Subset Selection

Nate Wells

Math 243: Stat Learning

October 21st, 2020

Outline

In today's class, we will...

- Investigate algorithms for selecting good subsets of predictors
- Discuss penalized regression as an alternate method of model selection

Section 1

Subset Selection

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

- Cross-validation: Computationally expensive

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

- Cross-validation: Computationally expensive

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

- Cross-validation: Computationally expensive
- Adjusted R^2: Penalizes non-helpful predictors, but may overestimate test error rate.

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

- Cross-validation: Computationally expensive
- Adjusted R^2: Penalizes non-helpful predictors, but may overestimate test error rate.
- C_p: penalizes training RSS by typical discrepancy between test and training.

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2)$$

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

- Cross-validation: Computationally expensive
- Adjusted R^2: Penalizes non-helpful predictors, but may overestimate test error rate.
- C_p: penalizes training RSS by typical discrepancy between test and training.

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2)$$

- Akaike information criterion (AIC): uses method of maximum likelihood, assuming Normal errors

$$\mathrm{AIC} = \frac{1}{n\hat{\sigma}^2}(\mathrm{RSS} + 2d\hat{\sigma}^2)$$

Methodology

Suppose we wish to find a linear model for Y with p predictors $X_1, \ldots, X_p$.

How do we determine the optimal collection of predictors?

Determine an appropriate selection criteria.

- Cross-validation: Computationally expensive
- Adjusted R^2: Penalizes non-helpful predictors, but may overestimate test error rate.
- C_p: penalizes training RSS by typical discrepancy between test and training.

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2)$$

- Akaike information criterion (AIC): uses method of maximum likelihood, assuming Normal errors

$$\mathrm{AIC} = \frac{1}{n\hat{\sigma}^2}(\mathrm{RSS} + 2d\hat{\sigma}^2)$$

- Bayesian information criterion (BIC): uses method of maximum likelihood and Bayes' Rule

$$\mathrm{BIC} = \frac{1}{n\hat{\sigma}^2}(\mathrm{RSS} + \ln n d\hat{\sigma}^2)$$

Best Subset

With p predictors, there are a total of 2^p possible MLR models.

- There are $\binom{p}{k}$ models using exactly k of p predictors

Best Subset

With p predictors, there are a total of 2^p possible MLR models.

- There are $\binom{p}{k}$ models using exactly k of p predictors

Theoretically, we can find the best model by fitting each possible model and selecting the best via appropriate selection criteria (C_p, AIC, BIC, R^2, CV)

Best Subset

With p predictors, there are a total of 2^p possible MLR models.

- There are $\binom{p}{k}$ models using exactly k of p predictors

Theoretically, we can find the best model by fitting each possible model and selecting the best via appropriate selection criteria (C_p, AIC, BIC, R^2, CV)

Downsides?

Best Subset

With p predictors, there are a total of 2^p possible MLR models.

- There are $\binom{p}{k}$ models using exactly k of p predictors

Theoretically, we can find the best model by fitting each possible model and selecting the best via appropriate selection criteria (C_p, AIC, BIC, R^2, CV)

Downsides?

- Computation time and storage grows exponentially in p

Best Subset

With p predictors, there are a total of 2^p possible MLR models.

- There are $\binom{p}{k}$ models using exactly k of p predictors

Theoretically, we can find the best model by fitting each possible model and selecting the best via appropriate selection criteria (C_p, AIC, BIC, R^2, CV)

Downsides?

- Computation time and storage grows exponentially in p
- May have low marginal improvement despite number of models fitted

Best Subset in R

We use the `regsubsets` function in the `leaps` library.

Best Subset in R

We use the `regsubsets` function in the `leaps` library.

- `regsubsets` uses the same syntax as `lm`. The `summary` function outputs the best set of variables for the given number of predictors

Best Subset in R

We use the `regsubsets` function in the `leaps` library.

