



# STOR 320 Modeling VII

Lecture 22

Yao Li

Department of Statistics and Operations Research

UNC Chapel Hill

# Final Presentation Time

Nov 12	Nov 17
11:15-11:25AM	11:15-11:25AM
11:25-11:35AM	11:25-11:35AM
11:35-11:45AM	11:35-11:45AM
11:45-11:55AM	11:45-11:55AM
11:55-12:05PM	11:55-12:05PM
12:05-12:15PM	12:05-12:15PM
12:15-12:25PM	12:15-12:25PM
12:25-12:35PM	12:25-12:35PM
12:35-12:45PM	12:35-12:45PM

- Schedule the presentation time before Nov 11.
- Submit slides via Sakai before 11:59PM on Nov 11.
- 5-7 minutes presentation.

# Introduction

- Big Data
  - Large Sample Size
  - Large Number of Variables
  - Traditional Methods are Difficult to Implement
  - Depends on the Available Technology
- Goal: Explore Approaches for Quick Filtering of Predictors
- Tutorial
  - Download Rmd
  - Install Package `> library(glmnet)`
  - Knit the Document
  - Read the Introduction

# Linear Models

- Consider the Following:

$$y_i = \beta_0 + X_{1i}\beta_1 + \dots + X_{pi}\beta_p + \epsilon_i$$

where  $i = 1, 2, 3, \dots, n$

- Matrix Representation

$$\mathbf{y} = \beta_0 + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where  $\mathbf{y} = [y_1, y_2, \dots, y_n]'$ ,

$$\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_p]'$$

$$\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, \dots, \epsilon_n]'$$

and

$$\mathbf{X} = \begin{bmatrix} X_{11} & X_{21} & \dots & X_{p1} \\ X_{12} & X_{22} & \dots & X_{p2} \\ \vdots & \vdots & \ddots & \vdots \\ X_{1n} & X_{2n} & \dots & X_{pn} \end{bmatrix}$$

# Linear Model

- Information About Model Matrix

$$\mathbf{X} = \begin{bmatrix} X_{11} & X_{21} & \cdots & X_{p1} \\ X_{12} & X_{22} & \cdots & X_{p2} \\ \vdots & \vdots & \ddots & \vdots \\ X_{1n} & X_{2n} & \cdots & X_{pn} \end{bmatrix}$$

This Matrix Should Be Standardized

- Once Standardized, The Intercept  $\beta_0$  is Unnecessary in the Model
- For Interpretability, the Response Vector  $\mathbf{y}$  Can Also Be Standardized

# Part 1: Simulate and Mediate

- Run Chunk 1
  - Simulating Response From a Linear Model
  - All Predictor Variables in  $X$  are Standardized
  - What is  $n$ ?
  - What is  $p$ ?
  - What do We Know About the True Signal We Want to Detect?

```
> rnorm()
```



Sparse

# Part 1-Chunk 2

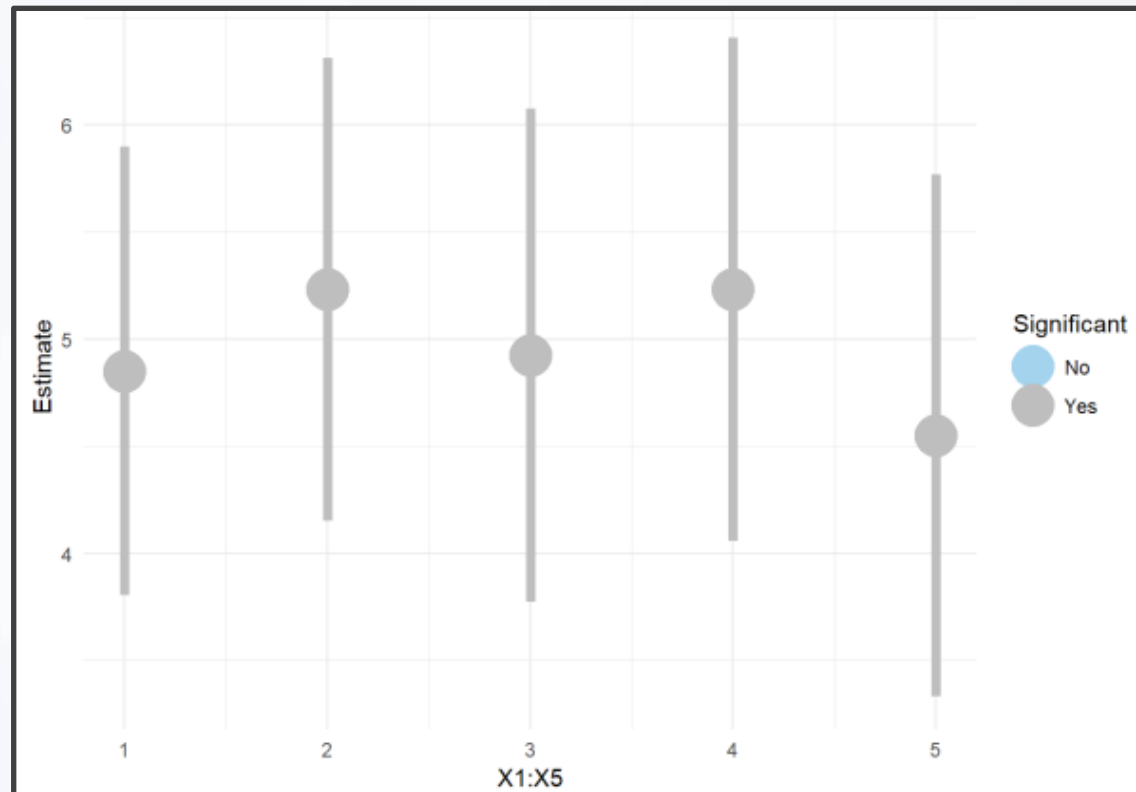
- Run Chunk 2
  - Fitting Naïve Linear Model
  - Obtaining Confidence Intervals for Parameters

```
> confint(lm.model)
```

- Figure Info
  - Show the Estimated Coefficients of Linear Model
  - Show Confidence Intervals for These Coefficients
  - What Does the Color Aesthetic Being Used For?

