Resampling: Cross-Validation and Bootstrapping

Prof Wells

STA 295: Stat Learning

February 27th, 2024

Outline

In today's class, we will...

- Define and discuss resampling and cross-validation
- Investigate methods of cross-validation (LOOCV and k-fold cv)
- Discuss the bootstrap for approximating distribution of statistics

Section 1

Validation

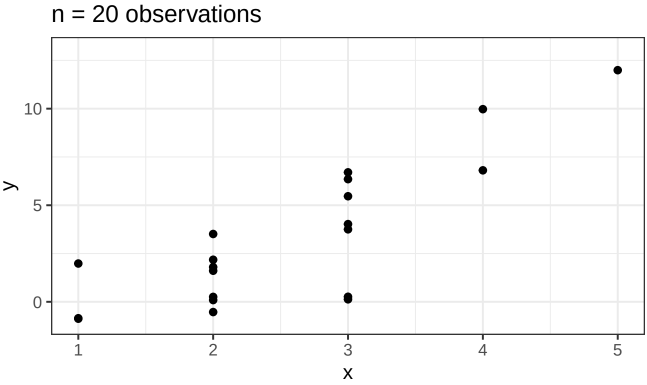
Model Selection Using Training Data

When sample size is small relative to the number of predictors, we might consider building and comparing models using **all** available data

Model Selection Using Training Data

When sample size is small relative to the number of predictors, we might consider building and comparing models using **all** available data

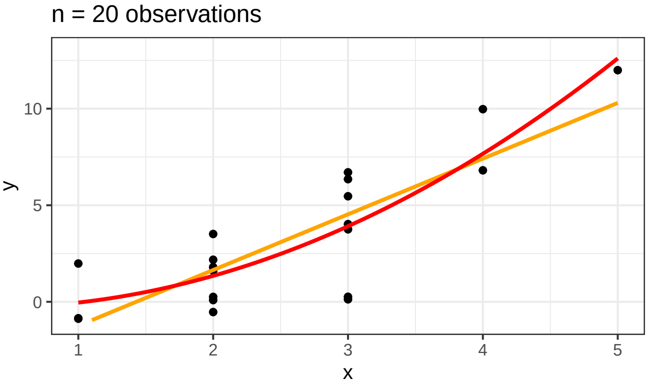
- Suppose we want to determine whether a linear or quadratic model is more appropriate for the following data set



Model Selection Using Training Data

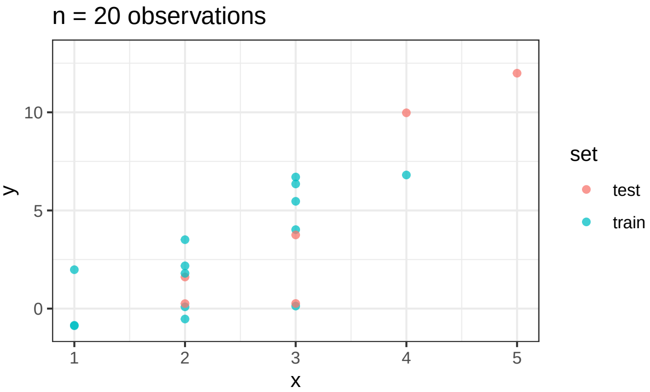
When sample size is small relative to the number of predictors, we might consider building and comparing models using **all** available data

- Suppose we want to determine whether a linear or quadratic model is more appropriate for the following data set



Model Selection Using Training Data

Dividing data into training and test sets might not be a good idea:



- Using a 70-30 split with $n = 20$ means only 6 observations in test set
- Train and test sets are likely very dissimilar

Model Selection Using Training Data

In this case, we can compare models using metrics computed solely on training data:

```
mod1 <- lm(y ~ x, data = my_data)
summary(mod1)
```

```
##
## Call:
## lm(formula = y ~ x, data = my_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4001 -0.9300  0.2575  1.7263  3.2217
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -4.1231     1.2710  -3.244  0.00451 **
## x             2.8842     0.4626   6.235 6.99e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.117 on 18 degrees of freedom
## Multiple R-squared:  0.6835, Adjusted R-squared:  0.6659
## F-statistic: 38.88 on 1 and 18 DF,  p-value: 6.989e-06
```

Model Selection Using Training Data

In this case, we can compare models using metrics computed solely on training data:

```
mod2 <- lm(y ~ poly(x, degree = 2, raw = T), data = my_data)
summary(mod2)
```

```
##
## Call:
## lm(formula = y ~ poly(x, degree = 2, raw = T), data = my_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7842 -0.9159 -0.0255  1.6695  2.7900
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -0.2383     2.4578  -0.097    0.924
## poly(x, degree = 2, raw = T)1  -0.3917     1.8617  -0.210    0.836
## poly(x, degree = 2, raw = T)2   0.5919     0.3270   1.810    0.088 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.995 on 17 degrees of freedom
## Multiple R-squared:  0.7347, Adjusted R-squared:  0.7034
## F-statistic: 23.53 on 2 and 17 DF,  p-value: 1.266e-05
```

Poll: Training Error

Consider a data set with n training observations and p potential predictors. Which of the following methods are likely to have the smallest **training** error rate?

a. Multilinear regression with p predictors
b. Simple linear regression with 1 predictor
c. Non-linear regression with a polynomial of 1 predictor
d. KNN with $K = 1$
e. KNN with $K = p$

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error
  - And not all models will have the same bias, making comparison difficult

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error
  - And not all models will have the same bias, making comparison difficult
- One fix is to partition data into training and test sets:

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error
  - And not all models will have the same bias, making comparison difficult
- One fix is to partition data into training and test sets:
  - Build the model using only the training data, then assess accuracy using only test data

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error
  - And not all models will have the same bias, making comparison difficult
- One fix is to partition data into training and test sets:
  - Build the model using only the training data, then assess accuracy using only test data
- Generally, we split data using random methods (each observation has equal chance of being in training or test set)

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error
  - And not all models will have the same bias, making comparison difficult
- One fix is to partition data into training and test sets:
  - Build the model using only the training data, then assess accuracy using only test data
- Generally, we split data using random methods (each observation has equal chance of being in training or test set)
- When deciding split ratio, need to balance two competing concerns:

Validation Set

- Assessing model accuracy only on training sets will usually underestimate error
    - And not all models will have the same bias, making comparison difficult
- One fix is to partition data into training and test sets:
    - Build the model using only the training data, then assess accuracy using only test data
- Generally, we split data using random methods (each observation has equal chance of being in training or test set)
- When deciding split ratio, need to balance two competing concerns:
    - Include enough data in training set to build accurate model
    - Include enough data in test set to provide reliable estimate of error
    - Generally, a 70-30 training test split tends to work well for most problems.

