University of Ottawa

Project Report: Project 3 LASSO Cross Validation

MAT 4376

Submitted to Dr. Rafal Kulik

Jimmy Huang

April 13, 2024

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Problem Statement

In the context of high-dimensional statistics, LASSO is known as *Least Absolute Selection and Shrinkage Operator* [3]. When we have $p > n$ where $p =$ the number of variables and $n =$ number of observations, we typically wish to implement LASSO instead of classical Ordinary Least Squares (OLS) regression, it was mentioned that in an earlier work that was done by Dr. Ryan Tibshirani at the University of California, Berkeley, states that:

" The lasso solution is unique when $\text{rank}(X) = p$, because the criterion is strictly convex, but this is not true when $\text{rank}(X) < p$, and in this case because when the number of variables exceeds the number of observations, $p > n$, we must have $\text{rank}(X) < p$." [5]

According to the work done by Dr. Trevor Hastie in section 2.5, we usually write the Lasso estimator in the Lagrangian form as the following:

$$\text{minimize}_{\beta \in \mathbf{R}^p} \{ \frac{1}{2N} ||y - X\beta||_2^2 + \lambda ||\beta||_1 \}$$

for some $\lambda \geq 0, y = (y_1, \cdots, y_N)$ denote the $N-$ vector of responses, and X be an $N \times p$ matrix with $x_i \in \mathbf{R}^p$ in its $i-th$ rows [3].

λ is a tunable parameter, we often want to find the best lambda possible for the Lasso model, usually, cross-validation and BIC selection are used. An existing package in R called *glmnet* [2] has a built-in function called *cv.glnment* that applies cross-validation for the model estimated, this report aims to write a function that does the same thing as *cv.glnment* and to compare the outputs produced by this function with *cv.glnment* from *glmnet.*

1.2 Project Goals

1. Write a function for LASSO Cross Validation

2. Avoid using sophisticated commands from *glmnet*

3. Compare and visualize the results, especially how the chosen λ depends on m where $m =$ number of training subset, which is one of the main inputs

4. Ensure that if my functions work for the first sample, they would also work for the test sample, and they would work everywhere else.

1.3 Expected Outcome

It is expected that the outputs produced from the functions that I wrote would be consistent with the outputs produced by *cv.glnment* from *glmnet*, or they would work at least as effectively as the *cv.glnment* function.

2 Data Source and Methodology

2.1 Data Source

The dataset was downloaded on Kaggle called Body Measurement [1], Table 6 on the appendix page has all of the descriptions for this dataset. This dataset contains $N = 507$ observations, with 25 variables ($p = 25$), one of them is a binary variable, and then the rest of them are all scalar variables. No missing values were found, and no manipulations were performed in this report. The dataset being used is as it is on the Kaggle page. This dataset tracks sex, and human characteristics, such as age, weight, height and so on among 507 respondents, therefore, this data set can be assumed as a classical health science example with a decent amount of predictors.

2.2 Design and Sample

The body measurement dataset is split into two samples, one with observations that were identified as male where sex = 1, and another one with observations that were identified as female, where sex = 0. The female sample is being used for developing and validation purposes (called development and validation phase), and the male sample is being used for testing purposes (called testing phase), if the functions that I have written work for the female sample, they should also work for the male sample, and then they should be applicable to any case that takes three major inputs: a response vector, a design matrix, and the number of folds in R.

The design matrix for the female sample is denoted as X_{Female}, the size of X_{Female} is 247×23, and the response vector is denoted as Y_{Female}, it is a 260×1 column vector, it indicates the weight of respondents in kilograms. The details of the female sample can be found in Table 7.

The design matrix for the male sample is denoted as X_{Male}, the size of X_{Female} is 247×23, and the response vector is denoted as Y_{Male}, it is a 247×1 column vector, it indicates the weight of respondents in kilograms. The details of the male sample can be found in Table 8.

2.3 Cross-Validation

The main objective of the report is to write a function that does cross-validation without using sophisticated commands from the package *glmnet*, and compare the result with the function *cv.glmnet* from *glmnet*. The functions that I wrote for this project are called *My_cv_Lasso* and *Varitant_My_cv_Lasso*, *My_cv_Lasso* takes 3 major inputs, namely, the design matrix, the response vector, and m, where m indicates the number of folds, *Varitant_My_cv_Lasso* takes 4 inputs, namely, the design matrix, the response vector, m, where m indicates the number of folds and a modifiable parameter μ.

The Cross-Validation (CV) works as the following according to the lecture note:

1. Divide the dataset n into m disjoint sets $D_1, \cdots, D_m$ of size n/m each.

2. For each $\lambda \in \Lambda$, evaluate $\hat{\beta}_{LASSO}(\lambda)$ the LASSO estimator based on the dataset $\frac{D}{D_h}, h = 1, \cdots, m$.

   Each D_h is treated as a *test dataset*, while D/D_h as a *training dataset*.

3. Thus, for each $\lambda \in \Lambda$ we get h LASSO estimator, making in total $h \times q$ LASSO estimators.

4. Define the loss function

$$CV(\lambda) = \sum_{h=1}^{m} \sum_{i:(X_i,Y_i)\in D_h} (Y_i - X_i^T \hat{\beta}_{LASSO}^{-(h)}(\lambda))^2$$

5. Choose λ that minimizes the loss function

2.4 Selection for Lambda

Selecting λ is a critical step in cross-validation, the default setting for λ in *cv.glment* comes from initial fit from *glmnet* function. The report uses Λ_i where $i = 1, 2, 3, 4$. Λ_1 is the set that was produced by *glmnet* for the female set, and Λ_2 is the set that was produced for the female set with the alternative approach. Λ_3 is the set that was produced by *glmnet* for the male set, and Λ_4 is the set that was produced for the female set with the alternative approach. The function *Variant_My_cv_Lasso* uses an alternative approach to estimate Λ.

It is proposed to use an alternative way to set up the sets $\Lambda_2 \wedge \Lambda_4$. In *Variant_My_cv_Lasso*, the proposed $\Lambda_2 \wedge \Lambda_4 \sim \text{Exp}(\mu)$ where $\mu = 5$.

The probability density function of an exponential distribution works as the following:

$$f(x;\mu) = \begin{cases} \mu \cdot \exp(-\mu \cdot x) & x \geq 0 \\ 0 & x < 0 \end{cases}$$

where $\mu > 0$

Variant_My_cv_Lasso uses *rexp* [4] function is R, it produces 100 random elements with rate = 5 as default. The prior belief in the *My_cv_Lasso* function is that the parameter space should try different values between 0 and 1, in practice, it is believed that using smaller values for λ would be a better way to optimize our Lasso model than using larger values. Because larger values for λ tend to shrink more coefficients to 0, it may not be the best way to optimize the model, using a random vector that follows an exponential distribution would provide an intensive space between 0 and 1, so that the prior belief is based on the assumption that it could be a better approach to select Λ than the default method by *glmnet*. The μ here is an adjustable parameter, in this case, $\mu = 5$ is assumed, while in practice, μ can be any number that is greater than 0.

2.5 Evaluation for the Chosen Lambda Depending on m

To access the chosen λ depending on the number of training subsets m, it was designed to use $m = 3, 4, \cdots, 19, 20$. It is assumed that $m < 21$ is a reasonable number to validate the consistency of each function, given that both sets have less than 260 observations, and *cv.glmnet* can only take $m \geq 3$ The comparisons are based on *cv. glmnet* from *glmnet* and *My_cv_Lasso* and its variant *Variant_My_cv_Lasso*.

2.6 Monte Carlo Method

Due to the randomness, the report compares the outputs using the Monte Carlo Method with 500 iterations for each case, the evaluations are based on $m = 5, 10, 15$. The comparisons are based on outputs produced by *cv.glmnet* and the outputs produced by *My_cv_Lasso* and *Variant_My_cv_Lasso*. The Monte Carol method is a simulation technique that allows us to see more variations, therefore, it is a robust way to validate outputs.

2.7 Efficiency and Elapsed Time

It is also crucial to check the running time of the customized functions and *cv.glmnet*, the elapsed times are also reported using *system.time* in R. It is designed to use a sample for each set with $m = 10$, and with 10 iterations.

3 Result

3.1 Coefficient Paths and Initial Fits for Both Sets

The following results are based on the initial fits, where the "optimal" λ has not been found yet. Figure 2 shows which coefficients would be selected if $\log(\lambda)$ equals a particular value, the y-axis shows the estimated values of the coefficients and the x-axis shows the $\log(\lambda)$ and the number on the top of each plot shows how many predictors would be kept when $\log(\lambda) =$ certain value.

The summary of the λ that the *glmnet* provided are shown as the following:

	n	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
For the Female Set	73	0.011	0.057	0.30	1.33	1.63	8.66
For the Male Set	72	0.012	0.065	0.34	1.44	1.77	9.21

Table 1: Summary of Λ Estimated by *glmnet*

3.2 Result of ***cv.glmnet***, ***My_cv_Lasso***, and ***Varitant_My_cv_Lasso*** In the Development and Validation phase

My_cv_Lasso The female set is used in the development and validation phase. The range of $\Lambda_1 \wedge \Lambda_2$ for the female set is shown as the following:

	n	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
Λ_1	73	0.011	0.057	0.30	1.33	1.63	8.66
Λ_2	100	0.0024	0.06	0.15	0.22	0.31	0.92

Table 2: Summary of Λ for the Female Set Determined by *cv.glmnet* and *My_cv_Lasso*

Figure 4 shows how they behaved. The following table shows the summary of the best lambda found by both *cv. glmnet*,*My_cv_Lasso* and *Variant_My_cv_Lasso*

It was shown that the λ found by the *Variant_My_cv_Lasso* behaved more conservatively

	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
By *cv.glmnet*	0.019	0.07	0.083	0.074	0.083	0.12
By *My_cv_Lasso*	0.020	0.07	0.083	0.076	0.083	0.13
By *Variant_My_cv_Lasso*	0.063	0.07	0.081	0.081	0.084	0.12

Table 3: Summary of λ Depending on m in the Development and Validation Phase

than the *cv. glmnet* and *My_cv_Lasso* as m increased. The *Variant_My_cv_Lasso* function suggested that the best possible λ could be around 0.08 for this case.

Pick $m = 5, 10, 15$. The Monte Carlo method gives the following results: Figure 5, Figure 6 andFigure 7 .The blue lines indicate the mean of best λ found, the red lines and the green lines indicate the 95% quantiles, where red lines indicate the lower bound (0.025), the green lines indicate the upper bound (0.975). The plots suggested that as m increased, the best λ estimated for the female set would be somewhere close to 0.08.

Table 9 shows the means of λ in the Monte Carlo simulation, and their corresponding Mean Square Errors (MSE) and how many variables are non-zero. It was found that all λ found the MSEs in the development and validation phase are around 3.11, this gives confidence in the efficiency of the functions *My_cv_Lasso* and *Variant_My_cv_Lasso*, they are expected to demonstrate the same ability in the testing phase as they did in the development and validation phase.