# Part 1-Chunk 2

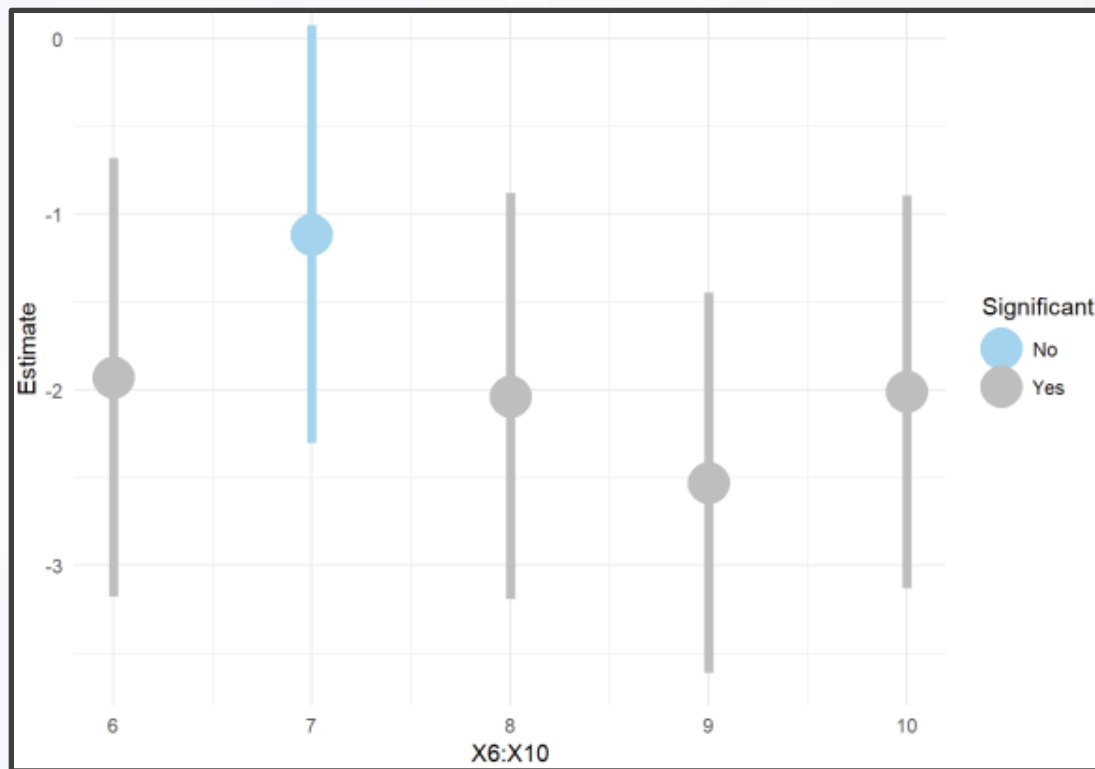
- Chunk 2 (Continued)
  - Knit the Document and Observe the 3 Graphics
  - Figure 1





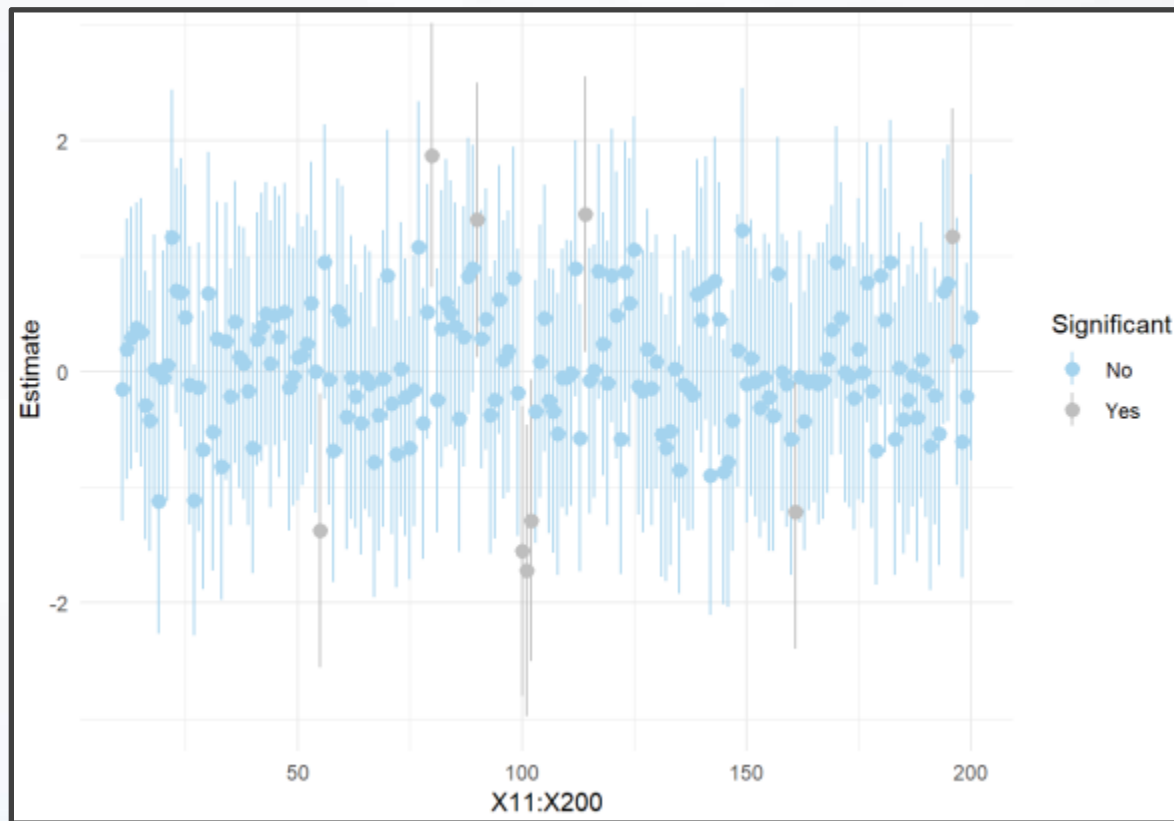
# Part 1-Chunk 2

- Chunk 2 (Continued)
  - Figure 2
  - What is the Problem?



# Part 1-Chunk 2

- Chunk 2 (Continued)
  - Figure 3
  - What is the Problem?