Fuel Economy

The `cars2010` data set from the `AppliedPredictiveModeling` package contains fuel efficiency and other variables for 1107 cars and trucks from 2010

```
##      EngDispl NumCyl Transmission      FE AirAspirationMethod NumGears
## 1088      4.7      8          AM6 28.0198  NaturallyAspirated        6
## 1089      4.7      8           M6 25.6094  NaturallyAspirated        6
## 1090      4.2      8           M6 26.8000  NaturallyAspirated        6
## 1091      4.2      8          AM6 25.0451  NaturallyAspirated        6
## 1092      5.2     10          AM6 24.8000  NaturallyAspirated        6
## 1093      5.2     10           M6 23.9000  NaturallyAspirated        6
##      TransLockup TransCreeperGear        DriveDesc IntakeValvePerCyl
## 1088           1                0 TwoWheelDriveRear                 2
## 1089           1                0 TwoWheelDriveRear                 2
## 1090           1                0     AllWheelDrive                 2
## 1091           1                0     AllWheelDrive                 2
## 1092           0                0     AllWheelDrive                 2
## 1093           0                0     AllWheelDrive                 2
##      ExhaustValvesPerCyl CarlineClassDesc VarValveTiming VarValveLift
## 1088                   2         2Seaters              1            0
## 1089                   2         2Seaters              1            0
## 1090                   2         2Seaters              1            0
## 1091                   2         2Seaters              1            0
## 1092                   2         2Seaters              1            0
## 1093                   2         2Seaters              1            0
```

Important Predictors

We are interested in modeling Fuel Efficiency (FE) as a function of other car attributes.

- Let's consider just numeric variable first:

```
cars2010 %>%
  select_if(is.numeric) %>%
  cor(cars2010$FE)
```

```
##                          [,1]
## EngDispl          -0.78739383
## NumCyl            -0.74021798
## FE                 1.00000000
## NumGears          -0.21128488
## TransLockup       -0.27193887
## TransCreeperGear  -0.06962168
## IntakeValvePerCyl  0.28034403
## ExhaustValvesPerCyl 0.33565285
## VarValveTiming     0.12495278
## VarValveLift       0.09621127
```

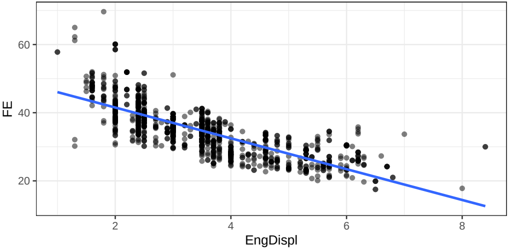
Important Predictors

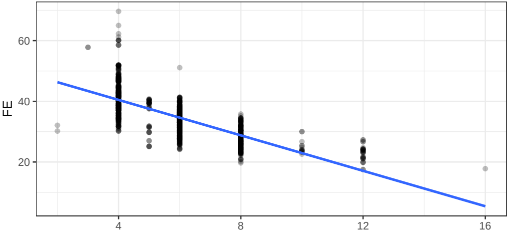
We are interested in modeling Fuel Efficiency (FE) as a function of other car attributes.

- Let's consider just numeric variable first:

```
cars2010 %>%
  select_if(is.numeric) %>%
  cor(cars2010$FE)
```

```
##                          [,1]
## EngDispl          -0.78739383
## NumCyl            -0.74021798
## FE                 1.00000000
## NumGears          -0.21128488
## TransLockup       -0.27193887
## TransCreeperGear  -0.06962168
## IntakeValvePerCyl  0.28034403
## ExhaustValvesPerCyl 0.33565285
## VarValveTiming     0.12495278
## VarValveLift       0.09621127
```



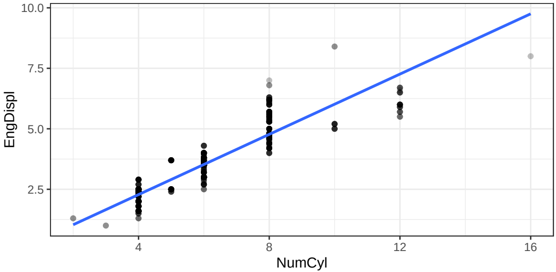


Collinearity

- We may want to include both `EngDispl` and `NumCyl` in our model for `FE`.
  - But we have reason to suspect that these variables are correlated with **each other**, since both measure the size of an engine

Collinearity

- We may want to include both EngDispl and NumCyl in our model for FE.
  - But we have reason to suspect that these variables are correlated with **each other**, since both measure the size of an engine




```
cor(cars2010$EngDispl, cars2010$NumCyl)
```

```
## [1] 0.90626
```

Validation Set

Let's create a validation set using `initial_split` in the `rsample` package

Validation Set

Let's create a validation set using `initial_split` in the `rsample` package

```
library(rsample)
set.seed(999)
cars_initial <- initial_split(cars2010)

cars_train <- training(cars_initial)
cars_val <- testing(cars_initial)
```

Validation Set

Let's create a validation set using `initial_split` in the `rsample` package

```
library(rsample)
set.seed(999)
cars_initial <- initial_split(cars2010)

cars_train <- training(cars_initial)
cars_val <- testing(cars_initial)
```

- The `dim` function in `rsample` returns the number of observations and variables present in a split:

```
cars_train %>% dim()
```

```
## [1] 830  14
```

```
cars_val %>% dim()
```

```
## [1] 277  14
```

Two Models

- Since `EngDispl` is most strongly correlated with `FE`, we will include it in our models.
- And we'll create another model that also includes `NumCyl`.

Two Models

- Since EngDispl is most strongly correlated with FE, we will include it in our models.
- And we'll create another model that also includes NumCyl.

```
mod1 <- lm(FE ~ EngDispl, data = cars_train)
summary(mod1)
```

```
##
## Call:
## lm(formula = FE ~ EngDispl, data = cars_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.766  -3.196  -0.502   2.744  27.000
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  51.0108     0.4683  108.93   <2e-16 ***
## EngDispl     -4.6501     0.1256  -37.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.7 on 828 degrees of freedom
## Multiple R-squared:  0.6235, Adjusted R-squared:  0.6231
## F-statistic:  1371 on 1 and 828 DF,  p-value: < 2.2e-16
```