- `regsubsets` uses the same syntax as `lm`. The `summary` function outputs the best set of variables for the given number of predictors
- Be default, `regsubsets` only returns up to the best eight models. But `nvmax` can be used to return as many variables as desired

Best Subset in R

We use the `regsubsets` function in the `leaps` library.

- `regsubsets` uses the same syntax as `lm`. The `summary` function outputs the best set of variables for the given number of predictors
- Be default, `regsubsets` only returns up to the best eight models. But `nvmax` can be used to return as many variables as desired
- Best is determined by *RSS*.

Best Subset in R

We use the `regsubsets` function in the `leaps` library.

- `regsubsets` uses the same syntax as `lm`. The `summary` function outputs the best set of variables for the given number of predictors
- Be default, `regsubsets` only returns up to the best eight models. But `nvmax` can be used to return as many variables as desired
- Best is determined by *RSS*.

```
library(palmerpenguins)
library(leaps)
penguins<-penguins %>% drop_na()

best_subset<-regsubsets(body_mass_g ~. , data = penguins, nvmax = 8)
```

Summary of regsubsets

- Stars indicate variable is included in model

```
## Subset selection object
## Call: regsubsets.formula(body_mass_g ~ ., data = penguins, nvmax = 8)
## 9 Variables  (and intercept)
##                   Forced in Forced out
## speciesChinstrap      FALSE      FALSE
## speciesGentoo         FALSE      FALSE
## islandDream           FALSE      FALSE
## islandTorgersen       FALSE      FALSE
## bill_length_mm        FALSE      FALSE
## bill_depth_mm         FALSE      FALSE
## flipper_length_mm     FALSE      FALSE
## sexmale               FALSE      FALSE
## year                  FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          speciesChinstrap speciesGentoo islandDream islandTorgersen
## 1  ( 1 ) " "              " "           " "         " "
## 2  ( 1 ) " "              "*"           " "         " "
## 3  ( 1 ) " "              "*"           " "         " "
## 4  ( 1 ) " "              "*"           " "         " "
## 5  ( 1 ) "*"              "*"           " "         " "
## 6  ( 1 ) "*"              "*"           " "         " "
## 7  ( 1 ) "*"              "*"           " "         " "
## 8  ( 1 ) "*"              "*"           " "         "*"
##          bill_length_mm bill_depth_mm flipper_length_mm sexmale year
## 1  ( 1 ) " "            " "           "*"                " "     " "
## 2  ( 1 ) " "            " "           " "                "*"     " "
## 3  ( 1 ) " "            " "           "*"                "*"     " "
## 4  ( 1 ) " "            "*"           "*"                "*"     " "
## 5  ( 1 ) " "            "*"           "*"                "*"     " "
## 6  ( 1 ) "*"            "*"           "*"                "*"     " "
## 7  ( 1 ) "*"            "*"           "*"                "*"     "*"
## 8  ( 1 ) "*"            "*"           "*"                "*"     "*"
```

Other Selection Metrics

The summary function can return selection metrics for each model.