# Part 1-Chunk3

- Run Chunk 3
  - Regression for Each Predictor
- Obtaining Coefficients
- Obtaining P-Values

```
> coef(individual.mod)
(Intercept)      x.200
  0.1257668    -0.3200960
```

Save

```
> summary(individual.mod)
```

Call:

```
lm(formula = y ~ ., data = SIM.DATA[, c(1, j + 1)])
```

Residuals:

Min	1Q	Median	3Q	Max
-47.252	-11.318	0.035	10.759	45.336

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	0.1258	0.7021	0.179	0.858
x.200	-0.3201	0.7230	-0.443	0.658

Save

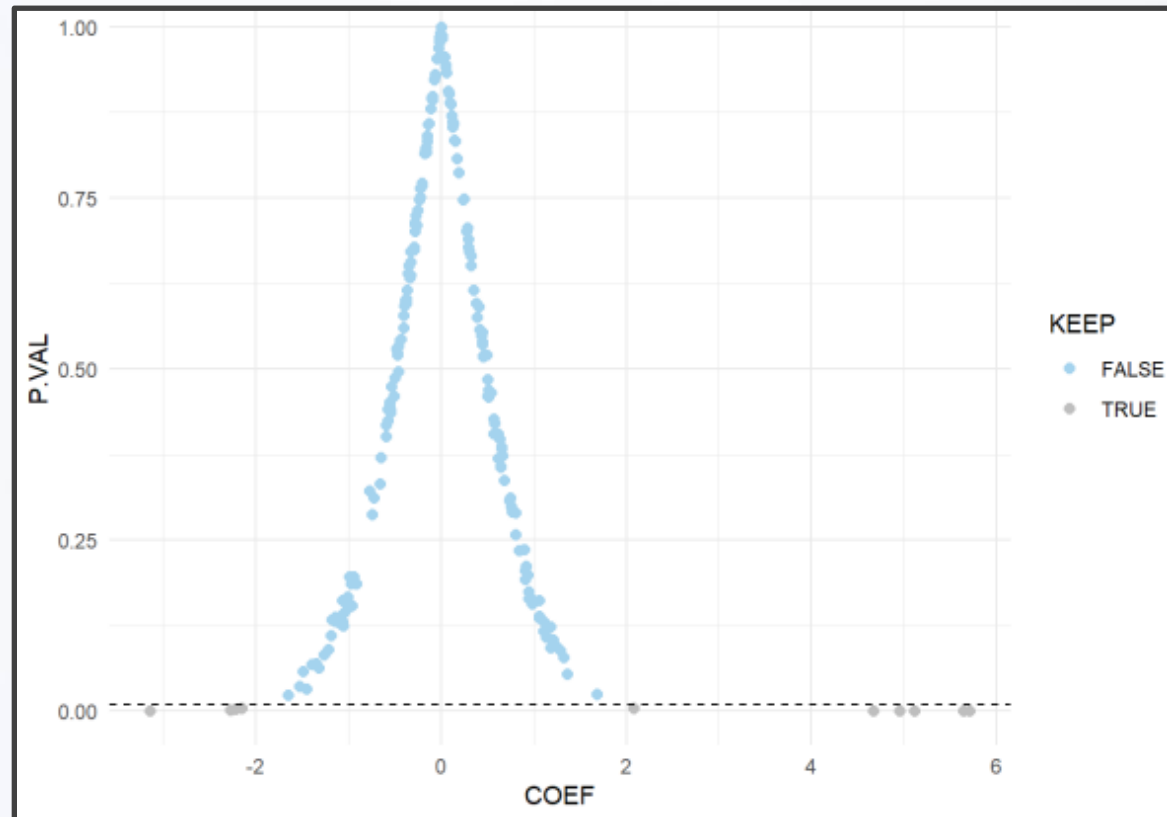
Residual standard error: 15.66 on 498 degrees of freedom

Multiple R-squared: 0.0003934, Adjusted R-squared: -0.001614

F-statistic: 0.196 on 1 and 498 DF, p-value: 0.6582

# Part 1-Chunk3

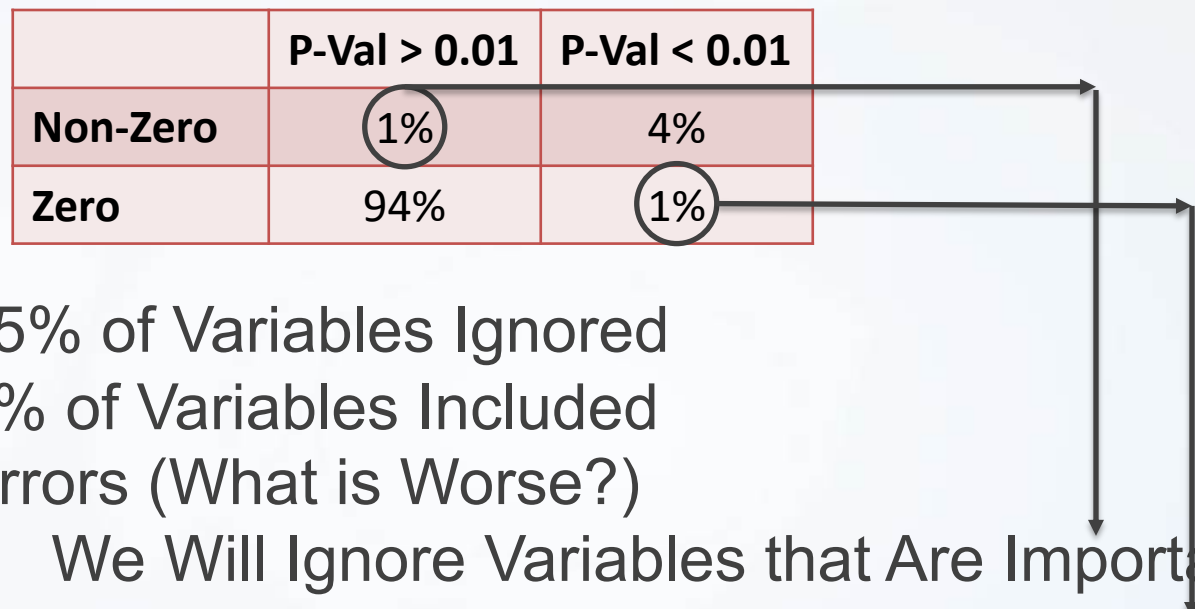
- Run Chunk 3
  - Figure Plots P-Values Against Coefficients