Two Models

- Since `EngDispl` is most strongly correlated with `FE`, we will include it in our models.
- And we'll create another model that also includes `NumCyl`.

```
mod1 <- lm(FE ~ EngDispl, data = cars_train)
summary(mod1)
```

```
##
## Call:
## lm(formula = FE ~ EngDispl, data = cars_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.766  -3.196  -0.502   2.744  27.000
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  51.0108     0.4683  108.93   <2e-16 ***
## EngDispl     -4.6501     0.1256  -37.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.7 on 828 degrees of freedom
## Multiple R-squared:  0.6235, Adjusted R-squared:  0.6231
## F-statistic:  1371 on 1 and 828 DF,  p-value: < 2.2e-16
```

```
mod2 <- lm(FE ~ EngDispl + NumCyl, data = cars_train)
summary(mod2)
```

```
##
## Call:
## lm(formula = FE ~ EngDispl + NumCyl, data = cars_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2623  -3.0929  -0.3346   2.6825  27.1432
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  51.6371     0.5341  96.678   <2e-16 ***
## EngDispl     -4.0121     0.2924 -13.724   <2e-16 ***
## NumCyl       -0.4795     0.1986  -2.415    0.016 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.686 on 827 degrees of freedom
## Multiple R-squared:  0.6261, Adjusted R-squared:  0.6252
## F-statistic: 692.5 on 2 and 827 DF,  p-value: < 2.2e-16
```

Two Models

- Since `EngDispl` is most strongly correlated with `FE`, we will include it in our models.
- And we'll create another model that also includes `NumCyl`.

```
mod1 <- lm(FE ~ EngDispl, data = cars_train)
summary(mod1)
```

```
##
## Call:
## lm(formula = FE ~ EngDispl, data = cars_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.766  -3.196  -0.502   2.744  27.000
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  51.0108     0.4683  108.93   <2e-16 ***
## EngDispl     -4.6501     0.1256  -37.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.7 on 828 degrees of freedom
## Multiple R-squared:  0.6235, Adjusted R-squared:  0.6231
## F-statistic:  1371 on 1 and 828 DF,  p-value: < 2.2e-16
```

```
mod2 <- lm(FE ~ EngDispl + NumCyl, data = cars_train)
summary(mod2)
```

```
##
## Call:
## lm(formula = FE ~ EngDispl + NumCyl, data = cars_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2623  -3.0929  -0.3346   2.6825  27.1432
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  51.6371     0.5341  96.678   <2e-16 ***
## EngDispl     -4.0121     0.2924 -13.724   <2e-16 ***
## NumCyl       -0.4795     0.1986  -2.415    0.016 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.686 on 827 degrees of freedom
## Multiple R-squared:  0.6261, Adjusted R-squared:  0.6252
## F-statistic: 692.5 on 2 and 827 DF,  p-value: < 2.2e-16
```

- The MLR model has lower RSE, higher R^2, and all predictors are significant at the $\alpha = 0.05$ level. But is it really the better model?

Assess on Validation Set

Let's check RMSE on the validation set.

Assess on Validation Set

Let's check RMSE on the validation set.

```
mod1_preds <- predict(mod1, cars_val)
mod1_rmse <- sqrt( mean( (cars_val$FE - mod1_preds)^2))
mod1_rmse
```

```
## [1] 4.403297
```

Assess on Validation Set

Let's check RMSE on the validation set.

```
mod1_preds <- predict(mod1, cars_val)
mod1_rmse <- sqrt( mean( (cars_val$FE - mod1_preds)^2))
mod1_rmse
```

```
## [1] 4.403297
```

```
mod2_preds <- predict(mod2, cars_val)
mod2_mse <- sqrt(mean( (cars_val$FE - mod2_preds)^2))
mod2_mse
```

```
## [1] 4.356728
```

Assess on Validation Set

Let's check RMSE on the validation set.

```
mod1_preds <- predict(mod1, cars_val)
mod1_rmse <- sqrt( mean( (cars_val$FE - mod1_preds)^2))
mod1_rmse
```

```
## [1] 4.403297
```

```
mod2_preds <- predict(mod2, cars_val)
mod2_mse <- sqrt(mean( (cars_val$FE - mod2_preds)^2))
mod2_mse
```

```
## [1] 4.356728
```

- The MLR model (`mod2`) has slightly lower RMSE than the SLR model (`mod1`)

Assess on Validation Set

Let's check RMSE on the validation set.

```
mod1_preds <- predict(mod1, cars_val)
mod1_rmse <- sqrt( mean( (cars_val$FE - mod1_preds)^2))
mod1_rmse
```

```
## [1] 4.403297
```

```
mod2_preds <- predict(mod2, cars_val)
mod2_mse <- sqrt(mean( (cars_val$FE - mod2_preds)^2))
mod2_mse
```

```
## [1] 4.356728
```

- The MLR model (`mod2`) has slightly lower RMSE than the SLR model (`mod1`)
  - But could this be a fluke of a random validation set?
  - That is, if we took a different random split into training / validation, would `mod2` still have lower RMSE? (Since the RMSE values are so close)

Problems with Validation Sets

What are some problems with the Training / Validation approach?

Problems with Validation Sets

What are some problems with the Training / Validation approach?

- If initial data set is small, this further restricts sample size available for model building.
    - Both model and test performance may have high variance.

Problems with Validation Sets

What are some problems with the Training / Validation approach?

- If initial data set is small, this further restricts sample size available for model building.
  - Both model and test performance may have high variance.
- A single test set doesn't give estimates for the range of error

Problems with Validation Sets

What are some problems with the Training / Validation approach?

- If initial data set is small, this further restricts sample size available for model building.
  - Both model and test performance may have high variance.
- A single test set doesn't give estimates for the range of error
- Susceptible to bias from particular choice of training set.

Problems with Validation Sets

What are some problems with the Training / Validation approach?

- If initial data set is small, this further restricts sample size available for model building.
    - Both model and test performance may have high variance.
- A single test set doesn't give estimates for the range of error
- Susceptible to bias from particular choice of training set.

Resampling is drawing many samples from your training data and refitting the model for each, in order to learn more about your model.

Problems with Validation Sets

What are some problems with the Training / Validation approach?

- If initial data set is small, this further restricts sample size available for model building.
  - Both model and test performance may have high variance.
- A single test set doesn't give estimates for the range of error
- Susceptible to bias from particular choice of training set.

Resampling is drawing many samples from your training data and refitting the model for each, in order to learn more about your model.

Cross-Validation is using resampling techniques to assess model accuracy.