3.3 Result of *cv.glmnet*, *My_cv_Lasso*, and *Varitant_My_cv_Lasso* In the Testing Phase

The ranges of $\Lambda_{3,4}$ are listed as the following table:

	n	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
Λ_3	72	0.012	0.065	0.34	1.44	1.77	9.21
Λ_4	100	0.0045	0.057	0.172	0.217	0.292	1.25

Table 4: Summary of $\Lambda_{3,4}$

For the chosen λ depending on the number of folds in the test phase, Figure 9 shows that *cv.glmnet*, *My_cv_Lasso* followed a similar tendency as m increased, while, the best λ found by *Varitant_My_cv_Lasso* behaved consistently around 0.08.

The following table shows the summary of the best λ found by both *cv.glmnet*, *My_cv_Lasso* and *Varitant_My_cv_Lasso*.

	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
By *cv.glmnet*	0.022	0.068	0.083	0.073	0.080	0.964
By *My_cv_Lasso*	0.022	0.07295	0.080	0.077	0.086	0.096
By *Variant_My_cv_Lasso*	0.062	0.076	0.079	0.079	0.08234	0.102

Table 5: Summary of λ Depending on m in the Testing Set

Using the Monte Carlo method, Figure 10, Figure 11 and Figure 12 showed that the mean of the best λ found in these cases would be somewhere around 0.068 to 0.07, which suggested the consistency of the results produced by *My_cv_Lasso* and *Varitant_My_cv_Lasso* are similar to the results produced by *cv.glmnet* in terms of findings the best λ. At the same time, it is noticeable that there does not exist a "Best" λ. By using the mean values in Figure 10, Figure 11 and Figure 12, they provide 9 different λ on the appendix page, namely, .Table 10. This table shows how many variables are kept and which λ produced the smallest MSE. It was shown that the corresponding MSEs are all around 4.54 with the mean value of the "best" lambdas found in the Monte Carlo simulation. It can be shown that there exists a consistency in these functions. And *My_cv_Lasso* and *Varitant_My_cv_Lasso* can work at least as effectively as *cv.glmnet* from the package *glmnet*.

3.4 Result of Elapsed Times

The elapsed times for each function in each set are also reported in the following table: Table 11 and Table 12. It was found that *Variant_My_cv_Lasso* took significantly longer elapsed time than *cv.glmnet* and *My_cv_Lasso* in both sets.

4 Conclusion

4.1 Discussion

In the final stage, it was found that the results support one of the expectations that functions that I wrote can work at least as effectively as *cv.glmnet*. In terms of the comparisons made in this report, such as chosen λ depending on the number of folds and the results obtained from the Monte Carlo method. While it is also noticeable that *Variant_My_cv_Lasso* and *My_cv_Lasso* took significantly longer processing time than *cv.glmnet*, for the *Variant_My_cv_Lasso* this could be due to the reason that there were 100 candidates for λ, while, both *My_cv_Lasso* and *cv.glmnet* used fewer candidates for λ than *Variant_My_cv_Lasso*. Moreover, although the proposed range of Λ in *Variant_My_cv_Lasso* demonstrates some sort of consistency in chosen λ depending on the number of folds in both sets and also in the Monte Carlo simulation, it remains challenging to conclude if this method works strictly better than the other two functions under the prior belief that $\mu = 5$. This suggests that the way *cv.glmnet* selects the range of Λ is more sensible. In future studies, it is suggested to expand the number of iterations using the Monte Carlo method, such as, with 5000 or 10000 iterations for each $m = 3, \cdots, 20$, and modify μ to some other number, but be sure that it searches over the parameter space intensively between 0 to the targeted number, and $\mu > 0$.

4.2 Conclusion

To sum up, the functions *Variant_My_cv_Lasso* and *My_cv_Lasso* can be interchanged with *cv.glmnet* from *glmnet* in terms of finding a sufficient λ for a Lasso regression, and other outputs produced by *Variant_My_cv_Lasso* and *My_cv_Lasso* are fairly consistent with *cv.glmnet* from *glmnet*. Furthermore, *Variant_My_cv_Lasso* and *My_cv_Lasso* do not heavily rely on *glmnet* commands. They only use the fitting and the standardized functions. Comparisons of chosen λ depending on the number of folds were also made, showing that they worked for both the development and validation set and the testing set. It is believed that these functions will work for any situation with a design matrix, response vector and a number of folds, and they will do the same thing as *cv_glmnet* from *glmnet*. With slight modifications, these functions could also accommodate ridge regression or an elastic net object. Additionally, *Variant_My_cv_Lasso* also allows users to define a range of Λ following an exponential distribution, which adds more variability and flexibility if a prior belief about Λ is valid.

References

[1] Maximilian Finsterwald. *Body measurements*. Feb. 2024. URL: https://www.kaggle.com/datasets/mexwell/body-measurements/data.

[2] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *Regularization Paths for Generalized Linear Models via Coordinate Descent*. 2010. URL: https://www.jstatsoft.org/v33/i01/.

[3] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman & Hall/CRC, 2015. ISBN: 1498712169.

[4] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2023. URL: https://www.R-project.org/.

[5] Ryan J. Tibshirani. *The Lasso Problem and Uniqueness*. 2012. arXiv: 1206.0313 [math.ST].

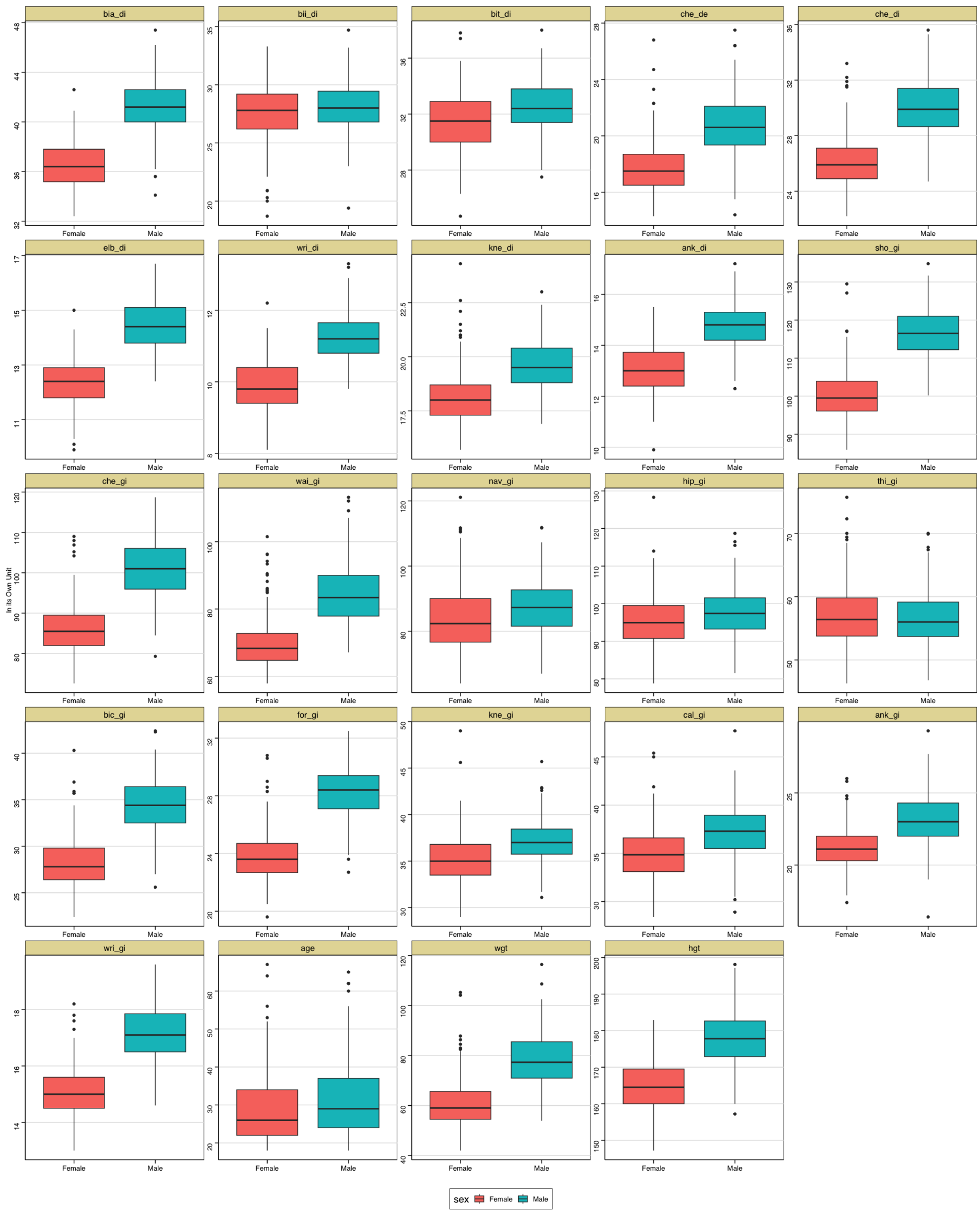
Appendices

1 Description of the Body Measurement Dataset

Variable Name	**Description**
bia_di	Respondent's biacromial diameter in centimeters
bii_di	Respondent's biiliac diameter in centimeters
bit_di	Respondent's bitrochanteric diameter in centimeters
che_de	Respondent's chest depth in centimeters
che_di	Respondent's chest diameter in centimeters
elb_di	Respondent's elbow diameter in centimeters
wri_di	Respondent's wrist diameter in centimeters
kne_di	Respondent's knee diameter in centimeters
ank_di	Respondent's ankle diameter in centimeters
sho_gi	Respondent's shoulder girth in centimeters
che_gi	Respondent's chest girth in centimeters
wai_gi	Respondent's waist girth in centimeters
nav_gi	Respondent's navel girth in centimeters
hip_gi	Respondent's hip girth in centimeters
thi_gi	Respondent's thigh girth in centimeters
bic_gi	Respondent's bicep girth in centimeters
for_gi	Respondent's forearm girth in centimeters
kne_gi	Respondent's knee girth in centimeters
cal_gi	Respondent's calf maximum girth in centimeters
ank_gi	Respondent's ankle minimum girth in centimeters
wri_gi	Respondent's wrist minimum girth in centimeters
age	Respondent's age in years
wgt	Respondent's weight in kilograms
hgt	Respondent's height in centimeters
sex	Respondent's sex (1=male, 0=female)

Table 6: Description of the Body Measurements Dataset from [1] with $N = 507$

Figure 1: General Description of the Body Measurement Dataset By Sex



2 Description of the Female Sample (Development and Validation Set)

Variable Name	Type
bia_di	Independent Variable
bii_di	Independent Variable
bit_di	Independent Variable
che_de	Independent Variable
che_di	Independent Variable
elb_di	Independent Variable
wri_di	Independent Variable
kne_di	Independent Variable
ank_di	Independent Variable
sho_gi	Independent Variable
che_gi	Independent Variable
wai_gi	Independent Variable
nav_gi	Independent Variable
hip_gi	Independent Variable
thi_gi	Independent Variable
bic_gi	Independent Variable
for_gi	Independent Variable
kne_gi	Independent Variable
cal_gi	Independent Variable
ank_gi	Independent Variable
wri_gi	Independent Variable
age	Independent Variable
hgt	Independent Variable
wgt	Dependent Variable

