Statistical Learning Project

Ammar Hasan 150454388

25 November 2018

Contents

1 Introduction **2**

2 Exploratory Data Analysis **2**

2.1 Data Spread and Location 2

2.2 Data Relationships 3

3 Unsupervised Learning **4**

3.1 Principle Component Analysis 4

4 Supervised Learning (Linear Modelling) **5**

4.1 Model Fitting 5

4.2 Model Evaluation Using k-fold Cross Validation and Best Model 7

5 Appendix **8**

5.1 A.1 Abbrevations and Shorthands 8

5.2 A.2 Functions 9

5.3 A.3 Tables 10

5.4 A.4 Plots 17

1 Introduction

This report summaries the steps undertaken to produce and evaluate linear regression models of the value of housing in Boston Standard Metropolitan. The model will be used to predict the value of logarithmic crime rate (lcrim) using other variables. The model would be built after exploratory and unsupervised statistical analysis of the data, which are carried first to gain an understanding on the data characteristics and structure before hand. The models would be build using both Subsetting (Best Fit) and Regularisation methods (LASSO and Ridge Fit) methods, these methods will be compared using k Fold Cross Validation with $k = 10$.

Any output (tables and plots) is placed in the Appendix at the end of the report with the R code that generated it. The description and analysis of the methods is found in the document body which cross references the appendix content.

2 Exploratory Data Analysis

Before building a linear model, it is wise to first understand the overall structure of the data itself to get a feeling for the data characteristics and how the variables relate to one another - especially how they relate to the response variable (lcrime).

2.1 Data Spread and Location

To analyse the distribution of the data, the quantiles and means will be examined. In particular, this part of the report will examine outliers, scale, consistency and certainty.

2.1.1 Mean Vector

A vector of mean averages was produced for all the predictors and the response variables in the Boston data-set and is shown in table 1, and shows that the means are well spread out from one another, which suggests a difference in the nature and scale of the measurements.

Examining the structure of the variables in the Boston data set using the ?Boston confirms that the nature of the measurements vary from Full-value property-tax rate per $10,000 for tax to Nitrogen oxides parts per 10 million for nox for instance, which obviously suggests that the measurements cannot be directly compared scale to scale as this might cause values with larger scale to dominate (standarisation might be required).

2.1.2 Box Plot and Quantiles

As previously stated in the last section, the variables are of different natures and scales, hence any scale to scale comparison needs standarisation. To standardise the data a scale transform was applied, and the variables spread was represented using a box plot in figure 1 is generated.

The following stands out of the plot:

- black, rm, zn and medv predictor variables have significantly more outliers than the other variables.And hence more uncertain and their averages can get skewed.
- chas predictor variable seems to have a very tight ranges that are practically identical. This is because this is a binary variable (as ?Boston would show).
- zn, rad and tax predictor variables have a short Q1 to Q2 range compared to the Q2 to Q3 range, suggesting that the lower values of the data are very tightly clustered. black has the opposite problem.

- rm, lstat, mdev and ptratio have long minimum and maximum ranges in comparison to their IQR ranges, and hence more extreme values. This means that their averages can get skewed.

2.2 Data Relationships

This section of the report will look into how the variables (predictors and response) in bivariate data relate and interact using numerical correlation matrices and graphic pairs plot

2.2.1 Pairs Plot

The following relationships stick out when observing the pairs plot for the response against other variables in figure 2:

- Relationships/correlation with lcrime:
  - age and medv have a moderate negative relationship with lcrime
  - nox and lstat have a strong/moderate positive relationship with lcrim
  - rm and zn both have weak/moderate negative relationships with lcrime
  - tax, rad, pratio, indus and black appear to have unclear correlation from the plot.
- Predictor variables relationships:
  - Some predictors have strong relations with one another: rm has a strong/moderate negative relationship with lstat but a strong positive one with medv. medv and lstat also have a strong/moderate negative relationship
- Chas (river dummy variable) and disf (distance to Boston employment centered) appear to have values in levels and are also difficult to analyse in a pairs plot

2.2.2 Correlation Matricies

The correlation matrix in table 2 confirms the findings of the previous section but also helps clarify some of relationships that were unclear before:

- Chas has a very weak relationship with lcrime, and a weak or very weak relationship with most other variables.
- disf has a moderate negative relationship with lcrime, a strong/moderate negative relationship with indus and a positive moderate relation with zn
- Tax and rad have strong relationships with lcrime that were difficult to spot before due to irregularities in their plots. Moreover, indus has a moderate/strong positive relationship with lcrime and ptartio a weak positive one.
- Tax and rad have a very strong positive relationship

2.2.3 Data Relationships Summary

To summarise, it seems that the relationships suggest that there are a couple of variables that might have a strong impact to model (e.g. rad or tax). Moreover, the relationships also suggest that many will be subsetted due to relationships that can be represented with other variables (e.g. rad for tax, or vice avers) or because of very poor relationships with all variables (e.g. chas).

