation

- We then exclude observation 2 use observations 1,3,..,n to fit a model.
- We use the estimates from that model to predict into observation 2
- And we proceed in that manner until we have predictions for every observation

$\hat{f}_{X^{\{-5\}}}(X^{\{-5\}})$ training

$\hat{y}^{LOOCV} = \hat{f}_{X^{\{-5\}}}(X^{\{5\}})$

training

$\hat{y}^{LOOCV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
12	12	4	2.4
11	10	8	4.6
15	14	6	3
	25	4	2

$X^{\{-5\}}$: X excluding observation 1

Resampling: Leave-One-Out Cross-Validation

- We then exclude observation 2 use observations 1,3,..,n to fit a model.
- We use the estimates from that model to predict into observation 2
- And we proceed in that manner until we have predictions for every observation

$\hat{f}_{X^{\{-6\}}}(X^{\{-6\}})$ training

$\hat{y}^{LOOCV} = \hat{f}_{X^{\{-6\}}}(X^{\{6\}})$

$\hat{y}^{LOOCV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
12	12	4	2.4
11	10	8	4.6
11	14	6	3
22	25	4	2

$X^{\{-6\}}$: X excluding observation 6

Leave-One-Out Cross-Validation

- At the end we have a series of $\hat{y}^{LOOCV}$.
- These were calculated using models that were trained on data excluding this observation
- Kind of like a training set, right? Like N (number of rows of the dataset) training sets.
- We can then calculate MSE_{CV} which is mean-squared-error calculated using ${\hat{y}_i}^{LOOCV}$s.

$\hat{y}^{LOOCV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
12	12	4	2.4
11	10	8	4.6
11	14	6	3
22	25	4	2

$$MSE_{LOOCV} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - {\hat{y}_i}^{LOOCV}\right)^2$$

Leave-One-Out Cross-Validation in R

- Many automatic ways to do it (see boot package) but we will try by hand
- In general performance metrics are lower (better) in-sample versus cross-validated

```r
# for loop of model
mods_LOOCV <- list()
preds_LOOCV <- NULL
for(i in 1:nrow(Auto)){
  mod = lm(mpg ~ .,
           data = Auto_sub %>% slice(-i))
  preds_LOOCV[i] <- predict(mod, newdata =
                              slice(Auto_sub,i))
  mods_LOOCV[[i]] <- mod
}
```

	RMSE (pred vs true)	R2 (pred vs true)
In-Sample	3.29	0.82
LOOCV	3.37	0.81

```r
# compute RMSE LOOCV
preds_DF <- data.frame(
  preds_LOOCV = preds_LOOCV,
  preds_insample = predict(mod_insample),
  true = Auto$mpg
)

library(caret)
RMSE(preds_DF$preds_LOOCV,preds_DF$true)
RMSE(preds_DF$preds_insample,preds_DF$true)
R2(preds_DF$preds_LOOCV,preds_DF$true)
R2(preds_DF$preds_insample,preds_DF$true)
```

K-Fold Cross-Validation in R

```r
#-------------------------------------------------
### k-fold Cross validation
#-------------------------------------------------
Auto_sub <-
  mutate(Auto_sub,
         folds = createFolds(Auto_sub$mpg,
                             k = 10, list = FALSE)
  )
```

```r
### K-Fold Cross Validation
nfolds <- 10
preds_10FoldCV_DF <- data.frame(
  folds = Auto_sub$folds,
  preds_10FoldCV = rep(NA,nrow(Auto_sub))
)
```

```
> Auto_sub$folds
  [1]  1  9  3  6  3  3  1  9  9  1  3 10  9  7  8  5  6  4  4  7 10  8  5
 [24]  3  9  5  4  7  7  1  2  3  7  1  2  7  5  7  1  5  1  9 10  2  2 10
 [47]  3  1  9 10  4  1  5  9  6  1  6  1  1  7  2  8  8  3 10 10 10  5  2
 [70]  1  1  9  1  2  8  3  1  2  5  2  2  8  7  9 10 10  5  4  4  7  3  3
 [93]  2  5  6  7 10  3  8  9 10 10  8 10  2  9  4  8  8  5 10  9  4  9  9
[116]  8  3  5  6  7  1  7  9  6  6  5  5  3 10  2  9  7  4  4  2  6  8  2
```

```
> preds_10FoldCV_DF
   folds preds_10FoldCV
1      1             NA
2      9             NA
3      3             NA
4      6             NA
5      3             NA
6      3             NA
7      1             NA
8      9             NA
9      9             NA
10     1             NA
11     3             NA
12    10             NA
```

K-Fold Cross-Validation in R

```
for(i in 1:nfolds){
  mod <- lm(mpg ~ .,
            data = Auto_sub %>%
              filter(folds != i))
  preds <- predict(mod,
                   newdata = filter(Auto_sub,
                                    folds == i))
  preds_10FoldCV_DF[preds_10FoldCV_DF$folds == i,"preds_10FoldCV"]  <- preds
}
```

	RMSE (pred vs true)	R2 (pred vs true)
In-Sample	3.293	0.8215
LOOCV	3.382	0.8118
10 Fold CV	3.357	0.8147

- On average 10-Fold CV diagnostic measures will be in-between in-sample measures and LOOCV measures.

K-Fold CV Versus LOOCV

- Advantages of K-Fold CV over LOOCV
  - Only need to estimate K models
- Disadvantages
  - Higher variance (more uncertainty in y_hats)
- Because of computational cost K-Fold CV more commonly used

$\hat{y}^{KCV}$	mpg	cyl	Displ
18	20	4	3
16	15	6	5
12	12	4	2.4
11	10	8	4.6
11	14	6	3
22	25	4	2

$$MSE_{KCV} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i{}^{KCV}\right)^2$$