Section 2

Resampling

k-fold Cross Validation

- k-fold CV randomly partitions data into k sets of size n/k.
    - One subset of size n/k is chosen to be the validation set
    - Remaining $k - 1$ subsets are used as training set to build the model.

k-fold Cross Validation

- k-fold CV randomly partitions data into k sets of size n/k.
  - One subset of size n/k is chosen to be the validation set
  - Remaining $k-1$ subsets are used as training set to build the model.
- The process is repeated for each possible validation set, and the average error rate computed among all partitions is computed

k-fold Cross Validation

- k-fold CV randomly partitions data into k sets of size n/k.
  - One subset of size n/k is chosen to be the validation set
  - Remaining $k-1$ subsets are used as training set to build the model.
- The process is repeated for each possible validation set, and the average error rate computed among all partitions is computed
- The cross-validation estimate $CV_{(k)}$ for average test MSE is therefore:

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i$$

k-fold Cross Validation

- k-fold CV randomly partitions data into k sets of size n/k.
  - One subset of size n/k is chosen to be the validation set
  - Remaining $k-1$ subsets are used as training set to build the model.
- The process is repeated for each possible validation set, and the average error rate computed among all partitions is computed
- The cross-validation estimate $CV_{(k)}$ for average test MSE is therefore:

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i$$

- Here, MSE_i is test MSE when the ith fold is used as validation set.

k-fold Cross Validation

- k-fold CV randomly partitions data into k sets of size n/k.
    - One subset of size n/k is chosen to be the validation set
    - Remaining $k-1$ subsets are used as training set to build the model.
- The process is repeated for each possible validation set, and the average error rate computed among all partitions is computed
- The cross-validation estimate $CV_{(k)}$ for average test MSE is therefore:

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i$$

- Here, MSE_i is test MSE when the ith fold is used as validation set.
- Since the partition into folds is random, $CV_{(k)}$ still has some variability. But less than just using a single validation set.

k-fold Cross Validation

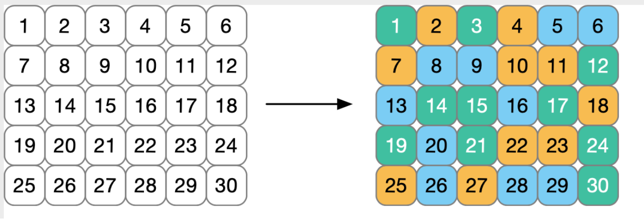
- k-fold CV randomly partitions data into k sets of size n/k.
  - One subset of size n/k is chosen to be the validation set
  - Remaining $k-1$ subsets are used as training set to build the model.
- The process is repeated for each possible validation set, and the average error rate computed among all partitions is computed
- The cross-validation estimate $CV_{(k)}$ for average test MSE is therefore:

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k}\mathrm{MSE}_i$$

- Here, MSE_i is test MSE when the ith fold is used as validation set.
- Since the partition into folds is random, $CV_{(k)}$ still has some variability. But less than just using a single validation set.
  - To reduce variability further, k-fold CV can be performed multiple times, and the results of $CV_{(k)}$ themselves averaged. This provides minimal variance AND minimal bias estimate of MSE

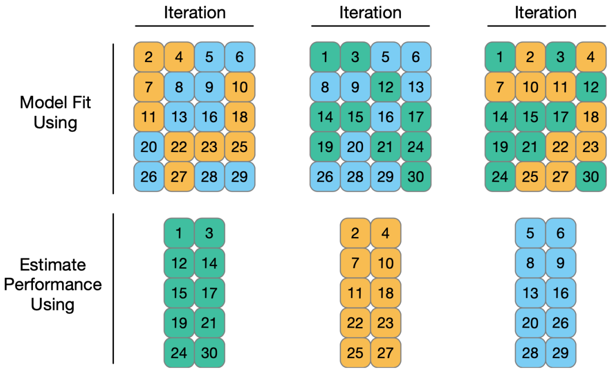
3-fold CV

- Consider 30 training observations below. Colors indicate a random fold allocation.



3-fold CV

- Each iteration uses 2 of the folds to build a model, and the remaining fold to assess performance.



- Overall performance is obtained by averaging across all 3 iterations.

CV Fuel Efficiency Models

- In Thursday's class, we'll discuss how to use the `rsample` package for cross-validating in R. For now, we'll look at the results of cross-validation.

CV Fuel Efficiency Models

- In Thursday's class, we'll discuss how to use the `rsample` package for cross-validating in R. For now, we'll look at the results of cross-validation.

Goal: Use 10-fold CV to assess whether `NumCyl` should be included in model for `FE` alongside `EngDispl`

CV Fuel Efficiency Models

- In Thursday's class, we'll discuss how to use the `rsample` package for cross-validating in R. For now, we'll look at the results of cross-validation.

Goal: Use 10-fold CV to assess whether `NumCyl` should be included in model for `FE` alongside `EngDispl`

- We first divide the data into 10 equally sized folds, and then use these folds to create 10 different splits of the data.

CV Fuel Efficiency Models

- In Thursday's class, we'll discuss how to use the `rsample` package for cross-validating in R. For now, we'll look at the results of cross-validation.

Goal: Use 10-fold CV to assess whether `NumCyl` should be included in model for `FE` alongside `EngDispl`

- We first divide the data into 10 equally sized folds, and then use these folds to create 10 different splits of the data.
  - Each *fold* represents 10% of the total data.
  - A *split* breaks the data into two parts: 90% for training and 10% for validation.
  - Each of the folds represents the validation set for exactly 1 split

CV Fuel Efficiency Models

- In Thursday's class, we'll discuss how to use the `rsample` package for cross-validating in R. For now, we'll look at the results of cross-validation.

Goal: Use 10-fold CV to assess whether `NumCyl` should be included in model for `FE` alongside `EngDispl`

- We first divide the data into 10 equally sized folds, and then use these folds to create 10 different splits of the data.
  - Each *fold* represents 10% of the total data.
  - A *split* breaks the data into two parts: 90% for training and 10% for validation.
  - Each of the folds represents the validation set for exactly 1 split
- For each of the 10 splits, we fit all models on the training set.

CV Fuel Efficiency Models

- In Thursday's class, we'll discuss how to use the `rsample` package for cross-validating in R. For now, we'll look at the results of cross-validation.

Goal: Use 10-fold CV to assess whether `NumCyl` should be included in model for `FE` alongside `EngDispl`

- We first divide the data into 10 equally sized folds, and then use these folds to create 10 different splits of the data.
  - Each *fold* represents 10% of the total data.
  - A *split* breaks the data into two parts: 90% for training and 10% for validation.
  - Each of the folds represents the validation set for exactly 1 split
- For each of the 10 splits, we fit all models on the training set.
- And for each of the 10 splits, we compute relevant error metrics on the validation set

RMSE on Folds

Below we summarize the RMSE for Model 1 and Model 2 for each of the 10 splits:

RMSE on Folds

Below we summarize the RMSE for Model 1 and Model 2 for each of the 10 splits:

id	rmse_mod1	rmse_mod2	diff
Fold01	4.242	4.203	0.039
Fold02	4.136	4.138	-0.002
Fold03	5.003	4.971	0.032
Fold04	5.085	5.026	0.060
Fold05	4.609	4.713	-0.105
Fold06	4.050	4.020	0.030
Fold07	5.285	5.245	0.041
Fold08	4.361	4.347	0.014
Fold09	4.911	4.855	0.057
Fold10	4.442	4.424	0.018

RMSE on Folds

Below we summarize the RMSE for Model 1 and Model 2 for each of the 10 splits:

id	rmse_mod1	rmse_mod2	diff
Fold01	4.242	4.203	0.039
Fold02	4.136	4.138	-0.002
Fold03	5.003	4.971	0.032
Fold04	5.085	5.026	0.060
Fold05	4.609	4.713	-0.105
Fold06	4.050	4.020	0.030
Fold07	5.285	5.245	0.041
Fold08	4.361	4.347	0.014
Fold09	4.911	4.855	0.057
Fold10	4.442	4.424	0.018

- Which folds were hardest to predict? Which were easiest?