Table 7: Description of the Female Sample with $n_{\text{Female}} = 260$

3 Description of the Male Sample (Testing Set)

Variable Name	**Type**
bia_di	Independent Variable
bii_di	Independent Variable
bit_di	Independent Variable
che_de	Independent Variable
che_di	Independent Variable
elb_di	Independent Variable
wri_di	Independent Variable
kne_di	Independent Variable
ank_di	Independent Variable
sho_gi	Independent Variable
che_gi	Independent Variable
wai_gi	Independent Variable
nav_gi	Independent Variable
hip_gi	Independent Variable
thi_gi	Independent Variable
bic_gi	Independent Variable
for_gi	Independent Variable
kne_gi	Independent Variable
cal_gi	Independent Variable
ank_gi	Independent Variable
wri_gi	Independent Variable
age	Independent Variable
hgt	Independent Variable
wgt	Dependent Variable

Table 8: Description of the Female Sample with $n_{\mathrm{Male}} = 247$

4 Initial Fits and Coefficient Paths

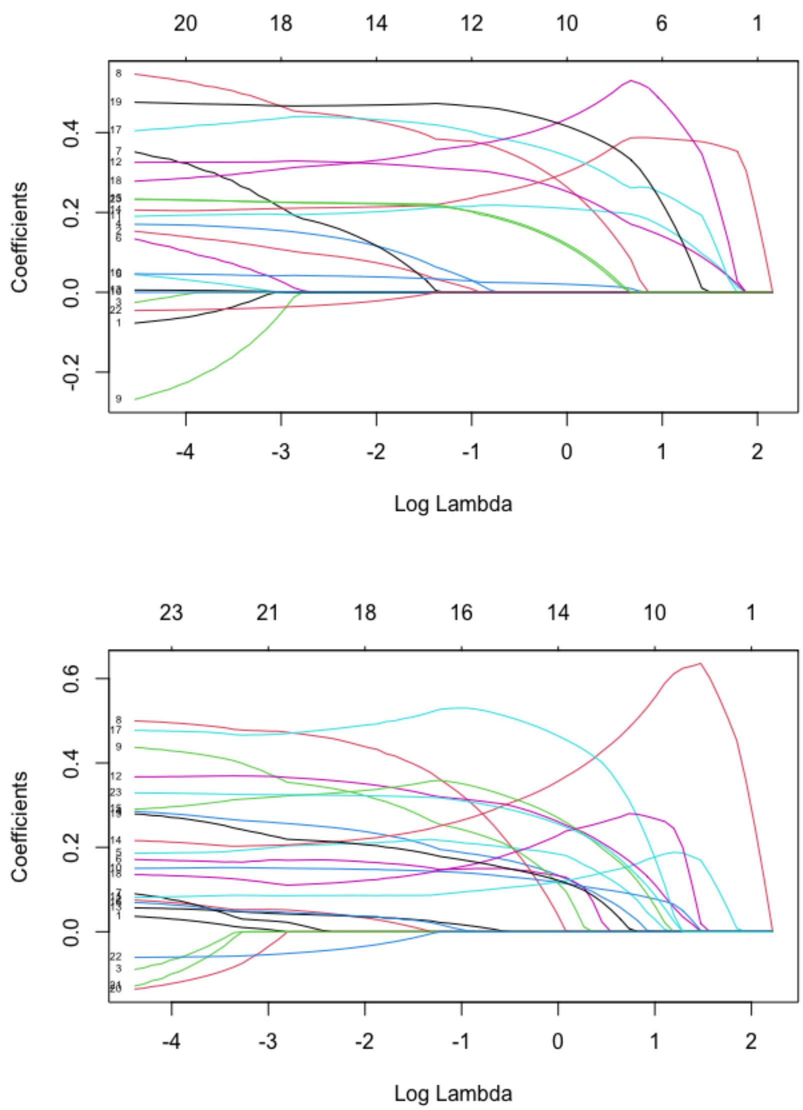


Figure 2: Coefficient Paths for the Female Set (top) the Male Set (bottom)