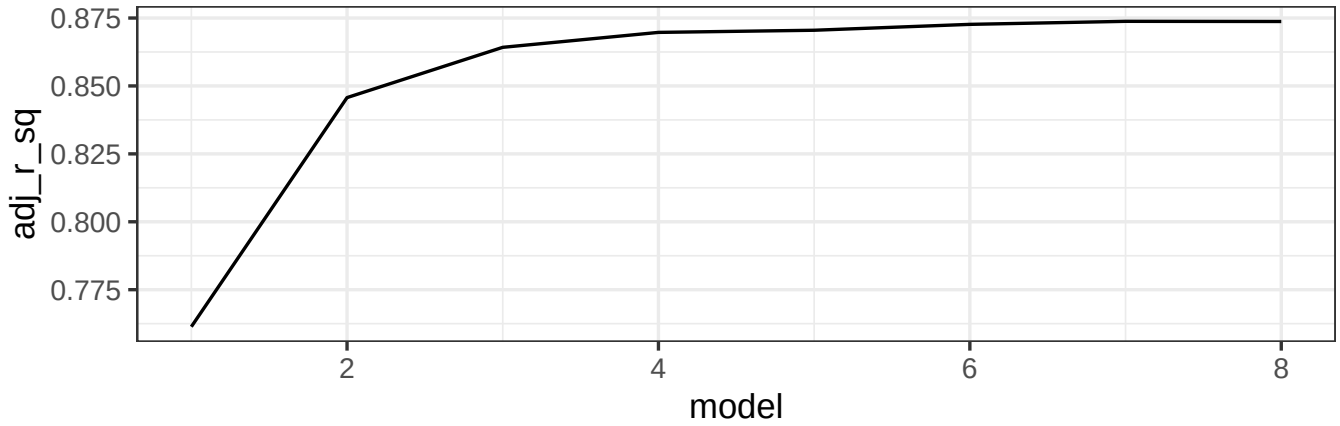
```
adj_r_sq<-summary(best_subset)$adjr2
rss<-summary(best_subset)$rss
cp<-summary(best_subset)$cp

d<-data.frame(model = 1:8, adj_r_sq, rss, cp )
d
```

```
##   model  adj_r_sq      rss         cp
## 1     1 0.7613734 51211963 294.805584
## 2     2 0.8457078 33012815  75.124367
## 3     3 0.8642104 28965893  27.829395
## 4     4 0.8697020 27709979  14.531285
## 5     5 0.8704945 27457472  13.455534
## 6     6 0.8726606 26915647   8.855638
## 7     7 0.8737834 26596486   6.967990
## 8     8 0.8737208 26527820   8.131576
```

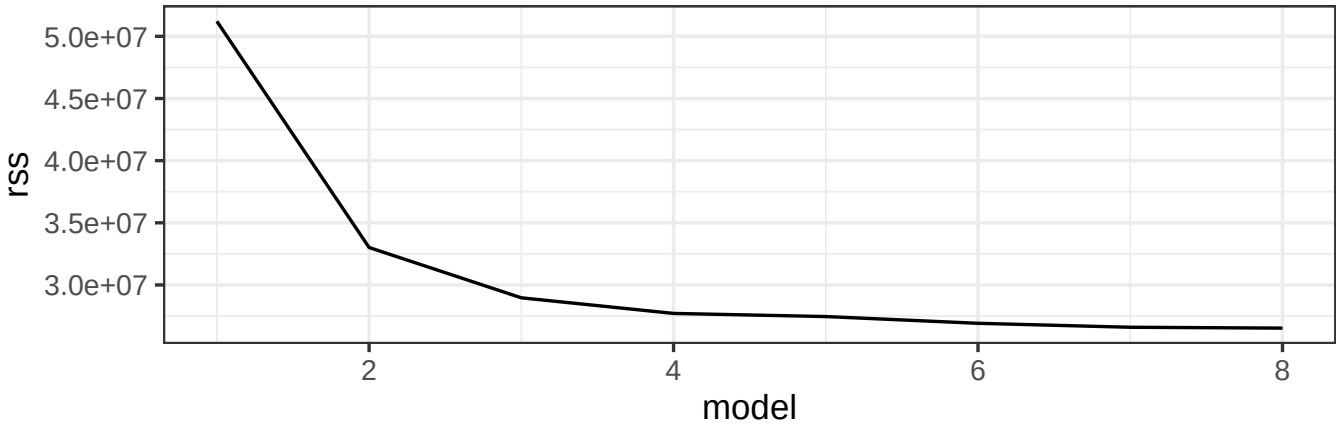
Plotting

We can use ggplot2 to visualize selection metric as a function of variable number

```
ggplot(d, aes(x = model, y = adj_r_sq))+geom_line()+theme_bw()
```




```
ggplot(d, aes(x = model, y = rss))+geom_line()+theme_bw()
```



Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p - 1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p-1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.
- Compared to Best Subset, forward selection computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p - 1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.
- Compared to Best Subset, forward selection computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Forward selection tends to favor parsimonous models

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p-1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.
- Compared to Best Subset, forward selection computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Forward selection tends to favor parsimonous models
- Downsides?