RMSE on Folds

Below we summarize the RMSE for Model 1 and Model 2 for each of the 10 splits:

id	rmse_mod1	rmse_mod2	diff
Fold01	4.242	4.203	0.039
Fold02	4.136	4.138	-0.002
Fold03	5.003	4.971	0.032
Fold04	5.085	5.026	0.060
Fold05	4.609	4.713	-0.105
Fold06	4.050	4.020	0.030
Fold07	5.285	5.245	0.041
Fold08	4.361	4.347	0.014
Fold09	4.911	4.855	0.057
Fold10	4.442	4.424	0.018

- Which folds were hardest to predict? Which were easiest?
- On which folds did model 1 perform better?

RMSE on Folds

Below we summarize the RMSE for Model 1 and Model 2 for each of the 10 splits:

id	rmse_mod1	rmse_mod2	diff
Fold01	4.242	4.203	0.039
Fold02	4.136	4.138	-0.002
Fold03	5.003	4.971	0.032
Fold04	5.085	5.026	0.060
Fold05	4.609	4.713	-0.105
Fold06	4.050	4.020	0.030
Fold07	5.285	5.245	0.041
Fold08	4.361	4.347	0.014
Fold09	4.911	4.855	0.057
Fold10	4.442	4.424	0.018

- Which folds were hardest to predict? Which were easiest?
- On which folds did model 1 perform better?
- Which model did better overall?

Conclusions

- To get the CV-estimated RMSE, we average model RMSE across all 10 splits:

Conclusions

- To get the CV-estimated RMSE, we average model RMSE across all 10 splits:

CV_RMSE_mod1	CV_RMSE_mod2
4.613	4.594

- So Model 2 (with `NumCyl`) indeed outperforms Model 1 (but only slightly)

Conclusions

- To get the CV-estimated RMSE, we average model RMSE across all 10 splits:

CV_RMSE_mod1	CV_RMSE_mod2
4.613	4.594

- So Model 2 (with `NumCyl`) indeed outperforms Model 1 (but only slightly)
- Which model should we use?

Conclusions

- To get the CV-estimated RMSE, we average model RMSE across all 10 splits:

CV_RMSE_mod1	CV_RMSE_mod2
4.613	4.594

- So Model 2 (with `NumCyl`) indeed outperforms Model 1 (but only slightly)
- Which model should we use?
  - If we are interested in obtaining the absolute best predictive performance, we might decide to use model 1.

Conclusions

- To get the CV-estimated RMSE, we average model RMSE across all 10 splits:

CV_RMSE_mod1	CV_RMSE_mod2
4.613	4.594

- So Model 2 (with `NumCyl`) indeed outperforms Model 1 (but only slightly)
- Which model should we use?
  - If we are interested in obtaining the absolute best predictive performance, we might decide to use model 1.
  - But the difference in accuracy between the two models was very slight. And model 1 is simpler than model 2, making it easier to interpret. We might elect to use Model 1

Conclusions

- To get the CV-estimated RMSE, we average model RMSE across all 10 splits:

CV_RMSE_mod1	CV_RMSE_mod2
4.613	4.594

- So Model 2 (with `NumCyl`) indeed outperforms Model 1 (but only slightly)

- Which model should we use?
  - If we are interested in obtaining the absolute best predictive performance, we might decide to use model 1.
  - But the difference in accuracy between the two models was very slight. And model 1 is simpler than model 2, making it easier to interpret. We might elect to use Model 1