5 Lambda in the Development and Validation Phase

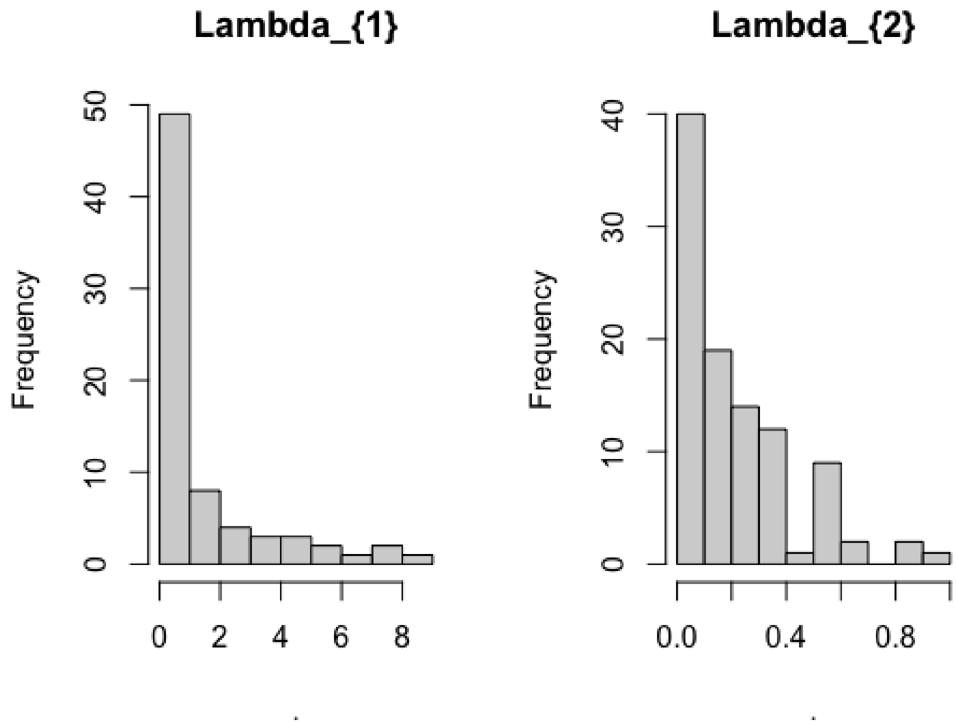


Figure 3: Left Hand Side Λ_1, Right Hand side Λ_2

6 How the Chosen Lambda Depends on the Number of Folds (Development and Validation Phase)

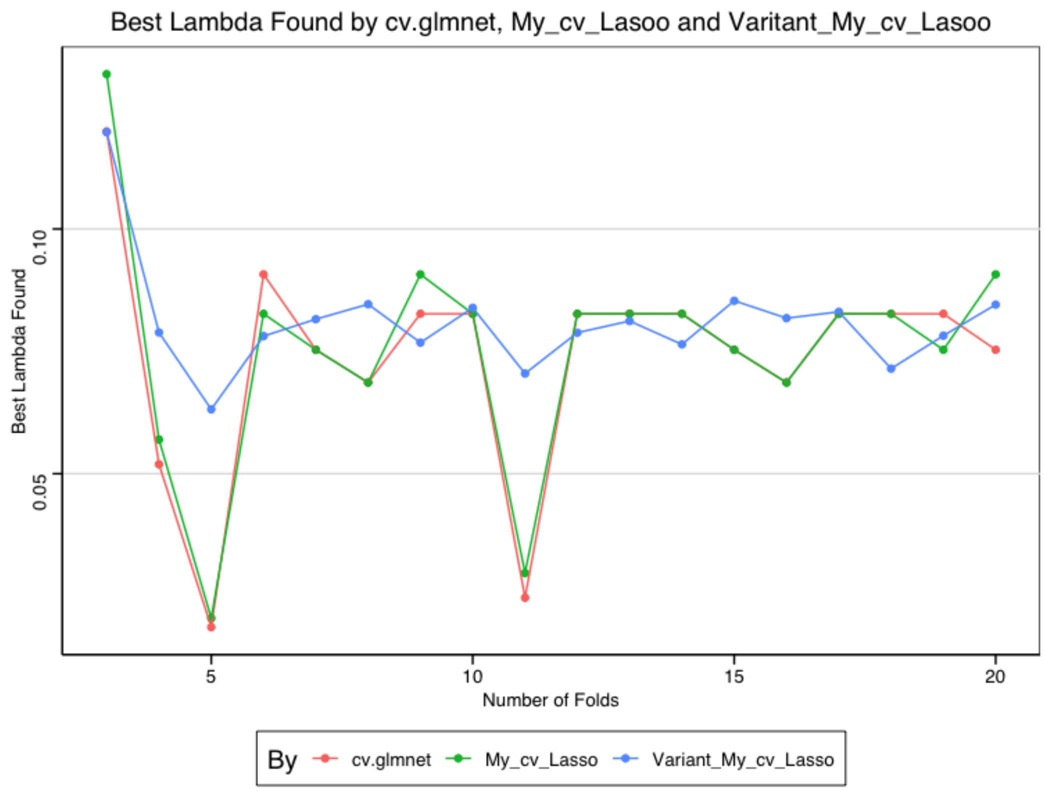


Figure 4: Chosen λ Depending on m for the Female Set

7 Monte Carlo Simulation in the Development and Validation Phase

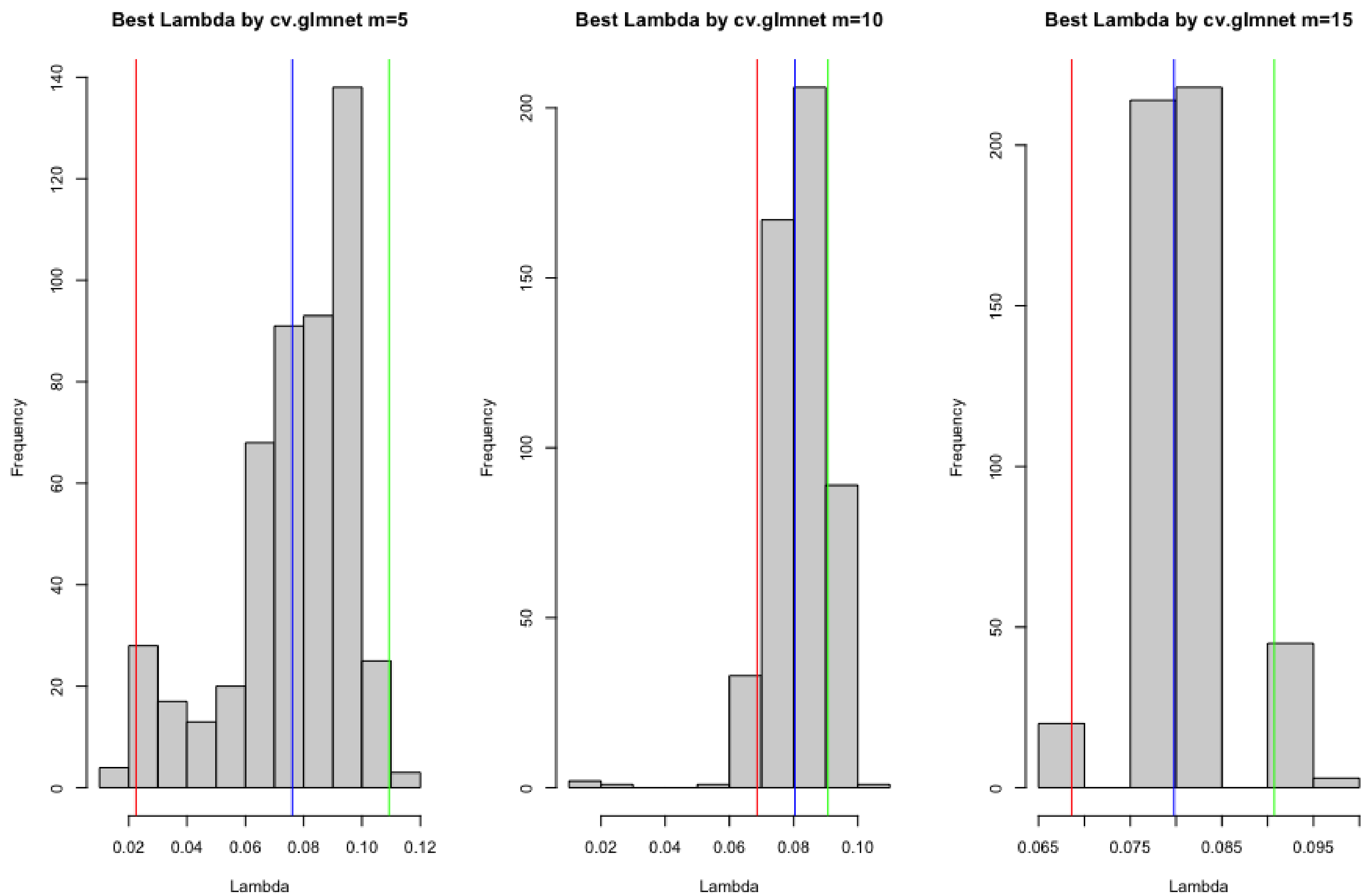


Figure 5: Monte Carlo Simulation for the Female Set using *cv.glmnet*

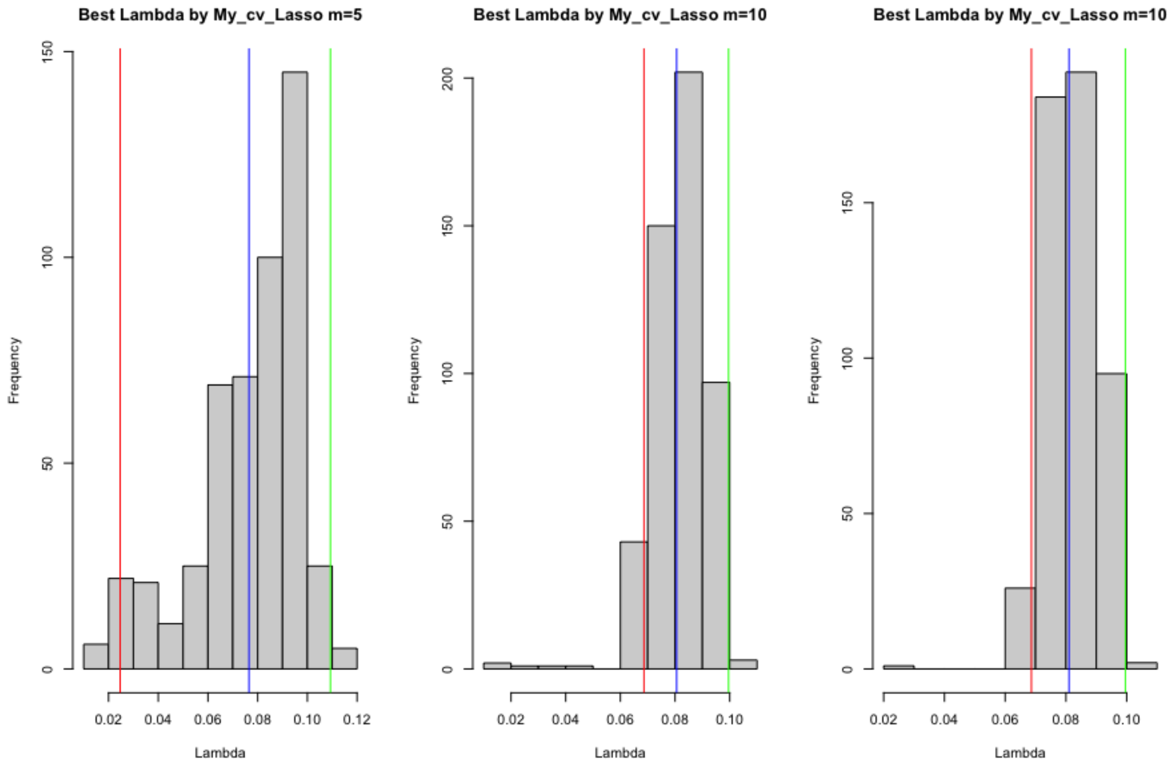


Figure 6: Monte Carlo Simulation for the Female Set Using *My_CV_Lasso*

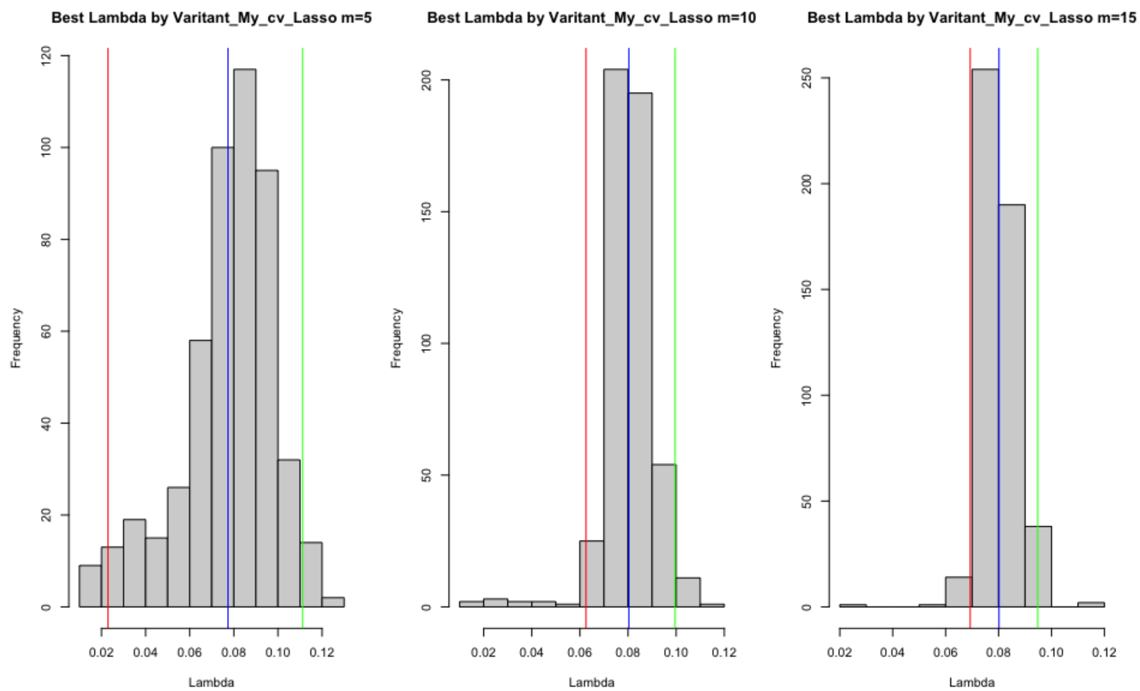


Figure 7: Monte Carlo Simulation for the Female Set Using *Variant_My_CV_Lasso*

8 Summary Table for the Development and Validation Phase

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
λ	0.076	0.0804	0.0798	0.0765	0.0805	0.081	0.0774	0.0803	0.0802
MSE	3.114	3.117	3.116	3.114	3.117	3.118	3.115	3.1168	3.1166
nonzero	15	15	15	15	15	14	15	15	15

Table 9: λ in Different Cases with their Corresponding MSE for the Female Set

Where nonzero indicates the number of variables that are not zero.

Where S_1 is the mean λ found in Figure 5 when $m = 5, S_2$ is the mean found λ in Figure 5 when $m = 10, S_3$ is the mean λ found in Figure 5 when $m = 15$.

Where S_4 is the mean λ found in Figure 6 when $m = 5, S_5$ is the mean λ found in Figure 6 when $m = 10, S_6$ is the mean λ found in Figure 6 when $m = 15$.

Where S_7 is the mean λ found in Figure 7 when $m = 5, S_8$ is the mean λ found in Figure 7 when $m = 10, S_9$ is the mean λ found in Figure 7 when $m = 15$.