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p-1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.
- Compared to Best Subset, forward selection computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Forward selection tends to favor parsimonous models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p-1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.
- Compared to Best Subset, forward selection computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Forward selection tends to favor parsimonous models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)
  - Early predictors may become redundant

Forward Selection

Forward selection is a *computationally efficient* alternative to best subset

- To perform forward selection, create the best 1 variable model. Then create $p-1$ new 2 variable models by adding each other predictor one-at-a-time to the existing 1-variable model. Repeat for 3 variables and so on.
- Compared to Best Subset, forward selection computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Forward selection tends to favor parsimonous models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)
  - Early predictors may become redundant
  - Can be unstable

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Backward elimination tends to favor in-depth models

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Backward elimination tends to favor in-depth models
- Downsides?

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Backward elimination tends to favor in-depth models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Backward elimination tends to favor in-depth models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)
  - Requires fewer predictors than observations

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Backward elimination tends to favor in-depth models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)
  - Requires fewer predictors than observations
  - Susceptible to multicollinearity

Backward Elimination

Backward Elimination is another *computationally efficient* alternative to best subset

- To perform backward selection, begin with full model. Then create $p-1$ new $p-1$ variable models by removing one-at-a-time each other predictor from the existing p-variable model. Repeat for $p-2$ variables and so on.
- Compared to Best Subset, backward elimination computation time grows polynomially in p: $\text{Num. Models} = 1 + \frac{p(p+1)}{2}$
- Backward elimination tends to favor in-depth models
- Downsides?
  - Not guaranteed to find the best model (or even something close to the best model)
  - Requires fewer predictors than observations
  - Susceptible to multicollinearity
  - Can be unstable

Forward/Backward Selection in R

We again use the `regsubsets` function in the `leaps` library.

```
forward_select<-regsubsets(body_mass_g ~. , data = penguins, nvmax = 8,
                           method = "forward")

backward_elim<-regsubsets(body_mass_g ~. , data = penguins, nvmax = 8,
                          method = "backward")
```

Summary of Forward Selection

```
summary(forward_select)
```

```
## Subset selection object
## Call: regsubsets.formula(body_mass_g ~ ., data = penguins, nvmax = 8,
##     method = "forward")
## 9 Variables  (and intercept)
##                   Forced in Forced out
## speciesChinstrap      FALSE      FALSE
## speciesGentoo         FALSE      FALSE
## islandDream           FALSE      FALSE
## islandTorgersen       FALSE      FALSE
## bill_length_mm        FALSE      FALSE
## bill_depth_mm         FALSE      FALSE
## flipper_length_mm     FALSE      FALSE
## sexmale               FALSE      FALSE
## year                  FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: forward
##          speciesChinstrap speciesGentoo islandDream islandTorgersen
## 1  ( 1 ) " "              " "           " "         " "
## 2  ( 1 ) " "              " "           " "         " "
## 3  ( 1 ) " "              "*"           " "         " "
## 4  ( 1 ) " "              "*"           " "         " "
## 5  ( 1 ) "*"              "*"           " "         " "
## 6  ( 1 ) "*"              "*"           " "         " "
## 7  ( 1 ) "*"              "*"           " "         " "
## 8  ( 1 ) "*"              "*"           " "         "*"
##          bill_length_mm bill_depth_mm flipper_length_mm sexmale year
## 1  ( 1 ) " "            " "           "*"                " "     " "
## 2  ( 1 ) " "            " "           "*"                "*"     " "
## 3  ( 1 ) " "            " "           "*"                "*"     " "
## 4  ( 1 ) " "            "*"           "*"                "*"     " "
## 5  ( 1 ) " "            "*"           "*"                "*"     " "
## 6  ( 1 ) "*"            "*"           "*"                "*"     " "
## 7  ( 1 ) "*"            "*"           "*"                "*"     "*"
## 8  ( 1 ) "*"            "*"           "*"                "*"     "*"
```