3 Unsupervised Learning

This section looks into applying unsupervised learning techniques to help in understanding patterns and structures in the data to interpret their effects on the model. In this report, Principle Component Analysis will be used to help find which set of predictors cause the most variation on the data to help with model coefficient interpretation.

3.1 Principle Component Analysis

3.1.1 Variation Proportions

Table 3 shows the summary of PCA run on the data, showing the variance each PC to and the accumulation of it. The summary shows that the first component contributed to 50% of the variance, and the first 4 contribute to 70%. Since the first 3 contribute to 70% and also where the variation curve in the scree plot in figure 3 diminishes for a second time, the first 3 PCs will be analysed.

3.1.2 Component Interpretation (According to Table 4)

3.1.2.1 Component 1

Dominated by positive lcrime, tax, indus and nox. This means that it represents areas with large non-retail business but high crime and nitrogen oxide pollution.

3.1.2.2 Component 2

Dominated by negative rm and medv, meaning that it represents less median house values and number of rooms.

3.1.2.3 Component 3

Dominated high accessibility highways (rad), tax rate (tax) and residual areas (zn).

3.1.2.4 Plot

A plot of the observations scores for component 1 vs component 2 is shown in figure 4 shows most observations score high for component 2, but are more varied for component 1. Meaning that most observations have relative low median house value and number of rooms, but a varied crime, pollution and non-retail business.

4 Supervised Learning (Linear Modelling)

In this section of the report linear models are fitting and tested against each other using K Fold Cross Validation, where $k = 10$ as it is usually the common choice and the default for the glmnet function. By the end of this section a summary will conclude what model is best and why.

4.1 Model Fitting

4.1.1 Subset Selection using Best Fit

Best Fit Subset Selection Tries all possible predictors p combinations for every number of predictors to find the "best" model using SS_E (optimisation problem). The method's results are shown in table 5 as the selected best predictors from 13 variables (everything except lcrime) to 1 variable. The method was carried on scaled predictors because as explained in the exploratory data analysis the scales differ, which might cause the coefficients be to harder to compare as the values with the larger scales would dominate.

4.1.1.1 Subsetting Results analysis

When we look at the results we can notice that the first variable to be dropped is tax. tax being the first dropped predictor seems to lines up with the discussion in the Data Relationships Summary, as it was stated that either tax/rad can be represented by the other (and hence can be dropped).

Moreover it was also stated in the Relationships summary that chas correlated little to lcrime and other variables (little to no prediction can be made with it), so it's no surprise that it was dropped second. rm and mdev also get removed early, probably because they are both strongly/moderately correlated to lstat and can be represented by it.

Also, it can be noticed that usually the highest impacting predictors in the PCs stick for longer before dropping with a few exceptions. This particularly true for PC1, as with the exception of tax (which was subsetted for rad) some of the high scoring variables stuck around for long (e.g. nox), which is not surprising since it would be sensible for the most variation causing variables to cause variation to lcrim (and hence become important for prediction).

4.1.1.2 Which Subset to Select

Nonetheless, to decide which number of predictors would be the best choice, there are various techniques that can be followed to evaluate whether removing variables is worthwhile (SS_E based or MSE based).

The results using based SS_E measurements (Adjusted R^2, BIC, Cp) are shown in figure 5, and they show that generally the gain from dropping out predictors is optimal somewhere around 6-10 predictors before becoming worse.

For the K Fold Cross Validation (10 fold picked here) also shown in figure 5, the best result occurs with 9 predictors (is identical to the adjusted Cp choice), and since this measurement is based on experimentation with tests errors it is preferred. The 10 Fold Cross Validation choice of 9 predictors would result in chas, rm, mdev and tax being removed from the model.