9 Lambda in the Testing Phase

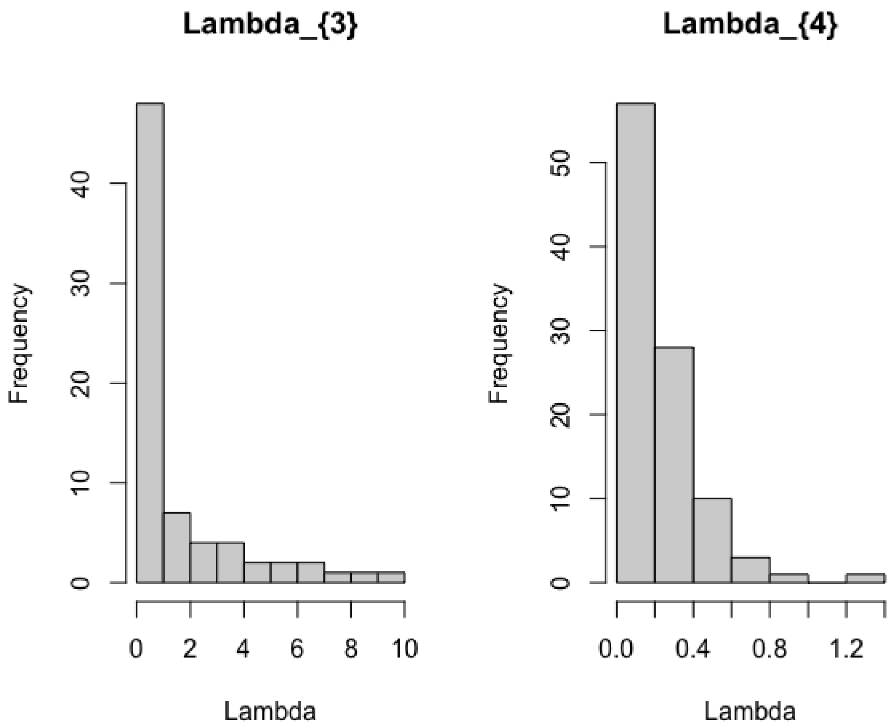


Figure 8: Left Hand Side Λ_3, Right Hand Side Λ_4

10 How the Chosen Lambda Depends on the Number of Folds (Testing Phase)

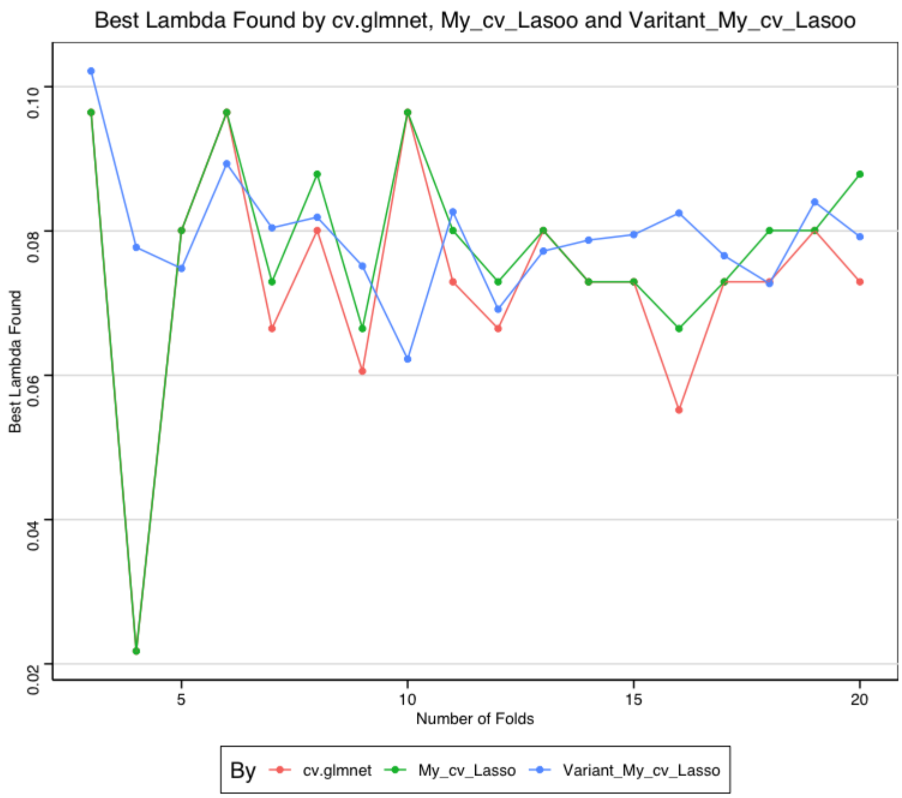


Figure 9: Lambda Depending on m

11 Monte Carlo Simulation for the Lambda in the Testing Set

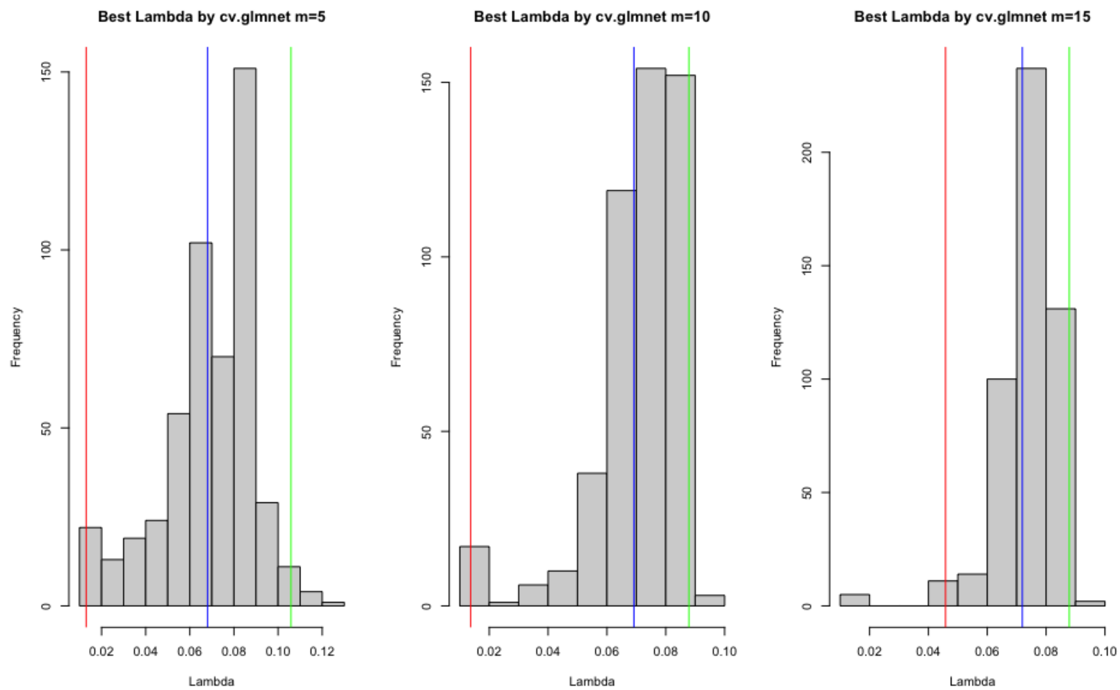


Figure 10: Monte Carlo Simulation for the Male Set using *cv.glmnet*

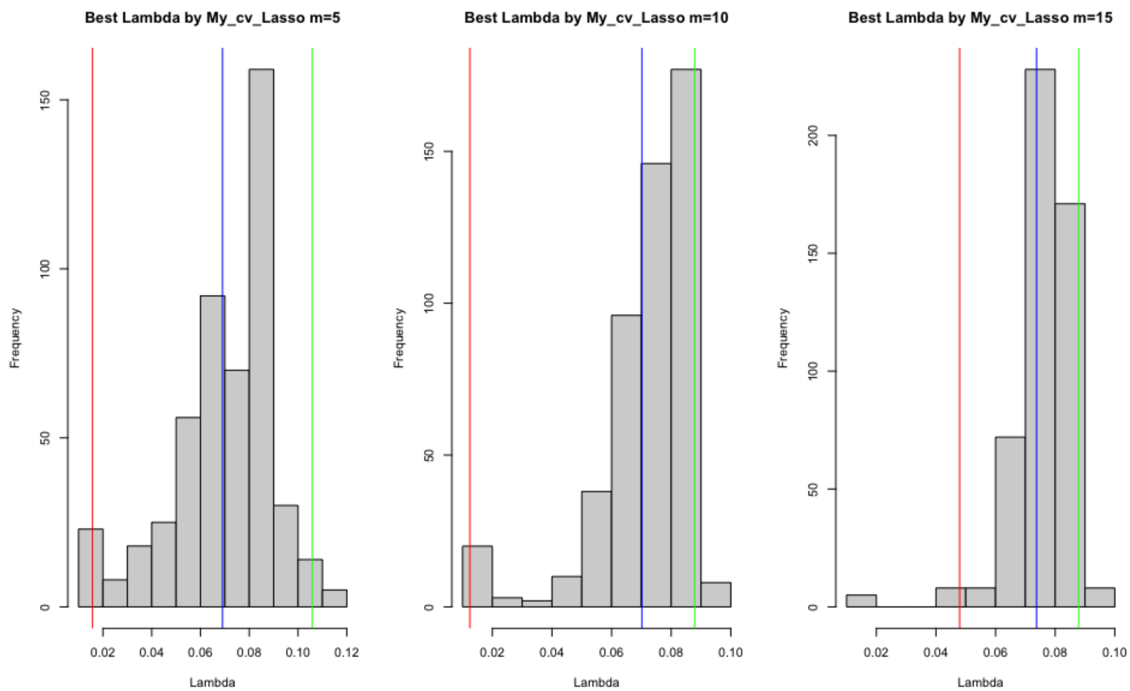


Figure 11: Monte Carlo Simulation for the Male Set using *My_cv_Lasso*

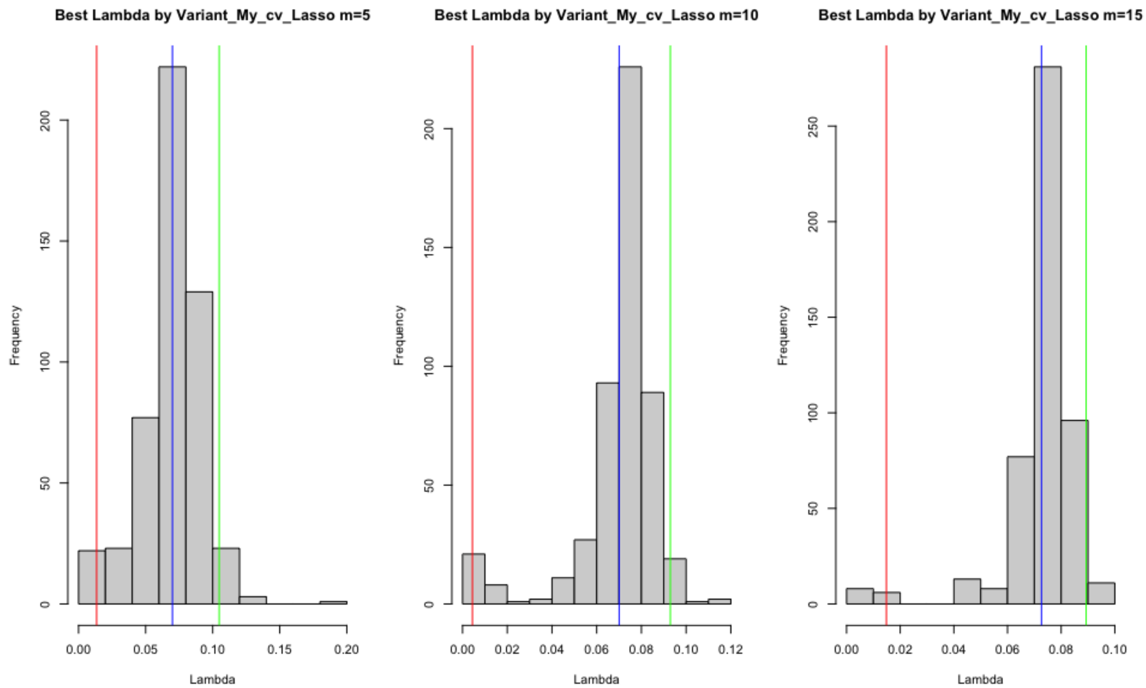


Figure 12: Monte Carlo Simulation for the Male Set using *Variant_My_cv_Lasso*

12 Summary Table for the Testing Phase

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
λ	0.06803	0.06917	0.07192	0.06903	0.07016	0.07374	0.06998	0.07005	0.07266
MSE	4.54098	4.54179	4.54327	4.54160	4.54240	4.54379	4.54225	4.54231	4.54353
nonzero	19	19	19	19	19	19	19	19	19

Table 10: λ in Different Cases with their Corresponding MSE for the Male Set

Where nonzero indicates the number of variables that are not zero.