Summary of Backward Elimination

```
summary(backward_elim)
```

```
## Subset selection object
## Call: regsubsets.formula(body_mass_g ~ ., data = penguins, nvmax = 8,
##     method = "backward")
## 9 Variables  (and intercept)
##                   Forced in Forced out
## speciesChinstrap      FALSE      FALSE
## speciesGentoo         FALSE      FALSE
## islandDream           FALSE      FALSE
## islandTorgersen       FALSE      FALSE
## bill_length_mm        FALSE      FALSE
## bill_depth_mm         FALSE      FALSE
## flipper_length_mm     FALSE      FALSE
## sexmale               FALSE      FALSE
## year                  FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: backward
##          speciesChinstrap speciesGentoo islandDream islandTorgersen
## 1  ( 1 ) " "              "*"           " "         " "
## 2  ( 1 ) " "              "*"           " "         " "
## 3  ( 1 ) " "              "*"           " "         " "
## 4  ( 1 ) " "              "*"           " "         " "
## 5  ( 1 ) "*"              "*"           " "         " "
## 6  ( 1 ) "*"              "*"           " "         " "
## 7  ( 1 ) "*"              "*"           " "         " "
## 8  ( 1 ) "*"              "*"           " "         "*"
##          bill_length_mm bill_depth_mm flipper_length_mm sexmale year
## 1  ( 1 ) " "            " "           " "                " "     " "
## 2  ( 1 ) " "            " "           " "                "*"     " "
## 3  ( 1 ) " "            " "           "*"                "*"     " "
## 4  ( 1 ) " "            "*"           "*"                "*"     " "
## 5  ( 1 ) " "            "*"           "*"                "*"     " "
## 6  ( 1 ) "*"            "*"           "*"                "*"     " "
## 7  ( 1 ) "*"            "*"           "*"                "*"     "*"
## 8  ( 1 ) "*"            "*"           "*"                "*"     "*"
```

Section 2

Penalized Regression

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Generally, under what circumstances will the **full model** perform well, compared to other subset models? *(Think about the Bias-Variance Tradeoff)*

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Generally, under what circumstances will the **full model** perform well, compared to other subset models? *(Think about the Bias-Variance Tradeoff)*

- If irreducible error is small ($\epsilon \sim N(0, \sigma^2)$ with $\sigma \approx 0$)

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Generally, under what circumstances will the **full model** perform well, compared to other subset models? *(Think about the Bias-Variance Tradeoff)*

- If irreducible error is small ($\epsilon \sim N(0, \sigma^2)$ with $\sigma \approx 0$)
- If model variance is high and model bias is low.

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Generally, under what circumstances will the **full model** perform well, compared to other subset models? *(Think about the Bias-Variance Tradeoff)*

- If irreducible error is small ($\epsilon \sim N(0, \sigma^2)$ with $\sigma \approx 0$)
- If model variance is high and model bias is low.

Suppose $\hat{y} = 10 + 0.01x_2 + 1000x_2$ is the best fitting linear model using X_1 and X_2.

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Generally, under what circumstances will the **full model** perform well, compared to other subset models? *(Think about the Bias-Variance Tradeoff)*

- If irreducible error is small ($\epsilon \sim N(0, \sigma^2)$ with $\sigma \approx 0$)
- If model variance is high and model bias is low.

Suppose $\hat{y} = 10 + 0.01x_2 + 1000x_2$ is the best fitting linear model using X_1 and X_2.

How might the bias and variance of the following model compare?

$$\hat{y} = 10 + 0.01x_2 + 500x_2$$

Motivation 1

Suppose we wish to build a linear model for Y using predictors $X_1, \ldots, X_p$ using n observations.

Generally, under what circumstances will the **full model** perform well, compared to other subset models? *(Think about the Bias-Variance Tradeoff)*

- If irreducible error is small ($\epsilon \sim N(0, \sigma^2)$ with $\sigma \approx 0$)
- If model variance is high and model bias is low.

Suppose $\hat{y} = 10 + 0.01x_2 + 1000x_2$ is the best fitting linear model using X_1 and X_2.