4.1.1.3 Selected Subset Coefficient Discussion

The resulting coefficients for the selected predictors is shown in table 6, and seems to show nox and rad as the strongest positively influencing predictors (largest positive values with the most positive impact on the response). This lines up with nox and rad positive strength in the first PC in the PCA analysis and their strong positive correlation. The same can be observed with the strongest negatively influencing component (zn) (negative effect on response), which also were strong in PC1 and had strong negative correlation with

lcrim. The intercept represents what value (logarithmic) the lcrim predictor would take when all the other predictor values are equal to zero. And since the data is standardised, the values are equal to zero when they are the mean, which means the intercept would represent the response when all the other predictors are average.

4.1.2 Ridge Regression

Ridge Regression uses a modified loss function to add a penalty to large coefficients to improve predictive solution (constraints variance). Ridge Regression method is also fed with scaled data here like Best Fit to avoid coefficients dominating due to scale differences.

4.1.2.1 Lambda Choice

Lambda decides how much the algorithm penalties large coefficient, the more it does so the more the values slope down as seen in figure 6. Most of the values seems clustered next to each other, except rad(8), nox(4) and zn (1), with rad especially dominating the coefficient sizes before dropping next to lambda 0. Nonetheless, to know which lambda choice would make the most sense, a 10 fold cross validation is carried in figure 7 and shows that the MSE start low and only starts to pick pace and go up once it crosses $log(lambda) = 0$. The exact minimum MSE is not clear from the graph, but using the "lambda.min" field returns a ~0.006 minimum (correct to 3DP) for the lambda (as seen in figure's 7 code).

4.1.2.2 Selected Lambda Coefficent Discussion

As seen in figure 6, nox(7), zn(1) and rad(8) seemed to be the most dominant coefficients and this can also be observed in the exact values with ideal lambda (~0.006) seen in table 7. The coefficients selected seem similar to the coefficients seen with best subset selection and hence the discussion of the most dominant values can be reiterated here, but with slightly different values due to the adjusted algorithm and no subsetting (all p's). Nonetheless, just like the Best Subset coefficient, the values most correlated to lcrime and the strongest in PC1 generally being the most dominant as discussed with the Best Subset coefficients. Lastly, the intercept is similar and represents the response when all the other predictors are average.

4.1.3 LASSO

LASSO is quite similar to Ridge Regression, however unlike ridge regression which always gets all p variables LASSO can subset (coefficients can become zero). This is because of the modified SS_E which takes the absolute value instead of the square of each coefficient. LASSO is also fed with scaled data here like the other two techniques to avoid coefficients dominating due to scale differences.

4.1.3.1 Lambda Choice

Similarly to Ridge Regression, lambda decides how much the algorithm penalises large coefficients. The values of coefficients against lambda can be seen in figure 8, the dominant coefficients are similar to Ridge, but some of the coefficient eventually converge to zero. Nonetheless, as with Ridge Regression, to find the ideal lambda value MSE in a 10 fold cross validation is used, and as seen in figure 7. The shape of the error plot is slightly different and this time it returns a minimum ~0.0163 (correct to 3DP).

4.1.3.2 Selected Lambda Coefficent Discussion

The values for the coefficients are quite similar to Ridge Regression coefficients dis-including 0.00 predictors, so again, the most positively/negatively response influencing predictors are similar minus the subsetted ones and the similar intercept represents the response when all the other predictors are average. Nonetheless, since LASSO's main difference with Ridge regression is its ability to subset the predictors they will be the

main concern of this discussion. It seems LASSO decided to subset tax, medv, rm and chas (at least correct to 3 DP), which are the same as the ones subsetted by the Best Fit (with 9 predictors), which as stated in its analysis are subsetted for being weakly correlated with lcrim (chas) or being correlated with other predictors that can represent them (tax, rm and medv).

4.2 Model Evaluation Using k-fold Cross Validation and Best Model

To evaluate all the models using cross validation, it is important to ensure that they use the same observations in the folds for the cross validation. To do this, the indexes for the folds for subset cross validation (using 10 folds) are passed to the regularisation fittings (LASSO and Ridge). The results are presented in figure 10.

As seen in figure 10 the regularisation methods (LASSO and Ridge Fit) and the subsetting method (Best Fit) seem to perform similarly for the given data. Best Fit seems to be the best, then LASSO then Ridge Fit.