Where S_1 is the mean λ found in figure 9 when $m = 5, S_2$ is the mean found λ in figure 9 when $m = 10, S_3$ is the mean λ found in figure 9 when $m = 15$.

Where S_4 is the mean λ found in figure 10 when $m = 5, S_5$ is the mean λ found in figure 10 when $m = 10, S_6$ is the mean λ found in figure 10 when $m = 15$.

Where S_7 is the mean λ found in figure 11 when $m = 5, S_8$ is the mean λ found in figure 11 when $m = 10, S_9$ is the mean λ found in figure 11 when $m = 15$.

13 Elapsed Time

cv.glmnet	*My_cv_Lasso*	*Variant_My_cv_Lasso*
0.0318s	3.6987s	4.954s

Table 11: Average Elapsed Time of 3 functions over 10 Iterations in the Development and Validation Set with $m = 10$

cv.glmnet	*My_cv_Lasso*	*Variant_My_cv_Lasso*
0.0327s	3.715s	5.091s

Table 12: Average Elapsed Time of 3 functions over 10 Iterations in the Testing Set with $m = 10$

14 R Code for This Project

The folder has a .r file, it would be better if you were to copy or run them

If you were to run the Monte Carlo Simulation for λ, I would have to warn you that it could take 5-10 minutes for each round

```
######
setwd("/Users/jameswong/Desktop/MAT /MAT4376/Project")
getwd()
library(glmnet)
library(dplyr)
library(ggplot2)
library(ggthemes)
library(reshape)
##### Make Sure that you have them installed
#### Loading the dataset downloaded
df2<-read.csv("bdims.csv")



############ Codes for Figure 1
#######################
#######
df5<-df2
df5<-melt(df5, id.var = "sex")
df5$sex<-factor(df5$sex,levels=c(0, 1),labels=c("Female", "Male"))
df5%>%
  ggplot(aes(x=sex,y=value))+
  geom_boxplot(aes(fill=sex))+
```

```
  facet_wrap(~variable,scales="free")+
  xlab(" ") + ylab("In its Own Unit") +
  ggtitle(" ")+
  theme_stata(scheme = "s1color")
###
######
#########################



##For the female sample (Developing and Validating)
##### 0 indicates female, 1 indicates male
table(df2$sex)
#### Selecting Females only
df3<-df2[df2$sex == 0,]
df3<-df3[,-25]
#######





##### Convert Design Matrix
X1<-as.matrix(df3[,names(df3) != "wgt"])

###### Response Vector
Y1<-as.matrix(df3$wgt)
#######Initial Fit
```

```
mod_lasso1<-glmnet(X1,Y1,alpha=1,family="gaussian")
##### Solution Path
plot(mod_lasso1,label = TRUE,xvar = "lambda")

##### First Try with cv.glmnet with m=10
set.seed(886)
mod_cv_2<-cv.glmnet(X1,Y1,family = "gaussian",alpha=1,nfolds=10)
plot(mod_cv_2)
mod_cv_2$lambda
### Where n comes from
length(mod_cv_2$lambda)
#### Summary of Lambda
mod_cv_2$lambda%>%
  summary()
###### The coefficients kept by cv.glmnet
coef(mod_cv_2) #### 14 variables kept



######### First function My_CV_Lasso
set.seed(886)
My_cv_Lasso<-function(X,Y,nfolds){
  ##### Step 1
  n<-nrow(X)
  store<-data.frame(Y,X)%>%
    mutate(fold=sample(rep(1:nfolds,length.out=n)))
  #### Step 2
```

```
obj_glmnet<-glmnet(X,Y,family = "gaussian",
                   alpha=1)
lambda_candidate<-obj_glmnet$lambda
cv_lasso_error<-matrix(0,nrow=length(lambda_candidate),ncol=nfolds)
for(i in seq_along(lambda_candidate)){
  lam1<-lambda_candidate[i]
  for(j in 1:nfolds){
    training_set<-store%>%
      filter(fold != j)
    test_set<-store%>%
      filter(fold == j)
    X_train<-training_set%>%######inactivate package MASS
      ##### or add dplyr:: in front of select
      dplyr::select(-Y,-fold)%>%
      as.matrix()
    Y_train<-training_set$Y
    X_test<-test_set%>%
      dplyr::select(-Y,-fold)%>%
      as.matrix()
    Y_test<-test_set$Y
    ###### Step 3
    lasso_mod<-glmnet(X_train,Y_train,
                      family = "gaussian",
                      alpha=1,lambda=lam1)
    ###### Step 4
    pred1<-predict(lasso_mod,X_test,s=lam1,"response")
    cv_lasso_error[i,j]<-mean((Y_test-pred1)^2)
```

```
    }
  }
  ########Step 5
  mean_cv<-rowMeans(cv_lasso_error)
  smallest_error<-which(mean_cv == min(mean_cv))
  best_lam<-lambda_candidate[smallest_error]
  ########Done
  return(list(candidate_for_lambda=lambda_candidate,
              mean_loss=mean_cv,
              best_lambda_picked=best_lam))
}


#### First try with My_CV_Lasso
res1<-My_cv_Lasso(X1,Y1,10)
plot(log(res1$candidate_for_lambda),res1$mean_loss,xlab="Log Lambda",
     ylab="Mean Squared Error")
abline(v=log(res1$best_lambda_picked),col="red")
res1




######### My_CV_Lasso_Varitant
set.seed(886)
Variant_My_cv_Lasso<-function(X,Y,nfolds,mu){
  ##### Step 1
```

```r
n<-nrow(X)
store<-data.frame(Y,X)%>%
  mutate(fold=sample(rep(1:nfolds,length.out=n)))
#### Step 2
lambda_candidate<-rexp(100,mu)
cv_lasso_error<-matrix(0,nrow=length(lambda_candidate),ncol=nfolds)
for(i in seq_along(lambda_candidate)){
  lam1<-lambda_candidate[i]
  for(j in 1:nfolds){
    training_set<-store%>%
      filter(fold != j)
    test_set<-store%>%
      filter(fold == j)
    X_train<-training_set%>%######inactivate package MASS
      ##### or add dplyr:: in front of select
      dplyr::select(-Y,-fold)%>%
      as.matrix()
    Y_train<-training_set$Y
    X_test<-test_set%>%
      dplyr::select(-Y,-fold)%>%
      as.matrix()
    Y_test<-test_set$Y
    ###### Step 3
    lasso_mod<-glmnet(X_train,Y_train,
                      family = "gaussian",
                      alpha=1,lambda=lam1)
    ###### Step 4
```

```
      pred1<-predict(lasso_mod,X_test,s=lam1,"response")
      cv_lasso_error[i,j]<-mean((Y_test-pred1)^2)
    }
  }
  ########Step 5
  mean_cv<-rowMeans(cv_lasso_error)
  smallest_error<-which(mean_cv == min(mean_cv))
  best_lam<-lambda_candidate[smallest_error]
  ########Done
  return(list(candidate_for_lambda=lambda_candidate,
              mean_loss=mean_cv,
              best_lambda_picked=best_lam))
}
set.seed(886)
res2<-Variant_My_cv_Lasso(X1,Y1,5,5)
res2
plot(log(res2$candidate_for_lambda), res2$mean_loss, xlab="Log Lambda",
     ylab="Mean Squared Error")
abline(v=log(res2$best_lambda_picked),col="red")

####### Compare the best lambdas picked
#### By glmnet
mod_cv_2

#### By My_CV_Lasso
res1$best_lambda_picked
min(res1$mean_loss)
```

```r
#### By My_CV_Lasso_Varitant
res2$best_lambda_picked
min(res2$mean_loss)



###### Lambda
### Female Set
mod_cv_2$lambda%>%
  hist(breaks=10,main="Lambda_{1}")



res2$candidate_for_lambda%>%
  hist(breaks=10,main="Lambda_{2}")

##############################################

###### Chosen Lambda depending on m

#### By cv.glmnet
set.seed(886)
m1<-3:20
best_lambdas_m<-rep(0,length(m1))
for(i in seq_along(m1)){
  m<-m1[i]
  results1<-My_cv_Lasso(X1,Y1,m)
  best_lambdas_m[i]<-results1$best_lambda_picked
}
```

```r
#### By My_CV_Lasso
set.seed(886)
best_lambdas_m_glmnet<-rep(0,length(m1))
for(i in seq_along(m1)){
  m<-m1[i]
  results2<-cv.glmnet(X1,Y1,family = "gaussian",alpha=1,nfolds=m)
  best_lambdas_m_glmnet[i]<-results2$lambda.min
}

##### By My_CV_Lasso_Variant
set.seed(886)
best_lambdas_m_varitant<-rep(0,length(m1))
for(i in seq_along(m1)){
  m<-m1[i]
  results3<-Variant_My_cv_Lasso(X1,Y1,m,mu=5)
  best_lambdas_m_varitant[i]<-results3$best_lambda_picked
}


lambda_fold_deve<-as.data.frame(cbind(m1,
                                      best_lambdas_m_glmnet,
                                      best_lambdas_m,
                                      best_lambdas_m_varitant))

lambda_fold_deve%>%
  summary()
```

```
df10<-melt(lambda_fold_deve, id.var = "m1")
df10$variable<-factor(df10$variable)
df10<-df10%>%
  mutate(variable=factor(variable,levels=c("best_lambdas_m_glmnet",
                                            "best_lambdas_m","best_lambdas_m_varitant"),
                         labels=c("cv.glmnet","My_cv_Lasso","Variant_My_cv_Lasso")))
df10%>%
  ggplot(aes(x=m1,y=value,col=variable))+geom_point()+
  geom_line()+theme_stata(scheme = "s1color")+
  labs(x="Number of Folds",
       y="Best Lambda Found",
       title="Best Lambda Found by cv.glmnet, My_cv_Lasoo and Varitant_My_cv_Lasoo",
       col="By")



############# Monte Carlo for the Female Set:

set.seed(886)
best_lam_glment_mc<-rep(0,500)
for(i in 1:length(best_lam_glment_mc)){
  cv_glmnet_fit<-cv.glmnet(X1,Y1,family = "gaussian",
                          alpha=1,nfolds=5)
  best_lam_glment_mc[i]<-cv_glmnet_fit$lambda.min
```

```
}
par(mfrow=c(1,3))
hist(best_lam_glment_mc,
     main="Best Lambda by cv.glmnet m=5",
     xlab="Lambda")
abline(v=mean(best_lam_glment_mc),
       col = "blue")
abline(v=quantile(best_lam_glment_mc,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_glment_mc,