How might the bias and variance of the following model compare?

$$\hat{y} = 10 + 0.01x_2 + 500x_2$$

When might this new model have lower MSE than the original model?

Motivation 2

Suppose $\hat{y} = 10 + 0.01x_1 + 1000x_2$ is again the best fitting linear model using X_1 and X_2.

Motivation 2

Suppose $\hat{y} = 10 + 0.01x_1 + 1000x_2$ is again the best fitting linear model using X_1 and X_2.

- Are we justified in saying that X_2 is a more important predictor than X_1?

Motivation 2

Suppose $\hat{y} = 10 + 0.01x_1 + 1000x_2$ is again the best fitting linear model using X_1 and X_2.

- Are we justified in saying that X_2 is a more important predictor than X_1?

Suppose we first standardize X_1 and X_2 by subtacting off their means and dividing by their standard deviations:

$$Z_1 = \frac{X_1 - \mu_1}{\sigma_1} \qquad Z_2 = \frac{X_2 - \mu_2}{\sigma_2}$$

Motivation 2

Suppose $\hat{y} = 10 + 0.01x_1 + 1000x_2$ is again the best fitting linear model using X_1 and X_2.

- Are we justified in saying that X_2 is a more important predictor than X_1?

Suppose we first standardize X_1 and X_2 by subtacting off their means and dividing by their standard deviations:

$$Z_1 = \frac{X_1 - \mu_1}{\sigma_1} \qquad Z_2 = \frac{X_2 - \mu_2}{\sigma_2}$$

- If we build a model and find $\hat{y} = 10 + 0.01z_1 + 1000z_2$, where Z_1 and Z_2 are standardized, are we now justified in saying that Z_2 is more important than Z_1?

Motivation 2

Suppose $\hat{y} = 10 + 0.01x_1 + 1000x_2$ is again the best fitting linear model using X_1 and X_2.

- Are we justified in saying that X_2 is a more important predictor than X_1?

Suppose we first standardize X_1 and X_2 by subtacting off their means and dividing by their standard deviations:

$$Z_1 = \frac{X_1 - \mu_1}{\sigma_1} \qquad Z_2 = \frac{X_2 - \mu_2}{\sigma_2}$$

- If we build a model and find $\hat{y} = 10 + 0.01z_1 + 1000z_2$, where Z_1 and Z_2 are standardized, are we now justified in saying that Z_2 is more important than Z_1?
  - How might the variance and bias of the following model compare to the standarized model?

$$\hat{y} = 10 + 0.02z_2 + 500z_1$$

Ridge Regression

Recall that least squares regression estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ for

$$\hat{y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

are obtained by finding the values of β that minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

Ridge Regression

Recall that least squares regression estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ for

$$\hat{y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

are obtained by finding the values of β that minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

To perform **Ridge Regression**, we instead find coefficients β that minimize

$$\text{RSS} + \lambda \sum_{i=1}^{p} \beta_i^2 \qquad \text{where } \lambda \geq 0 \text{ is tuning parameter}$$

Ridge Regression

Recall that least squares regression estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ for

$$\hat{y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

are obtained by finding the values of β that minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

To perform **Ridge Regression**, we instead find coefficients β that minimize

$$\text{RSS} + \lambda \sum_{i=1}^{p} \beta_i^2 \qquad \text{where } \lambda \geq 0 \text{ is tuning parameter}$$

Why?

Ridge Regression

Recall that least squares regression estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ for

$$\hat{y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

are obtained by finding the values of β that minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

To perform **Ridge Regression**, we instead find coefficients β that minimize

$$\text{RSS} + \lambda \sum_{i=1}^{p} \beta_i^2 \qquad \text{where } \lambda \geq 0 \text{ is tuning parameter}$$

Why?

- The term $\lambda \sum_{i=1}^{p} \beta_i^2$ is the **shrinkage penalty**, and is small when the β are small.