- Once we decide on a model type to use, we should go back and refit the chosen model **on the entire data set**

```
best_mod <- lm(FE ~ EngDispl + NumCyl, data = cars2010)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	51.3541	0.4593	111.8138	0.0000
EngDispl	-3.7454	0.2507	-14.9409	0.0000
NumCyl	-0.5880	0.1722	-3.4139	0.0007

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged
  - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged
  - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)
- But LOOCV has significant drawbacks:

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged
  - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)
- But LOOCV has significant drawbacks:
  - Because it requires fitting n models, LOOCV is computationally intensive

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged
  - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)
- But LOOCV has significant drawbacks:
  - Because it requires fitting n models, LOOCV is computationally intensive
  - As any two models fit using LOOCV differ with respect to only two observations, the model estimates for different folds are very highly correlated.

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
    - The model is fit on all but one observation, and then tested on the lone observation.
    - Process is repeated so that every observation is used as test point. The results are averaged
    - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)
- But LOOCV has significant drawbacks:
    - Because it requires fitting n models, LOOCV is computationally intensive
    - As any two models fit using LOOCV differ with respect to only two observations, the model estimates for different folds are very highly correlated.
    - As only one point is used in each validation set, RMSE estimates are highly variable between folds.

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged
  - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)
- But LOOCV has significant drawbacks:
  - Because it requires fitting n models, LOOCV is computationally intensive
  - As any two models fit using LOOCV differ with respect to only two observations, the model estimates for different folds are very highly correlated.
  - As only one point is used in each validation set, RMSE estimates are highly variable between folds.
  - LOOCV does not consistently have higher variance or lower bias than k-fold CV. But it tends to produce RMSE estimates that are less accurate than other techniques.

LOOCV

- The special case when n folds are used on a data set with n observations is called **Leave One Out Cross-Validation** (LOOCV)
  - The model is fit on all but one observation, and then tested on the lone observation.
  - Process is repeated so that every observation is used as test point. The results are averaged
  - Because every possible model is fit, LOOCV estimates are a deterministic function of training set (unlike other CV)
- But LOOCV has significant drawbacks:
  - Because it requires fitting n models, LOOCV is computationally intensive
  - As any two models fit using LOOCV differ with respect to only two observations, the model estimates for different folds are very highly correlated.
  - As only one point is used in each validation set, RMSE estimates are highly variable between folds.
  - LOOCV does not consistently have higher variance or lower bias than k-fold CV. But it tends to produce RMSE estimates that are less accurate than other techniques.
  - LOOCV should rarely be used.

Section 3

The Bootstrap

Why Bootstrap?

So, you want to know how a particular statistic is distributed?

Why Bootstrap?

So, you want to know how a particular statistic is distributed?

- Suppose you are interested in the distribution of slopes $\hat{\beta}_3$ of the interaction term in an MLR model under random sampling:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 \cdot X_2$$

Why Bootstrap?

So, you want to know how a particular statistic is distributed?

- Suppose you are interested in the distribution of slopes $\hat{\beta}_3$ of the interaction term in an MLR model under random sampling:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 \cdot X_2$$

- The classic approach:

Why Bootstrap?

So, you want to know how a particular statistic is distributed?

- Suppose you are interested in the distribution of slopes $\hat{\beta}_3$ of the interaction term in an MLR model under random sampling:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 \cdot X_2$$

- The classic approach:
  - Write the statistic $\hat{\beta}_3$ as a function of the sample observations $x_1, \cdots, x_n$ and use properties of random variables to derive the theoretical distribution for $\hat{b}eta_3$. Make some (often unreasonable) model assumptions

Why Bootstrap?

So, you want to know how a particular statistic is distributed?

- Suppose you are interested in the distribution of slopes $\hat{\beta}_3$ of the interaction term in an MLR model under random sampling:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 \cdot X_2$$

- The classic approach:
  - Write the statistic $\hat{\beta}_3$ as a function of the sample observations $x_1, \cdots, x_n$ and use properties of random variables to derive the theoretical distribution for $\hat{b}eta_3$. Make some (often unreasonable) model assumptions
  - Look up the theoretical distribution based on someone else's attempt to do part (1).

Why Bootstrap?

So, you want to know how a particular statistic is distributed?

- Suppose you are interested in the distribution of slopes $\hat{\beta}_3$ of the interaction term in an MLR model under random sampling:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3 X_1 \cdot X_2$$

- The classic approach:
  - Write the statistic $\hat{\beta}_3$ as a function of the sample observations $x_1, \cdots, x_n$ and use properties of random variables to derive the theoretical distribution for $\hat{b}eta_3$. Make some (often unreasonable) model assumptions
  - Look up the theoretical distribution based on someone else's attempt to do part (1).
  - Hope that the sample size is large enough to allow the Central Limit Theorem to come into play so that the statistic is approximately Normal

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set
  - Plot and summarize the distribution of the statistic.

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set
  - Plot and summarize the distribution of the statistic.
  - The problem?

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set
  - Plot and summarize the distribution of the statistic.
  - The problem?
  - Its unreasonable to assume we can collect a large number of sample sets

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set
  - Plot and summarize the distribution of the statistic.
  - The problem?
  - Its unreasonable to assume we can collect a large number of sample sets
- The bootstrap approach:

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
    - Obtain a large number of sample sets and compute the statistic of interest on each set
    - Plot and summarize the distribution of the statistic.
    - The problem?
    - Its unreasonable to assume we can collect a large number of sample sets
- The bootstrap approach:
    - Assume that your sample is large enough to be "representative' ' of your population.

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set
  - Plot and summarize the distribution of the statistic.
  - The problem?
  - Its unreasonable to assume we can collect a large number of sample sets
- The bootstrap approach:
  - Assume that your sample is large enough to be “representative’ ’ of your population.
  - Create a new bootstrap sample by sampling **with replacement** from your original sample, a number of times equal to your original sample size.

The Resampling Approach

As an alternative to using the theoretical distribution, use simulation to approximate.

- The optimistic approach:
  - Obtain a large number of sample sets and compute the statistic of interest on each set
  - Plot and summarize the distribution of the statistic.
  - The problem?
  - Its unreasonable to assume we can collect a large number of sample sets
- The bootstrap approach:
  - Assume that your sample is large enough to be “representative’ ’ of your population.
  - Create a new bootstrap sample by sampling **with replacement** from your original sample, a number of times equal to your original sample size.
  - Repeat the process to create many bootstrap samples. Compute the statistic of interest on each, plot the results and calculate desired property of the bootstrapped sampling distribution.

Bootstrap Demo

Suppose we have two predictors X_1 and X_2, with quantitative response Y. Moreover,

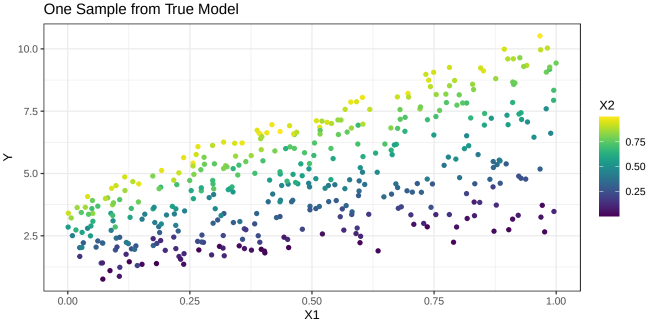
$$Y = 1 + 2 \cdot X_1 + 3 \cdot X_2 + 5 \cdot X_1 \cdot X_2 + \epsilon \qquad \epsilon \sim N(0, \sigma = 0.3)$$

Bootstrap Demo

Suppose we have two predictors X_1 and X_2, with quantitative response Y. Moreover,

$$Y = 1 + 2 \cdot X_1 + 3 \cdot X_2 + 5 \cdot X_1 \cdot X_2 + \epsilon \qquad \epsilon \sim N(0, \sigma = 0.3)$$

- Suppose we have one sample of 400 observations from this true model:

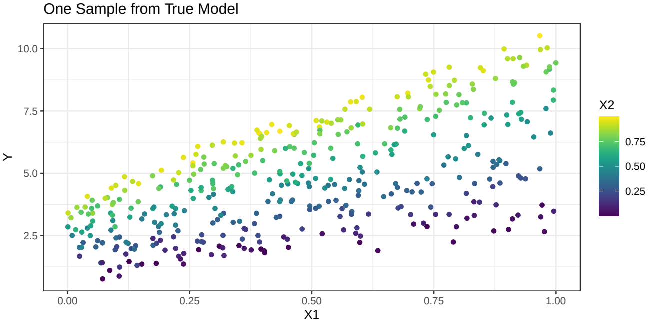


Bootstrap Demo

Suppose we have two predictors X_1 and X_2, with quantitative response Y. Moreover,

$$Y = 1 + 2 \cdot X_1 + 3 \cdot X_2 + 5 \cdot X_1 \cdot X_2 + \epsilon \qquad \epsilon \sim N(0, \sigma = 0.3)$$

- Suppose we have one sample of 400 observations from this true model:



- Note the interaction effect on the plot.

Model Estimates

Below are the model coefficient estimates from our sample of 400 points:

Model Estimates

Below are the model coefficient estimates from our sample of 400 points:

```
my_mod<-lm(Y ~ X1*X2, data = d)
summary(my_mod)$coefficients
```

```
##              Estimate Std. Error  t value      Pr(>|t|)
## (Intercept) 0.9322897 0.05707249 16.33519  3.189598e-46
## X1          2.0704025 0.10192437 20.31313  2.279221e-63
## X2          3.0463411 0.09602184 31.72550 8.528186e-111
## X1:X2       4.9197380 0.17085912 28.79412  3.738267e-99
```

```
##         RSE
## 1 0.2824672
```

Model Estimates

Below are the model coefficient estimates from our sample of 400 points:

```
my_mod<-lm(Y ~ X1*X2, data = d)
summary(my_mod)$coefficients
```

```
##              Estimate Std. Error  t value      Pr(>|t|)
## (Intercept) 0.9322897 0.05707249 16.33519  3.189598e-46
## X1          2.0704025 0.10192437 20.31313  2.279221e-63
## X2          3.0463411 0.09602184 31.72550 8.528186e-111
## X1:X2       4.9197380 0.17085912 28.79412  3.738267e-99
```

```
##         RSE
## 1 0.2824672
```

- For reference, the true model was

$$Y = Y = 1 + 2 \cdot X_1 + 3 \cdot X_2 + 5 \cdot X_1 \cdot X_2 + \epsilon \qquad \epsilon \sim N(0, \sigma = 0.3)$$

The Simulation Approach

To understand the distribution of $\hat{\beta}_3$ due to random sampling...

The Simulation Approach

To understand the distribution of $\hat{\beta}_3$ due to random sampling. . .

1. Propose a specific true model

The Simulation Approach

To understand the distribution of $\hat{\beta}_3$ due to random sampling. . .

1. Propose a specific true model
2. Simulate 1000 sets of sample data from the model, each of size 400.

The Simulation Approach

To understand the distribution of $\hat{\beta}_3$ due to random sampling. . .

1. Propose a specific true model
2. Simulate 1000 sets of sample data from the model, each of size 400.
3. For each simulated sample, fit the linear model with interaction, and record the slope on the interaction term.

The Simulation Approach

To understand the distribution of $\hat{\beta}_3$ due to random sampling...

1. Propose a specific true model
2. Simulate 1000 sets of sample data from the model, each of size 400.
3. For each simulated sample, fit the linear model with interaction, and record the slope on the interaction term.

```
## # A tibble: 1,000 x 1
##    simulated_slope
##              <dbl>
##  1            5.24
##  2            5.06
##  3            5.23
##  4            5.32
##  5            5.42
##  6            4.98
##  7            5.06
##  8            4.81
##  9            4.95
## 10            5.10
## # i 990 more rows
```

The Simulation Approach

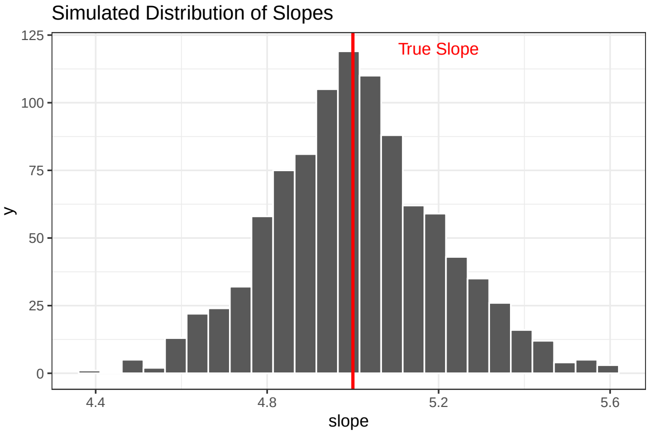
To understand the distribution of $\hat{\beta}_3$ due to random sampling...

1. Propose a specific true model
2. Simulate 1000 sets of sample data from the model, each of size 400.
3. For each simulated sample, fit the linear model with interaction, and record the slope on the interaction term.

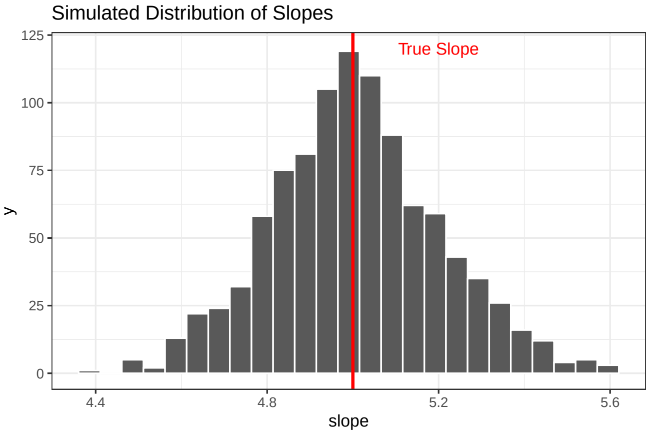
```
## # A tibble: 1,000 x 1
##    simulated_slope
##              <dbl>
##  1            5.24
##  2            5.06
##  3            5.23
##  4            5.32
##  5            5.42
##  6            4.98
##  7            5.06
##  8            4.81
##  9            4.95
## 10            5.10
## # i 990 more rows
```

4. Plot the collection of simulated slopes:

Simulation Distribution



Simulation Distribution



5. Calculate relevant statistics from the simulation distribution

```
##   true_slope mean_slope  sd_slope
## 1          5   5.006795 0.1968506
```

The Bootstrap Approach

Instead of proposing a (likely false) model and generating data from it, we can use the 1 sample we do have:

The Bootstrap Approach

Instead of proposing a (likely false) model and generating data from it, we can use the 1 sample we do have:

```
##          X1        X2        Y
## 1 0.1903066 0.6712289 4.448777
## 2 0.9108393 0.6409498 7.849781
## 3 0.2277161 0.1087580 1.670640
## 4 0.8249905 0.6546378 7.228559
## 5 0.9155760 0.7840531 8.631492
## 6 0.5052083 0.7231319 6.224075
```

The Bootstrap Approach

Instead of proposing a (likely false) model and generating data from it, we can use the 1 sample we do have:

```
##          X1        X2        Y
## 1 0.1903066 0.6712289 4.448777
## 2 0.9108393 0.6409498 7.849781
## 3 0.2277161 0.1087580 1.670640
## 4 0.8249905 0.6546378 7.228559
## 5 0.9155760 0.7840531 8.631492
## 6 0.5052083 0.7231319 6.224075
```

We can create a bootstrap sample:

```
set.seed(135)
a_bootstrap_sample <- slice_sample(original_sample, n = 400, replace = T)
```

The Bootstrap Approach

Instead of proposing a (likely false) model and generating data from it, we can use the 1 sample we do have:

```
##          X1        X2        Y
## 1 0.1903066 0.6712289 4.448777
## 2 0.9108393 0.6409498 7.849781
## 3 0.2277161 0.1087580 1.670640
## 4 0.8249905 0.6546378 7.228559
## 5 0.9155760 0.7840531 8.631492
## 6 0.5052083 0.7231319 6.224075
```

We can create a bootstrap sample:

```
set.seed(135)
a_bootstrap_sample <- slice_sample(original_sample, n = 400, replace = T)
```

Will the bootstrap sample contain duplicate observations? Yes.

The Bootstrap Approach

Instead of proposing a (likely false) model and generating data from it, we can use the 1 sample we do have:

```
##          X1        X2        Y
## 1 0.1903066 0.6712289 4.448777
## 2 0.9108393 0.6409498 7.849781
## 3 0.2277161 0.1087580 1.670640
## 4 0.8249905 0.6546378 7.228559
## 5 0.9155760 0.7840531 8.631492
## 6 0.5052083 0.7231319 6.224075
```

We can create a bootstrap sample:

```
set.seed(135)
a_bootstrap_sample <- slice_sample(original_sample, n = 400, replace = T)
```

Will the bootstrap sample contain duplicate observations? Yes.

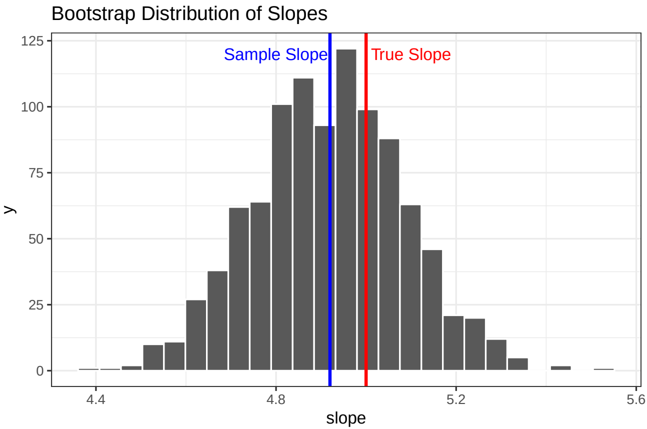
```
##   number_of_uniques prop_of_original
## 1               246            0.615
```

The Bootstrap Approach, cont'd

Now, we create 1000 bootstrap samples (each of size 400) and calculate the slope of each.

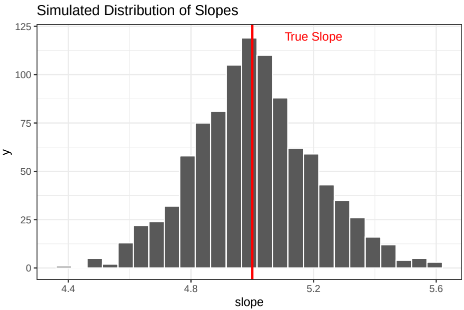
The Bootstrap Approach, cont'd

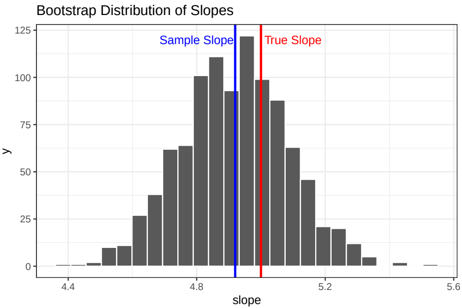
Now, we create 1000 bootstrap samples (each of size 400) and calculate the slope of each.



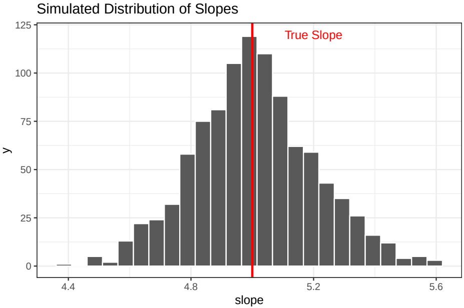

```
##   true_slope mean_slope  sd_slope
## 1          5   4.917985 0.1667698
```

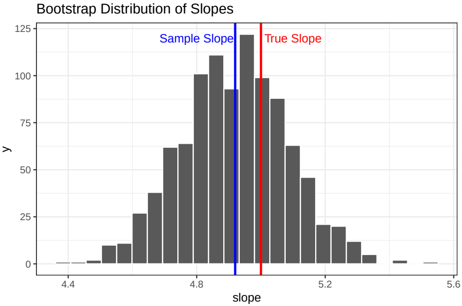
Side-by-Side Comparison





Side-by-Side Comparison





- We compare features of the simulated distribution and the bootstrap distribution:

```
## # A tibble: 2 x 5
##   method mean_slope sd_slope q.025 q.975
##   <chr>       <dbl>    <dbl> <dbl> <dbl>
## 1 boot         4.92    0.167  4.60  5.24
## 2 sim          5.01    0.197  4.62  5.41
```

CV verus Bootstrapping

Both are computationally intensive methods that involve sampling from your data set to learn more about your estimate/model.

CV verus Bootstrapping

Both are computationally intensive methods that involve sampling from your data set to learn more about your estimate/model.

Cross-validation: Often used for *model assessment* and *model selection*.

CV verus Bootstrapping

Both are computationally intensive methods that involve sampling from your data set to learn more about your estimate/model.

Cross-validation: Often used for *model assessment* and *model selection*.

- Partition data into test and train
- Fit model to train, predict on test
- Iterate though all possible *folds*
- Compute aggregate measure of predictive ability

CV verus Bootstrapping

Both are computationally intensive methods that involve sampling from your data set to learn more about your estimate/model.

Cross-validation: Often used for *model assessment* and *model selection*.

- Partition data into test and train
- Fit model to train, predict on test
- Iterate though all possible *folds*
- Compute aggregate measure of predictive ability

Bootstrapping: Often used for *quantifying uncertainty*.

CV verus Bootstrapping

Both are computationally intensive methods that involve sampling from your data set to learn more about your estimate/model.

Cross-validation: Often used for *model assessment* and *model selection*.

- Partition data into test and train
- Fit model to train, predict on test
- Iterate though all possible *folds*
- Compute aggregate measure of predictive ability

Bootstrapping: Often used for *quantifying uncertainty*.

- Draw a bootstrap sample of size n from your data *with replacement*.
- Compute estimate of interest
- Consider distribution of bootstrap estimates over many samples