# Part 1-Chunk 3

- Run Chunk 3
  - Suppose We Were to Keep Only the Predictor Variables that Had P-Values < 0.01
  - Observe the Table

	P-Val > 0.01	P-Val < 0.01
Non-Zero	1%	4%
Zero	94%	1%



- 95% of Variables Ignored
- 5% of Variables Included
- Errors (What is Worse?)
  - We Will Ignore Variables that Are Important
  - We Will Include Variables that Are Irrelevant

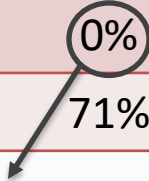
# Part 1-Chunk 4

- Chunk 4
  - Try to Find the Smallest Cutoff Value So That We are Not Missing Important Variables
  - To Ensure We are Not Missing Important Variables, Should we Increase or Decrease the Original Cutoff (0.01)
  - What Cutoff Works?
  - Try Multiple Cutoffs and Observe the Table
  - Run the Code Inside the Chunk Until All 10 Important Variables are Retained for the Future

# Part 1-Chunk 4

- Chunk 4 (Continued)
  - Traditional Choice: 0.20
  - Output in Table

	P-Val > 0.01	P-Val < 0.01
Non-Zero	0%	5%
Zero	71%	24%



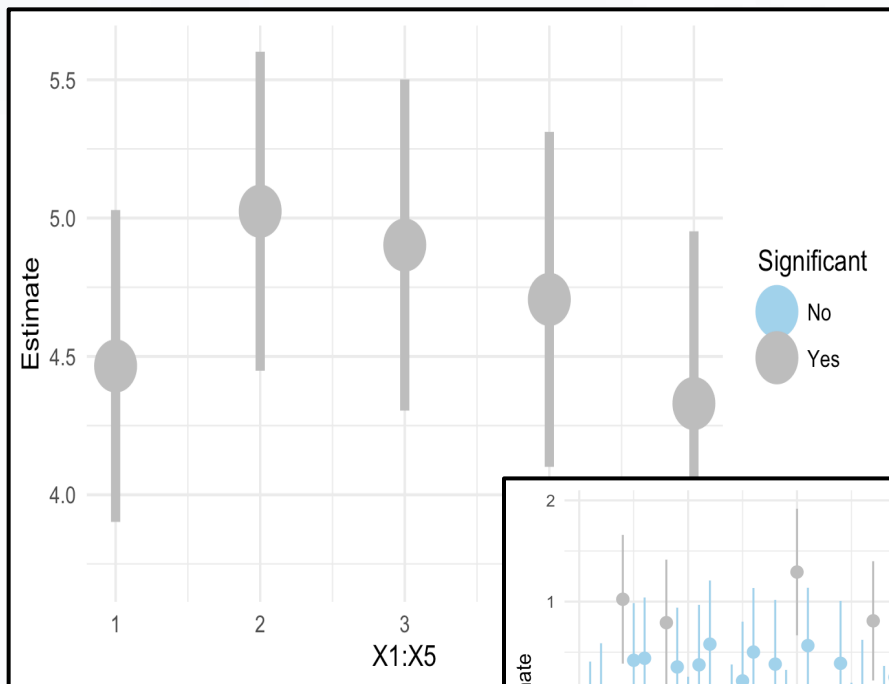
None of the Non-Zero Parameters Will Be Ignored

- Fit Linear Model for Variables Kept in Consideration

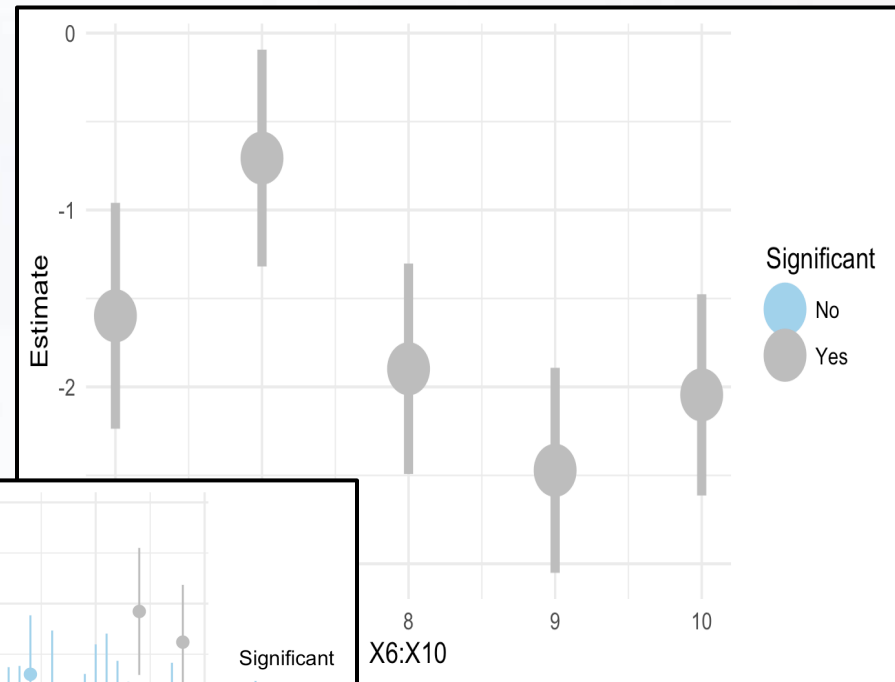
```
> lm(y~.,data=SIM.DATA[,c(1,which(KEEP)+1)])
```