Nonetheless, the regularisation methods would be the better bet, since Best Fit is incredibly slow due its exhaustive methods and the advantage is that it provides in exchange is small. For the regularisation methods, since LASSO requires less predictors it should be more convenient and easier to use and hence is my choice of out of the two. LASSO had subsetted out chas, rm, tax and medv as stated before either because they had poor correlation with the response (lcrim) or could be represented by other predictors (rm, tax and medv).

Lastly, it is important to be careful when using these models, because as stated in the Box Plots and Quantiles analysis, some of the variables have a lot of extreme values and outliers and this might effect the reliability of the models. While the regularisation method penalty helps in reducing the effect of extreme values, they might still not be complete immune to their effects.

5 Appendix

This section contains all supplementary material and is divided into three sections ("Abbreviations and Short hands", "Functions", "Tables" and "Plots"). The code required to generate the supplementary material is also included

5.1 A.1 Abbrevations and Shorthands

5.1.1 Abbrevations

(X)DP: X Decimal Points

PCA: Principle Component Analysis

PC: Principle Component

SS_E: Residual Sum of Squares

MSE: Mean Square Error

R^2 or Rsq: Coefficient of Determination

C_p: Mallow's C_p

BIC: Bayesian Information Criterion

LASSO: Least Absolute Shrinkage and Selection Operator

CV: Cross Validation

p: Predictors (number of)

n: Rows/Observations (number of)

5.1.2 Variable Shorthands

lcrim: Natural logarithm of the per capita crime rate by town.

zn: Proportion of residential land zoned for lots over 25,000 sq.ft.

indus: Proportion of non-retail business acres per town.

chas: Charles River dummy variable (=1 if tract bounds river; =0 otherwise).

nox: Nitrogen oxides concentration (parts per 10 million).

rm: Average number of rooms per dwelling.

age: Proportion of owner-occupied units built prior to 1940.

disf: A numerical vector representing an ordered categorical variable with four levels depending on the weighted mean of the distances to five Boston employment centres (=1 if distance < 2.5, =2 if 2.5 <= distance < 5, =3 if 5 <= distance < 7.5, =4 if distance >= 7.5).

rad: Index of accessibility to radial highways.

tax: Full-value property-tax rate per $10,000.

pratio: Pupil-teacher ratio by town.

black: 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town.

lstat: Lower status of the population (percent).

medv: Median value of owner-occupied homes in $1000s.

5.2 A.2 Functions

```r
# Table function with rounding (uses kable)
table = function(dataset, cap, dp = 3){
   kable(format(round(dataset, dp), nsmall = dp), caption = cap)
}

# Table function (non num)
tableTxt = function(dataset, cap){
  kable(dataset, caption = cap)
}

# Predict for reg subsets (no predict function)
predict.regsubsets = function(object, newdata, id, ...) {
  form = as.formula(object$call[[2]])
  mat = model.matrix(form, newdata)
  coefi = coef(object, id=id)
  xvars = names(coefi)
  return(mat[,xvars] %*% coefi)
}
```

Table 1: Variables Mean Vector

	x
lcrim	-0.780
zn	11.364
indus	11.137
chas	0.069
nox	0.555
rm	6.285
age	31.425
disf	1.960
rad	9.549
tax	408.237
ptratio	18.456
black	356.674
lstat	12.653
medv	22.533

5.3 A.3 Tables

5.3.1 Code to Generate Table 1 (Transposed and Correct to 3DP)

```
table(colMeans(Boston), 'Variables Mean Vector')
```

Table 2: Correlation Matrix (3DP)