Ridge Regression

Recall that least squares regression estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ for

$$\hat{y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

are obtained by finding the values of β that minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

To perform **Ridge Regression**, we instead find coefficients β that minimize

$$\text{RSS} + \lambda \sum_{i=1}^{p} \beta_i^2 \qquad \text{where } \lambda \geq 0 \text{ is tuning parameter}$$

Why?

- The term $\lambda \sum_{i=1}^{p} \beta_i^2$ is the **shrinkage penalty**, and is small when the β are small.
- With a shrinkage penalty, the algorithm prefers models with lower coefficients.

Ridge Regression

Recall that least squares regression estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ for

$$\hat{y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

are obtained by finding the values of β that minimize

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

To perform **Ridge Regression**, we instead find coefficients β that minimize

$$\text{RSS} + \lambda \sum_{i=1}^{p} \beta_i^2 \qquad \text{where } \lambda \geq 0 \text{ is tuning parameter}$$

Why?

- The term $\lambda \sum_{i=1}^{p} \beta_i^2$ is the **shrinkage penalty**, and is small when the β are small.
- With a shrinkage penalty, the algorithm prefers models with lower coefficients.
- This tends to reduce variance, at the cost of increased bias.

Effects of the Tuning Parameter

Goal: Find β which minimize $\mathrm{RSS} + \lambda \sum_{i=1}^{p} \beta_p^2$

Effects of the Tuning Parameter

Goal: Find β which minimize $\mathrm{RSS} + \lambda \sum_{i=1}^{p} \beta_p^2$

What will happen to β_0 as $\lambda \to \infty$? As $\lambda \to 0$?

Effects of the Tuning Parameter

Goal: Find β which minimize $\mathrm{RSS} + \lambda \sum_{i=1}^{p} \beta_p^2$

What will happen to β_0 as $\lambda \to \infty$? As $\lambda \to 0$?

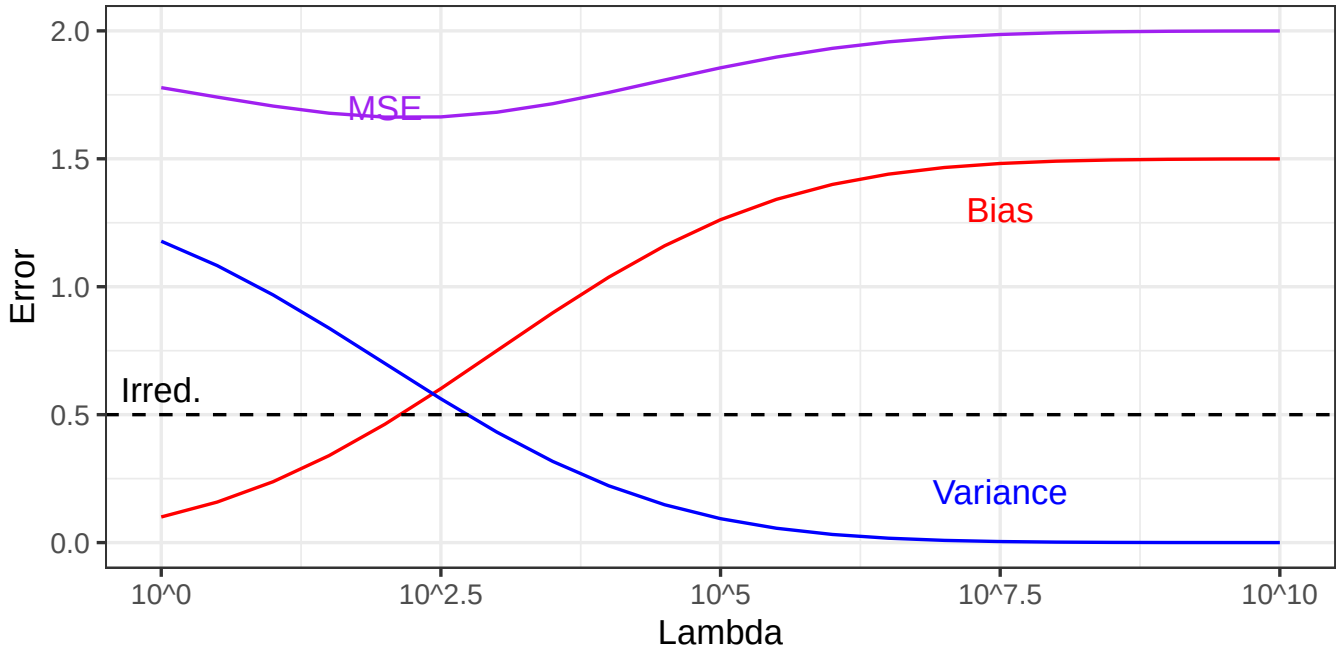
What happens to MSE as $\lambda \to 0$ or $\lambda \to 1$?