# Part 1-Chunk 4

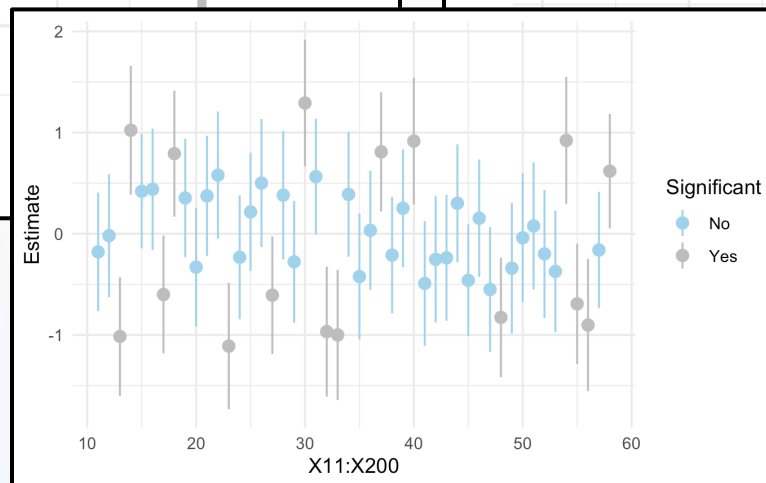
- Suppose Cutoff is 0.2
  - Figure 1



- Figure 2



- Figure 3





# Part 1: Recap

- Recap
  - Before Building Complex Models We are Performing a Simple Screening Procedure
- Problems
  - We May Lose Variables with Significant Interactions
  - We May Still Have Too Many
  - We May Retain Variables that are Highly Correlated

# Shrinkage Estimation

- Classic Linear Model Estimation
  - Minimize Sum of Squared Error

$$SSE = \sum [y_i - (\beta_0 + \mathbf{x}_i' \boldsymbol{\beta})]^2$$

- Optimization: Find  $\widehat{\beta}_0$  and  $\widehat{\boldsymbol{\beta}}$  that Make SSE as Small as Possible
  - $\widehat{\beta}_0$  and  $\widehat{\boldsymbol{\beta}}$  are Easily Found Using Matrix Representation
- Regularized Estimation
  - Produces Biased Estimates
  - Shrinks Coefficients Toward 0
  - Favors Smaller Models
  - May Lead to a Better Model for Out-of-Sample Prediction

# Shrinkage Estimation

- Three Popular Methods
  - Download R Package
  - Penalized SSE

```
> library(glmnet)
```

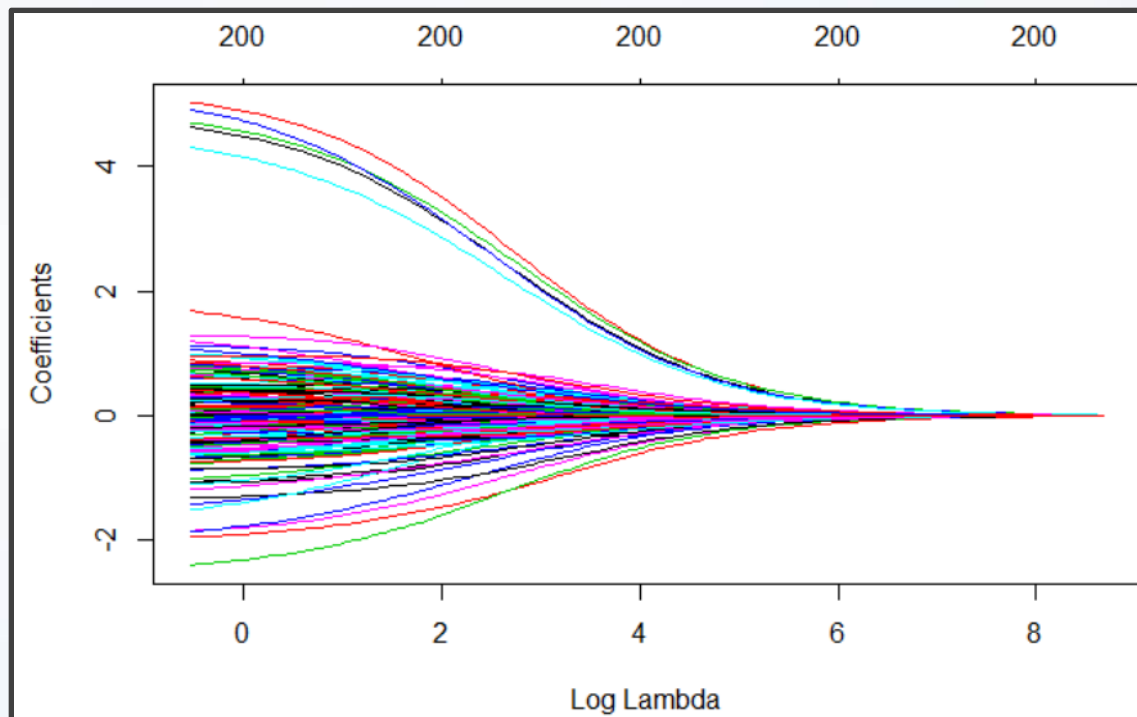
$$PSSE = SSE + \lambda[(1 - \alpha) \sum_{i=1}^p \beta_i^2 + \alpha \sum_{i=1}^p |\beta_i|]$$

- Variations
  - Ridge (1970):  $\lambda = 1$  &  $\alpha = 0$
  - Lasso (1996):  $\lambda = 1$  &  $\alpha = 1$
  - Elastic Net (2005)  
 $\lambda = 1$  &  $0 < \alpha < 1$
- Notice When
  - $\lambda = 0$  PSSE=SSE
  - As  $\lambda$  Gets Bigger, the Coefficients Approach 0