                  prob=c(0.025,0.975))[2],
       col = "green")

#####################################################

set.seed(886)
best_lam_glment_mc_10<-rep(0,500)
for(i in 1:length(best_lam_glment_mc)){
  cv_glmnet_fit<-cv.glmnet(X1,Y1,
                           family = "gaussian",
                           alpha=1,nfolds=10)
  best_lam_glment_mc_10[i]<-cv_glmnet_fit$lambda.min
}
hist(best_lam_glment_mc_10,
     main="Best Lambda by cv.glmnet m=10",
     xlab="Lambda")
```

```
abline(v=mean(best_lam_glment_mc_10),
       col = "blue")
abline(v=quantile(best_lam_glment_mc_10,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_glment_mc_10,
                  prob=c(0.025,0.975))[2],
       col = "green")
#################m=15
set.seed(886)
best_lam_glment_mc_15<-rep(0,500)
for(i in 1:length(best_lam_glment_mc_15)){
  cv_glmnet_fit_m15<-cv.glmnet(X1,Y1,
                               family = "gaussian",
                               alpha=1,nfolds=15)
  best_lam_glment_mc_15[i]<-cv_glmnet_fit_m15$lambda.min
}
hist(best_lam_glment_mc_15,
     main="Best Lambda by cv.glmnet m=15",
     xlab="Lambda")
abline(v=mean(best_lam_glment_mc_15),
       col = "blue")
abline(v=quantile(best_lam_glment_mc_15,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_glment_mc_15,
                  prob=c(0.025,0.975))[2],
```

```
       col = "green")


###################################
##### My_cv_Lasso Monte Carlo Simulation
set.seed(886)
best_lam_mc_my_cv_lasso_m5<-rep(0,500)
for(i in 1:length(best_lam_mc_my_cv_lasso_m5)){
  my_cv_glmnet_fitm5<-My_cv_Lasso(X1,Y1,5)
  best_lam_mc_my_cv_lasso_m5[i]<-my_cv_glmnet_fitm5$best_lambda_picked
}
par(mfrow=c(1,3))
hist(best_lam_mc_my_cv_lasso_m5,
     main="Best Lambda by My_cv_Lasso m=5",
     xlab="Lambda")
abline(v=mean(best_lam_mc_my_cv_lasso_m5),
       col = "blue")
abline(v=quantile(best_lam_mc_my_cv_lasso_m5,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_mc_my_cv_lasso_m5,
                  prob=c(0.025,0.975))[2],
       col = "green")



######################m=10
set.seed(886)
best_lam_mc_my_cv_lasso_m10<-rep(0,500)
```

```r
for(i in 1:length(best_lam_mc_my_cv_lasso_m10)){
  my_cv_glmnet_fit_m10<-My_cv_Lasso(X1,Y1,10)
  best_lam_mc_my_cv_lasso_m10[i]<-my_cv_glmnet_fit_m10$best_lambda_picked
}
hist(best_lam_mc_my_cv_lasso_m10,
     main="Best Lambda by My_cv_Lasso m=10",
     xlab="Lambda")
abline(v=mean(best_lam_mc_my_cv_lasso_m10),
       col = "blue")
abline(v=quantile(best_lam_mc_my_cv_lasso_m10,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_mc_my_cv_lasso_m10,
                  prob=c(0.025,0.975))[2],
       col = "green")


##################m=15
set.seed(886)
best_lam_mc_my_cv_lasso_m15<-rep(0,500)
for(i in 1:length(best_lam_mc_my_cv_lasso_m15)){
  my_cv_glmnet_fit_m15<-My_cv_Lasso(X1,Y1,15)
  best_lam_mc_my_cv_lasso_m15[i]<-my_cv_glmnet_fit_m15$best_lambda_picked
}
hist(best_lam_mc_my_cv_lasso_m15,
     main="Best Lambda by My_cv_Lasso m=10",
     xlab="Lambda")
```

```
abline(v=mean(best_lam_mc_my_cv_lasso_m15),
       col = "blue")
abline(v=quantile(best_lam_mc_my_cv_lasso_m15,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_mc_my_cv_lasso_m15,
                  prob=c(0.025,0.975))[2],
       col = "green")




########################################################
set.seed(886)
best_lam_varitant_mc_my_cv_lasso_m5<-rep(0,500)
for(i in 1:length(best_lam_varitant_mc_my_cv_lasso_m5)){
  my_varitant_cv_glmnet_fitm5<-Variant_My_cv_Lasso(X1,Y1,5,5)
  best_lam_varitant_mc_my_cv_lasso_m5[i]<-my_varitant_cv_glmnet_fitm5$best_lambda_picked
}
par(mfrow=c(1,3))
hist(best_lam_varitant_mc_my_cv_lasso_m5,
     main="Best Lambda by Varitant_My_cv_Lasso m=5",
     xlab="Lambda")
abline(v=mean(best_lam_varitant_mc_my_cv_lasso_m5),
       col = "blue")
abline(v=quantile(best_lam_varitant_mc_my_cv_lasso_m5,
                  prob=c(0.025,0.975))[1],
```

```
       col = "red")
abline(v=quantile(best_lam_varitant_mc_my_cv_lasso_m5,
                  prob=c(0.025,0.975))[2],
       col = "green")
#########m=10
set.seed(886)
best_lam_varitant_mc_my_cv_lasso_m10<-rep(0,500)
for(i in 1:length(best_lam_varitant_mc_my_cv_lasso_m10)){
  my_varitant_cv_glmnet_fitm10<-Variant_My_cv_Lasso(X1,Y1,10,5)
  best_lam_varitant_mc_my_cv_lasso_m10[i]<-my_varitant_cv_glmnet_fitm10$best_lambda_pick
}
hist(best_lam_varitant_mc_my_cv_lasso_m10,
     main="Best Lambda by Varitant_My_cv_Lasso m=10",
     xlab="Lambda")
abline(v=mean(best_lam_varitant_mc_my_cv_lasso_m10),
       col = "blue")
abline(v=quantile(best_lam_varitant_mc_my_cv_lasso_m10,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_varitant_mc_my_cv_lasso_m10,
                  prob=c(0.025,0.975))[2],
       col = "green")

####################m=15
set.seed(886)
best_lam_varitant_mc_my_cv_lasso_m15<-rep(0,500)
for(i in 1:length(best_lam_varitant_mc_my_cv_lasso_m15)){
```

```
  my_varitant_cv_glmnet_fitm15<-Variant_My_cv_Lasso(X1,Y1,15,5)
  best_lam_varitant_mc_my_cv_lasso_m15[i]<-my_varitant_cv_glmnet_fitm15$best_lambda_pick
}
hist(best_lam_varitant_mc_my_cv_lasso_m15,
     main="Best Lambda by Varitant_My_cv_Lasso m=15",
     xlab="Lambda")
abline(v=mean(best_lam_varitant_mc_my_cv_lasso_m15),
       col = "blue")
abline(v=quantile(best_lam_varitant_mc_my_cv_lasso_m15,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_varitant_mc_my_cv_lasso_m15,
                  prob=c(0.025,0.975))[2],
       col = "green")




##############################
#mean(best_lam_glment_mc)
#mean(best_lam_glment_mc_10)
#mean(best_lam_glment_mc_15)
#mean(best_lam_mc_my_cv_lasso_m5)
#mean(best_lam_mc_my_cv_lasso_m10)
#mean(best_lam_mc_my_cv_lasso_m15)
#mean(best_lam_varitant_mc_my_cv_lasso_m5)
```

```
#mean(best_lam_varitant_mc_my_cv_lasso_m10)
#mean(best_lam_varitant_mc_my_cv_lasso_m15)



S1<-c(0.0760686,0.08039263,0.07978809,0.0765162,
      0.08057883,0.08102547,0.07735165,0.08033728,0.08015581)
mod_lasso_female<-glmnet(X1,Y1,alpha=1,family="gaussian",lambda=S1)
#### Which variables are kept
#mod_lasso_female%>%
#  coef()
pred_female<-predict.glmnet(mod_lasso_female,newx=X1,s=S1,type="response")
mse_female<-apply(pred_female,2, function(pred_female_col) mean((pred_female_col-Y1)^2))
mse_female
min(mse_female)
pred_female_s<-predict.glmnet(mod_lasso_female,newx=X1,s=S1,type="nonzero")
pred_female_s



####################
### Checking Elapsed Time
time_1<-rep(0,10)
time_2<-rep(0,10)
time_3<-rep(0,10)
for(i in 1:10){
  time_1[i]<-system.time(
    female_glmnet<-cv.glmnet(X1,Y1,family="gaussian",alpha=1,nfolds=10)
    )[3]
```

```
  time_2[i]<-system.time(
    female_my_cv_lasso<-My_cv_Lasso(X1,Y1,10)
  )[3]
  time_3<-system.time(
    female_varitant_my_cv_lasso<-Variant_My_cv_Lasso(X1,Y1,10,5)
  )[3]
}
mean(time_1)
mean(time_2)
mean(time_3)

#################################

####################################
############################
############################
##### Testing set (Male Set)
df4<-df2[df2$sex == 1,]
df4<-df4[,-25]
X2<-as.matrix(df4[,names(df4) != "wgt"])
Y2<-df4$wgt
mod_lasso2<-glmnet(X2,Y2,alpha=1,family="gaussian")
plot(mod_lasso2,label = TRUE,xvar = "lambda")
mod_cv_3<-cv.glmnet(X2,Y2,family = "gaussian",alpha=1,nfolds=10)
plot(mod_cv_3)
mod_cv_3
coef(mod_cv_3)
```

```
######## Lambda depending on m
set.seed(7777)
m1<-3:20
best_lambdas_m_male_cv_glmnet<-rep(0,length(m1))
for(i in seq_along(m1)){
  m<-m1[i]
  results4<-cv.glmnet(X2,Y2,family = "gaussian",alpha=1,nfolds=m)
  best_lambdas_m_male_cv_glmnet[i]<-results4$lambda.min
}

set.seed(7777)
best_lambdas_male_my_cv<-rep(0,length(m1))
for(i in seq_along(m1)){
  m<-m1[i]
  results5<-My_cv_Lasso(X2,Y2,m)
  best_lambdas_male_my_cv[i]<-results5$best_lambda_picked
}
set.seed(7777)
best_lambdas_male_varitant<-rep(0,length(m1))
for(i in seq_along(m1)){
  m<-m1[i]
  results6<-Variant_My_cv_Lasso(X2,Y2,m,mu=5)
  best_lambdas_male_varitant[i]<-results6$best_lambda_picked
}


lambda_fold_test<-as.data.frame(cbind(m1,
```

```
                                              best_lambdas_m_male_cv_glmnet,
                                              best_lambdas_male_my_cv,
                                              best_lambdas_male_varitant))


df11<-melt(lambda_fold_test, id.var = "m1")
df11$variable<-factor(df11$variable)
df11<-df11%>%
  mutate(variable=factor(variable,levels=c("best_lambdas_m_male_cv_glmnet",