Effects of the Tuning Parameter

Goal: Find β which minimize $\mathrm{RSS} + \lambda \sum_{i=1}^{p} \beta_p^2$

What will happen to β_0 as $\lambda \to \infty$? As $\lambda \to 0$?

What happens to MSE as $\lambda \to 0$ or $\lambda \to 1$?



Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

What type of models will ridge regression prefer (in terms of β_1 and β_2)?

Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

What type of models will ridge regression prefer (in terms of β_1 and β_2)?

- Recall the shrinkage penalty is $\lambda \sum_{i=1}^{2} \beta_i^2 = \lambda(1000^2 + 0.01^2)$
- Since $\beta_2 = 1000$ is much larger than $\beta_1 = 0.01$, ridge regression will prioritize reducing β_2 over β_1.

Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

What type of models will ridge regression prefer (in terms of β_1 and β_2)?

- Recall the shrinkage penalty is $\lambda \sum_{i=1}^{2} \beta_i^2 = \lambda(1000^2 + 0.01^2)$
- Since $\beta_2 = 1000$ is much larger than $\beta_1 = 0.01$, ridge regression will prioritize reducing β_2 over β_1.
- Will this actually produce good MSE for a fixed λ?

Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

What type of models will ridge regression prefer (in terms of β_1 and β_2)?

- Recall the shrinkage penalty is $\lambda \sum_{i=1}^{2} \beta_i^2 = \lambda(1000^2 + 0.01^2)$
- Since $\beta_2 = 1000$ is much larger than $\beta_1 = 0.01$, ridge regression will prioritize reducing β_2 over β_1.
- Will this actually produce good MSE for a fixed λ?
  - Only if standard deviation of x_2 is much, much larger than that of x_1

Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

What type of models will ridge regression prefer (in terms of β_1 and β_2)?

- Recall the shrinkage penalty is $\lambda \sum_{i=1}^{2} \beta_i^2 = \lambda(1000^2 + 0.01^2)$
- Since $\beta_2 = 1000$ is much larger than $\beta_1 = 0.01$, ridge regression will prioritize reducing β_2 over β_1.
- Will this actually produce good MSE for a fixed λ?
  - Only if standard deviation of x_2 is much, much larger than that of x_1

Ridge regression is most efficient if predictors are standardarized first.

Standardization

Suppose X_1 and X_2 are non-standardized predictors and the best fitting linear model is

$$\hat{y} = 10 + 0.01x_1 + 1000x_2$$

What type of models will ridge regression prefer (in terms of β_1 and β_2)?

- Recall the shrinkage penalty is $\lambda \sum_{i=1}^{2} \beta_i^2 = \lambda(1000^2 + 0.01^2)$
- Since $\beta_2 = 1000$ is much larger than $\beta_1 = 0.01$, ridge regression will prioritize reducing β_2 over β_1.
- Will this actually produce good MSE for a fixed λ?
  - Only if standard deviation of x_2 is much, much larger than that of x_1

Ridge regression is most efficient if predictors are standardarized first.

$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{\hat{\sigma}_j}$$

Where x_{ij} is the ith observation of the jth predictor, $\bar{x}_j$ is the sample mean of the jth predictor, and $\hat{\sigma}_j$ is the sample st. dev. of the jth predictor.