# Part 2: Ridge

- Run Chunk 1
  - Ridge Penalty

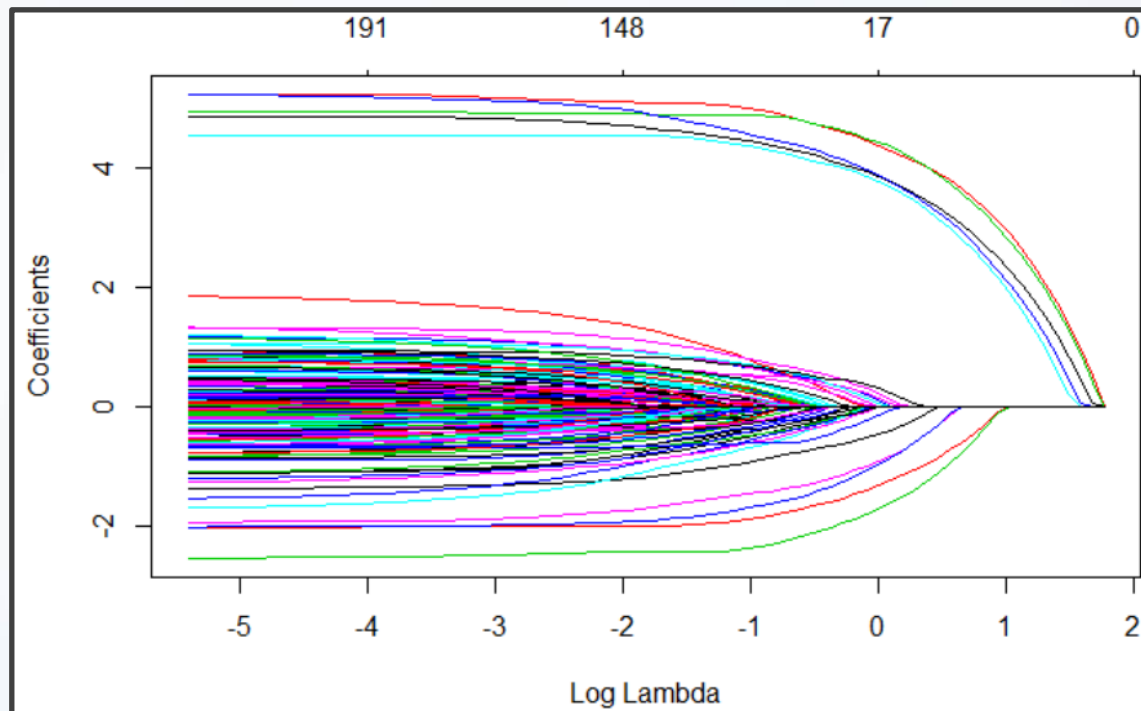
```
> ridge.mod=glmnet(x=as.matrix(SIM.DATA[,-1]),  
+                  y=as.vector(SIM.DATA[,1]),  
+                  alpha=0)  
> plot(ridge.mod,xvar="lambda")
```



# Part 2: Lasso

- Run Chunk 2
  - Lasso Penalty

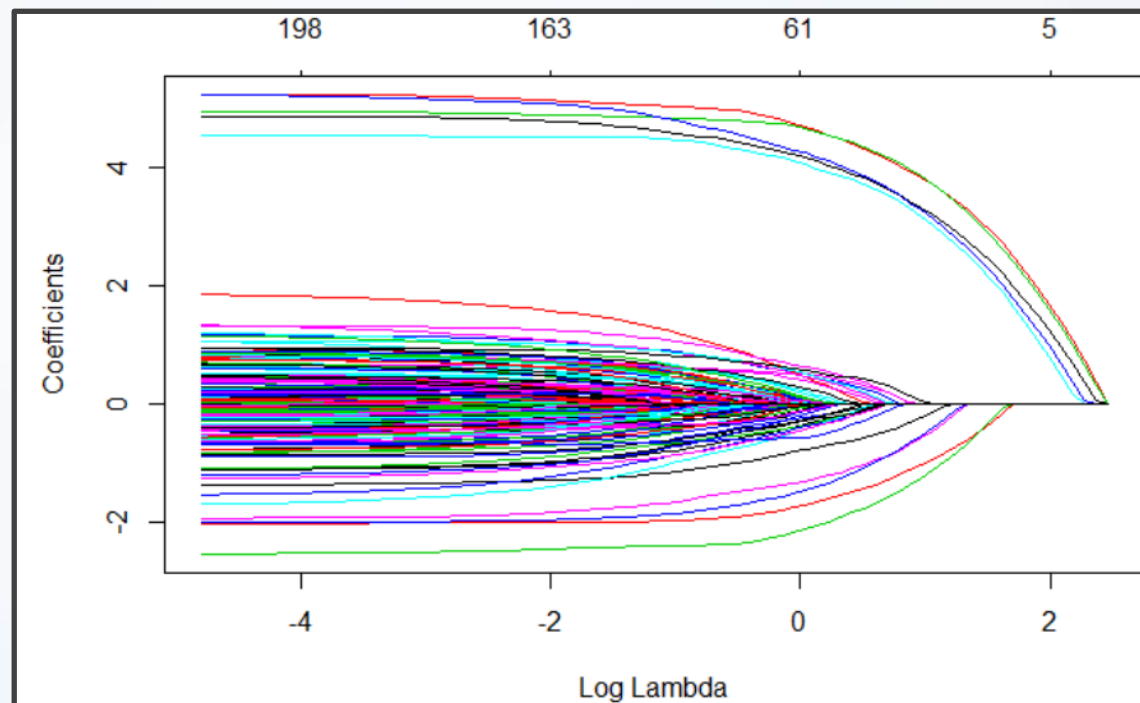
```
> lasso.mod=glmnet(x=as.matrix(SIM.DATA[,-1]),  
+                  y=as.vector(SIM.DATA[,1]),  
+                  alpha=1)  
> plot(lasso.mod,xvar="lambda")
```



# Part 2: Elastic Net

- Run Chunk 3
  - Elastic Net Penalty

```
> enet.mod=glmnet(x=as.matrix(SIM.DATA[,-1]),  
+                  y=as.vector(SIM.DATA[,1]),  
+                  alpha=1/2)  
> plot(enet.mod,xvar="lambda")
```



# Next Steps

- Tuning Parameters
  - Use Cross-Validation to Choose Tuning Parameters  $\lambda$  &  $\alpha$
  - Constraints
    - $\lambda > 0$
    - $0 \leq \alpha \leq 1$
  - Best Approach:
    - Divide Data Into Train & Test
    - Loop Over a Vector of Alpha
    - Find Best Lambda for Each Alpha Considered Using CV in Train
    - For Each Alpha and Best Lambda, Predict on Test and Select Alpha and Lambda that Minimize MSE

# Part 2-Chunk 4

- Run Chunk 4
  - Illustration of 10 Fold CV
  - Finding Best Combination of Alpha and Lambda

alpha	lambda	MSE
0.0	17.282127	176.3021
0.1	7.837234	146.4758
0.2	5.180181	139.9872
0.3	3.453454	133.7793
0.4	2.590091	130.7873
0.5	2.495819	132.6983
0.6	1.895081	129.1495
0.7	1.624355	128.1601
0.8	1.559887	129.2083
0.9	1.386566	128.5799
1.0	1.247909	128.0857