                                           "best_lambdas_male_my_cv","best_lambdas_male_
                         labels=c("cv.glmnet","My_cv_Lasso","Variant_My_cv_Lasso")))
df11%>%
  ggplot(aes(x=m1,y=value,col=variable))+geom_point()+
  geom_line()+theme_stata(scheme = "s1color")+
  labs(x="Number of Folds",
       y="Best Lambda Found",
       title="Best Lambda Found by cv.glmnet, My_cv_Lasoo and Varitant_My_cv_Lasoo",
       col="By")


#################
set.seed(777)
res4<-Variant_My_cv_Lasso(X2,Y2,10,5)
plot(log(res4$candidate_for_lambda), res4$mean_loss, xlab="Log Lambda",
     ylab="Mean Squared Error")
abline(v=log(res4$best_lambda_picked),col="red")
############
```

```
par(mfrow=c(1,2))
hist(mod_cv_3$lambda,main="Lambda_{3}",xlab="Lambda")
hist(res4$candidate_for_lambda,main="Lambda_{4}",xlab="Lambda")
min(res4$mean_loss)
mod_cv_3

####################
### Monte Carlo
####m=5
set.seed(777)
best_lam_glment_mc_male_m5<-rep(0,500)
for(i in 1:length(best_lam_glment_mc_male_m5)){
  cv_glmnet_fit_male_m5<-cv.glmnet(X2,Y2,family = "gaussian",
                              alpha=1,nfolds=5)
  best_lam_glment_mc_male_m5[i]<-cv_glmnet_fit_male_m5$lambda.min
}
par(mfrow=c(1,3))
hist(best_lam_glment_mc_male_m5,
     main="Best Lambda by cv.glmnet m=5",
     xlab="Lambda")
abline(v=mean(best_lam_glment_mc_male_m5),
       col = "blue")
abline(v=quantile(best_lam_glment_mc_male_m5,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_glment_mc_male_m5,
                  prob=c(0.025,0.975))[2],
```

```
       col = "green")
########m=10
set.seed(777)
best_lam_glment_mc_male_m10<-rep(0,500)
for(i in 1:length(best_lam_glment_mc_male_m10)){
  cv_glmnet_fit_male_m10<-cv.glmnet(X2,Y2,family = "gaussian",
                                  alpha=1,nfolds=10)
  best_lam_glment_mc_male_m10[i]<-cv_glmnet_fit_male_m10$lambda.min
}
hist(best_lam_glment_mc_male_m10,
     main="Best Lambda by cv.glmnet m=10",
     xlab="Lambda")
abline(v=mean(best_lam_glment_mc_male_m10),
       col = "blue")
abline(v=quantile(best_lam_glment_mc_male_m10,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_glment_mc_male_m10,
                  prob=c(0.025,0.975))[2],
       col = "green")

####################m=15
set.seed(777)
best_lam_glment_mc_male_m15<-rep(0,500)
for(i in 1:length(best_lam_glment_mc_male_m15)){
  cv_glmnet_fit_male_m15<-cv.glmnet(X2,Y2,family = "gaussian",
                                   alpha=1,nfolds=15)
```

```
  best_lam_glment_mc_male_m15[i]<-cv_glmnet_fit_male_m15$lambda.min
}
hist(best_lam_glment_mc_male_m15,
     main="Best Lambda by cv.glmnet m=15",
     xlab="Lambda")
abline(v=mean(best_lam_glment_mc_male_m15),
       col = "blue")
abline(v=quantile(best_lam_glment_mc_male_m15,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(best_lam_glment_mc_male_m15,
                  prob=c(0.025,0.975))[2],
       col = "green")
#############################
summary(best_lam_glment_mc_male_m10)
######### My_cv_Lasso
set.seed(777)
lam_mc_my_cv_lasso_male_m5<-rep(0,500)
for(i in 1:length(lam_mc_my_cv_lasso_male_m5)){
  my_cv_glmnet_fit_male5<-My_cv_Lasso(X2,Y2,5)
  lam_mc_my_cv_lasso_male_m5[i]<-my_cv_glmnet_fit_male5$best_lambda_picked
}
par(mfrow=c(1,3))
hist(lam_mc_my_cv_lasso_male_m5,
     main="Best Lambda by My_cv_Lasso m=5",
     xlab="Lambda")
abline(v=mean(lam_mc_my_cv_lasso_male_m5),
```

```
        col = "blue")
abline(v=quantile(lam_mc_my_cv_lasso_male_m5,
                  prob=c(0.025,0.975))[1],
        col = "red")
abline(v=quantile(lam_mc_my_cv_lasso_male_m5,
                  prob=c(0.025,0.975))[2],
        col = "green")


######################
set.seed(777)
lam_mc_my_cv_lasso_male_m10<-rep(0,500)
for(i in 1:length(lam_mc_my_cv_lasso_male_m10)){
  my_cv_glmnet_fit_male10<-My_cv_Lasso(X2,Y2,10)
  lam_mc_my_cv_lasso_male_m10[i]<-my_cv_glmnet_fit_male10$best_lambda_picked
}
hist(lam_mc_my_cv_lasso_male_m10,
     main="Best Lambda by My_cv_Lasso m=10",
     xlab="Lambda")
abline(v=mean(lam_mc_my_cv_lasso_male_m10),
        col = "blue")
abline(v=quantile(lam_mc_my_cv_lasso_male_m10,
                  prob=c(0.025,0.975))[1],
        col = "red")
abline(v=quantile(lam_mc_my_cv_lasso_male_m10,
                  prob=c(0.025,0.975))[2],
        col = "green")
```

```
###############################m=15
set.seed(777)
lam_mc_my_cv_lasso_male_m15<-rep(0,500)
for(i in 1:length(lam_mc_my_cv_lasso_male_m15)){
  my_cv_glmnet_fit_male15<-My_cv_Lasso(X2,Y2,15)
  lam_mc_my_cv_lasso_male_m15[i]<-my_cv_glmnet_fit_male15$best_lambda_picked
}
hist(lam_mc_my_cv_lasso_male_m15,
     main="Best Lambda by My_cv_Lasso m=15",
     xlab="Lambda")
abline(v=mean(lam_mc_my_cv_lasso_male_m15),
       col = "blue")
abline(v=quantile(lam_mc_my_cv_lasso_male_m15,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(lam_mc_my_cv_lasso_male_m15,
                  prob=c(0.025,0.975))[2],
       col = "green")

#####################################################
######### Variant_My_cv_Lasso m=5
set.seed(777)
lammc_v_my_cv_lasso_male_m5<-rep(0,500)
for(i in 1:length(lammc_v_my_cv_lasso_male_m5)){
  v_my_cv_glmnet_fit_male5<-Variant_My_cv_Lasso(X2,Y2,5,5)
  lammc_v_my_cv_lasso_male_m5[i]<-v_my_cv_glmnet_fit_male5$best_lambda_picked
```

```
}
par(mfrow=c(1,3))
hist(lammc_v_my_cv_lasso_male_m5,
     main="Best Lambda by Variant_My_cv_Lasso m=5",
     xlab="Lambda")
abline(v=mean(lammc_v_my_cv_lasso_male_m5),
       col = "blue")
abline(v=quantile(lammc_v_my_cv_lasso_male_m5,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(lammc_v_my_cv_lasso_male_m5,
                  prob=c(0.025,0.975))[2],
       col = "green")


###################### m=10
set.seed(777)
lammc_v_my_cv_lasso_male_m10<-rep(0,500)
for(i in 1:length(lammc_v_my_cv_lasso_male_m10)){
  v_my_cv_glmnet_fit_male10<-Variant_My_cv_Lasso(X2,Y2,10,5)
  lammc_v_my_cv_lasso_male_m10[i]<-v_my_cv_glmnet_fit_male10$best_lambda_picked
}
hist(lammc_v_my_cv_lasso_male_m10,
     main="Best Lambda by Variant_My_cv_Lasso m=10",
     xlab="Lambda")
abline(v=mean(lammc_v_my_cv_lasso_male_m10),
       col = "blue")
```

```
abline(v=quantile(lammc_v_my_cv_lasso_male_m10,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(lammc_v_my_cv_lasso_male_m10,
                  prob=c(0.025,0.975))[2],
       col = "green")

#########################m=15
set.seed(777)
lammc_v_my_cv_lasso_male_m15<-rep(0,500)
for(i in 1:length(lammc_v_my_cv_lasso_male_m15)){
  v_my_cv_glmnet_fit_male15<-Variant_My_cv_Lasso(X2,Y2,15,5)
  lammc_v_my_cv_lasso_male_m15[i]<-v_my_cv_glmnet_fit_male15$best_lambda_picked
}
hist(lammc_v_my_cv_lasso_male_m15,
     main="Best Lambda by Variant_My_cv_Lasso m=15",
     xlab="Lambda")
abline(v=mean(lammc_v_my_cv_lasso_male_m15),
       col = "blue")
abline(v=quantile(lammc_v_my_cv_lasso_male_m15,
                  prob=c(0.025,0.975))[1],
       col = "red")
abline(v=quantile(lammc_v_my_cv_lasso_male_m15,
                  prob=c(0.025,0.975))[2],
       col = "green")
```

```
######## Compare MSE
#### when using different lambdas
#mean(best_lam_glment_mc_male_m5)
#mean(best_lam_glment_mc_male_m10)
#mean(best_lam_glment_mc_male_m15)
#mean(lam_mc_my_cv_lasso_male_m5)
#mean(lam_mc_my_cv_lasso_male_m10)
#mean(lam_mc_my_cv_lasso_male_m15)
#mean(lammc_v_my_cv_lasso_male_m5)
#mean(lammc_v_my_cv_lasso_male_m10)
#mean(lammc_v_my_cv_lasso_male_m15)
##################################
####################################
S2<-c(0.06802671,0.06917023,0.07191867,0.06902735,
      0.0701646,0.07374311,0.06998219,0.07005154,0.07265663)
mod_lasso_male<-glmnet(X2,Y2,alpha=1,family="gaussian",lambda=S2)
####### glmnet takes ascending order
#### Which variables are kept
mod_lasso_male%>%
  coef()
pred_male<-predict.glmnet(mod_lasso_male,newx=X2,s=S2,type="response")
mse_male<-apply(pred_male,2, function(pred_male_col) mean((pred_male_col-Y2)^2))
mse_male
min(mse_male)
pred_male_s<-predict.glmnet(mod_lasso_male,newx=X2,s=S2,type="nonzero")
pred_male_s
```

```r
######################
time_4<-rep(0,10)
time_5<-rep(0,10)
time_6<-rep(0,10)
for(i in 1:10){
  time_4[i]<-system.time(
    male_glmnet<-cv.glmnet(X2,Y2,family="gaussian",alpha=1,nfolds=10)
  )[3]
  time_5[i]<-system.time(
    male_my_cv_lasso<-My_cv_Lasso(X2,Y2,10)
  )[3]
  time_6<-system.time(
    male_varitant_my_cv_lasso<-Variant_My_cv_Lasso(X2,Y2,10,5)
  )[3]
}
mean(time_4)
mean(time_5)
mean(time_6)
```