Best:  $\alpha = 1$  &  $\lambda = 1.25$



# Part 2-Chunk 5

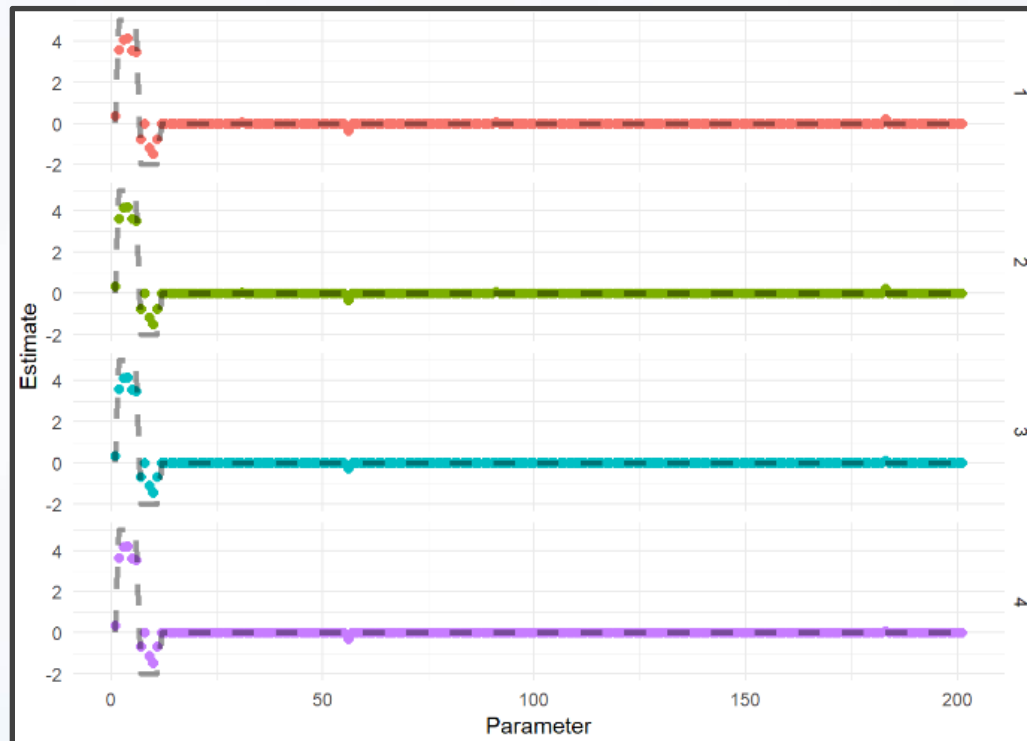
- Run Chunk 5
  - The Top 4 Models

##	alpha	lambda	MSE
## 1	0.6	1.895081	129.1495
## 2	0.7	1.624355	128.1601
## 3	0.9	1.386566	128.5799
## 4	1.0	1.247909	128.0857

- Question: How Different Are These Models?
- For Each Alpha & Lambda,
  - Get Final Coefficients
  - Compare Across Models
  - Compare to True Values

# Part 2-Chunk 5

- Chunk 5 (Continued)
  - Visualizing Top Four
    - Points Show Estimates
    - Dashed Line Shows Truth



# Part 3: More Application

- Built-In Data `> mpg`
  - $n=234$
  - Focus is on Modeling Hwy MPG
  - Subset Data to Include Only Wanted Covariates

<b>year</b> <int>	<b>displ</b> <dbl>	<b>cyl</b> <int>	<b>drv</b> <chr>	<b>cty</b> <int>	<b>hwy</b> <int>	<b>fl</b> <chr>	<b>class</b> <chr>
1999	1.8	4	f	18	29	p	compact
1999	1.8	4	f	21	29	p	compact
2008	2.0	4	f	20	31	p	compact
2008	2.0	4	f	21	30	p	compact
1999	2.8	6	f	16	26	p	compact
1999	2.8	6	f	18	26	p	compact

- There are  $p=7$  Covariates
- Difficulty
  - Fitting all Combinations
  - Considering All 2-Way Interaction Terms

# Part 3-Chunk 1

- Run Chunk 1
  - Creating Model Matrix
    - Up to 2-Way Interactions
    - Now,  $p=114$
  - Model Selection is Difficult
    - Dividing Data into Train & Test is Not Advised ( $n=234$ )
- Run Chunk 2
  - Only a Few Options

<b>alpha</b> <dbl>	<b>lambda</b> <dbl>	<b>CV.Error</b> <dbl>
0.00	1.44063441	1.722966
0.25	0.55006214	1.620769
0.50	0.18956825	1.488094
0.75	0.10492193	1.456773
1.00	0.04942052	1.411025

Lowest Estimation of Prediction Error

# Part 3-Chunk 2

- Chunk 2 (Continued)
  - Understanding cv.glmnet Object
    - `$lambda` = Contains Vector of Lambda Auto-Generated
    - `$cvm` = Cross Validated Estimate of Error for Each Lambda in `$lambda`
    - `$lambda.min` = The Lambda that Leads to Smallest CV Measure of Error
    - `$lambda.1se` = The Largest Value of Lambda Such That Error is Within 1 SD of the Error Using `$lambda.min`

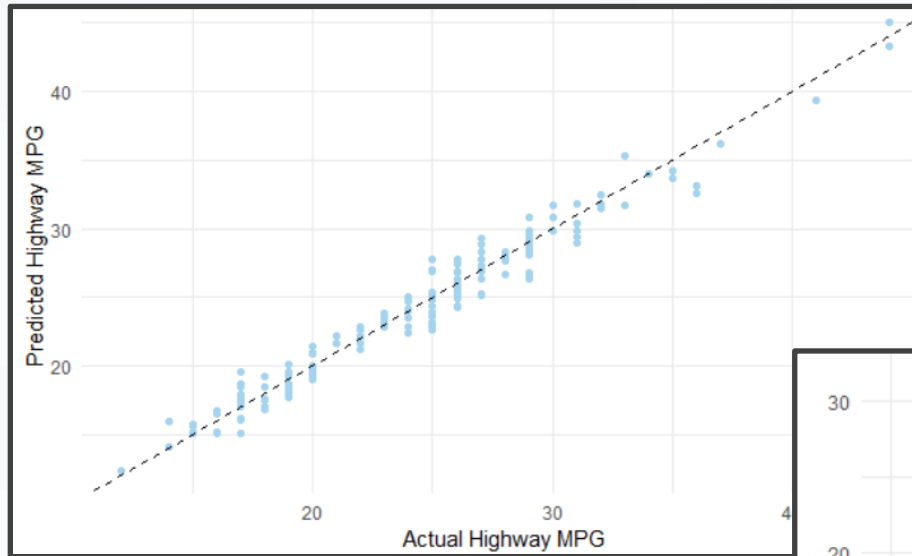
# Part 3-Chunk 3

- Run Chunk 3
  - Next
    - Use Best Alpha and Lambda
    - Observe the Non-Zero Coefficients
    - Plot Predictions and Errors
- Table of Non-Zero Coefficients
  - Before  $p=114$
  - Now  $p=28$

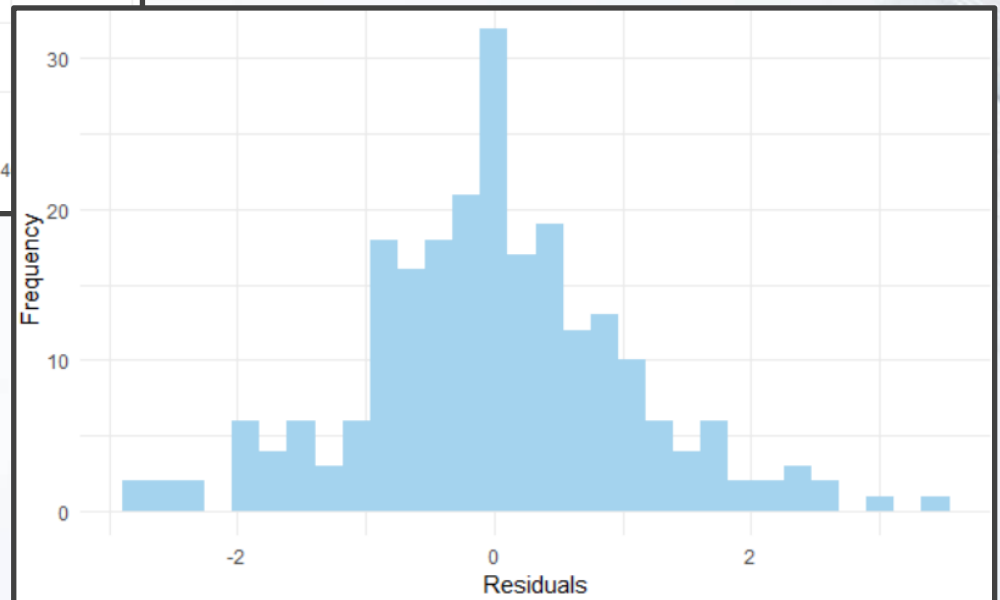
```
## # A tibble: 29 x 2
##   Parameter      Estimate
##   <chr>          <dbl>
## 1 Int           -123.
## 2 year            0.0660
## 3 cty             0.799
## 4 flr            -1.37
## 5 flr           -0.0629
## 6 classpickup   -0.104
## 7 classssuv    -1.37
## 8 year:cyl      -0.0000392
## 9 year:drv      0.0000955
## 10 year:cty     0.0000565
## 11 year:classmidsize 0.0000259
## 12 year:classpickup -0.000659
## 13 displ:drv     0.127
## 14 displ:classmidsize 0.0317
## 15 displ:classssuv -0.178
## 16 cyl:flr      -0.143
## 17 cyl:flr     -0.0973
## 18 cyl:classcompact 0.0462
## 19 cyl:classssuv -0.0262
## 20 drv:cty      0.0466
## 21 drv:cty      0.0282
## 22 drv:fld      2.54
## 23 drv:classsubcompact -0.0754
## 24 cty:classminivan -0.0574
## 25 cty:classpickup -0.106
## 26 flr:classmidsize 0.488
## 27 flp:classsubcompact -1.42
## 28 fld:classssuv -0.552
## 29 flp:classssuv -0.431
```

# Part 3-Chunk 3

- Comparing Predict and Actual



- Distribution of Residuals



# Part 3: Example 2

- Ecdat Data > Participation
  - Labor Market Participation of Married Women in Switzerland
  - Data From 1981
    - 872 Married Women
    - Variables
      - Participation (Binary)
      - Non-Labor Income (log transformed)
      - Age (Scaled by 10)
      - Education (Years)
      - # of Young Children
      - # of Older Children
      - Foreigner (Binary)



# Part 3-Chunk 4

- Run Chunk 4
  - Observe the Data

	lfp	lnnlinc	age	educ	nyc	noc	foreign
1	no	10.78750	3.0	8	1	1	no
2	yes	10.52425	4.5	8	0	1	no
3	no	10.96858	4.6	9	0	0	no
4	no	11.10500	3.1	11	2	0	no
5	no	11.10847	4.4	12	0	2	no
6	yes	11.02825	4.2	12	0	1	no

- We Would Like to Build a Model to Predict Labor Involvement
- Method: Logistic Regression

# Part 3-Chunk 5

- Run Chunk 5
  - Only a Few Options

<b>alpha</b> <dbl>	<b>lambda</b> <dbl>	<b>CV.Error</b> <dbl>
0.00	0.18805333	0.3405963
0.25	0.06542368	0.3291284
0.50	0.04745928	0.3348624
0.75	0.03472433	0.3348624
1.00	0.02604325	0.3348624

Lowest Estimation of Prediction Error

- Notice Using Binomial Family
- What is the Purpose of the Following?

```
type.measure="class"
```

# Part 3-Chunk 6

- Run Chunk 6
  - Only Considering Best Choices
- Observe the Coefficients
  - Useful Variables?
  - Useless Variables?
- Observe the Confusion Matrix
  - Misspecification Error Matches What We Saw
  - $0.329 = \frac{78+209}{393+78+209+192}$
- Write Code That Counts
  - # of Labor Participants
  - # of Predicted Participants