	lcrim	zn	indus	chas	nox	rm	age	disf	rad	tax	ptratio	black	lstat	medv
lcrim	1.000	-0.517	0.731	0.028	0.789	-0.307	-0.658	-0.683	0.853	0.828	0.390	-0.479	0.627	-0.454
zn	-0.517	1.000	-0.534	-0.043	-0.517	0.312	0.570	0.612	-0.312	-0.315	-0.392	0.176	-0.413	0.360
indus	0.731	-0.534	1.000	0.063	0.764	-0.392	-0.645	-0.727	0.595	0.721	0.383	-0.357	0.604	-0.484
chas	0.028	-0.043	0.063	1.000	0.091	0.091	-0.087	-0.082	-0.007	-0.036	-0.122	0.049	-0.054	0.175
nox	0.789	-0.517	0.764	0.091	1.000	-0.302	-0.731	-0.776	0.611	0.668	0.189	-0.380	0.591	-0.427
rm	-0.307	0.312	-0.392	0.091	-0.302	1.000	0.240	0.213	-0.210	-0.292	-0.356	0.128	-0.614	0.695
age	-0.658	0.570	-0.645	-0.087	-0.731	0.240	1.000	0.758	-0.456	-0.506	-0.262	0.274	-0.602	0.377
disf	-0.683	0.612	-0.727	-0.082	-0.776	0.213	0.758	1.000	-0.477	-0.537	-0.234	0.322	-0.511	0.291
rad	0.853	-0.312	0.595	-0.007	0.611	-0.210	-0.456	-0.477	1.000	0.910	0.465	-0.444	0.489	-0.382
tax	0.828	-0.315	0.721	-0.036	0.668	-0.292	-0.506	-0.537	0.910	1.000	0.461	-0.442	0.544	-0.469
ptratio	0.390	-0.392	0.383	-0.122	0.189	-0.356	-0.262	-0.234	0.465	0.461	1.000	-0.177	0.374	-0.508
black	-0.479	0.176	-0.357	0.049	-0.380	0.128	0.274	0.322	-0.444	-0.442	-0.177	1.000	-0.366	0.333
lstat	0.627	-0.413	0.604	-0.054	0.591	-0.614	-0.602	-0.511	0.489	0.544	0.374	-0.366	1.000	-0.738
medv	-0.454	0.360	-0.484	0.175	-0.427	0.695	0.377	0.291	-0.382	-0.469	-0.508	0.333	-0.738	1.000

5.3.2 Code to Generate Table 2 (Correct to 3DP)

```
table(cor(Boston), 'Correlation Matrix (3DP)')
```

Table 3: PCA Summary (Contribution to Variation)

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12	PC13	PC14
Standard deviation	2.645	1.286	1.118	0.934	0.927	0.809	0.638	0.588	0.500	0.460	0.434	0.389	0.322	0.233
Proportion of Variance	0.500	0.118	0.089	0.062	0.061	0.047	0.029	0.025	0.018	0.015	0.013	0.011	0.007	0.004
Cumulative Proportion	0.500	0.618	0.707	0.769	0.831	0.878	0.907	0.931	0.949	0.964	0.978	0.989	0.996	1.000

5.3.3 Code to Generate Table 3 (Correct to 3DP)

```
# Perform PCA based on the standardised data (means and data nature vary)
pca = prcomp(Boston, scale=TRUE)
table(summary(pca)$importance, 'PCA Summary (Contribution to Variation)')
```

5.3.4 Code to Generate Table 4 (Correct to 3DP)

```
# List PC 1, 2 and 3
table(pca$rotation[,1:3], 'PCA Components')
```

Table 4: PCA Components

	PC1	PC2	PC3
lcrim	0.341	-0.136	0.181
zn	-0.239	0.058	0.394
indus	0.324	-0.095	-0.070
chas	-0.001	-0.387	-0.255
nox	0.320	-0.227	-0.087
rm	-0.190	-0.492	0.285
age	-0.291	0.208	0.264
disf	-0.294	0.287	0.220
rad	0.295	-0.078	0.450
tax	0.315	-0.043	0.381
ptratio	0.196	0.331	0.116
black	-0.190	0.019	-0.378
lstat	0.297	0.238	-0.150
medv	-0.251	-0.475	0.095

5.3.5 Code to Generate Table 5

```
# Get scaled results and response
X_raw = as.matrix(Boston[-1])
X = scale(X_raw)
y = Boston[,1]

# fit model
bss = regsubsets(y ~ ., data=data.frame(y,X), method="exhaustive", nvmax= 13)

# Summarise
bssSummary = summary(bss)

tableTxt(bssSummary$outmat, "Best Subset Selection")
```

Table 5: Best Subset Selection

	zn	indus	chas	nox	rm	age	disf	rad	tax	ptratio	black	lstat	medv
1 (1)								*					
2 (1)				*				*					
3 (1)	*			*				*					
4 (1)	*			*				*				*	
5 (1)	*			*			*	*				*	
6 (1)	*			*			*	*			*	*	
7 (1)	*			*		*	*	*			*	*	
8 (1)	*			*		*	*	*		*	*	*	
9 (1)	*	*		*		*	*	*		*	*	*	
10 (1)	*	*		*		*	*	*		*	*	*	*
11 (1)	*	*		*	*	*	*	*		*	*	*	*
12 (1)	*	*	*	*	*	*	*	*		*	*	*	*
13 (1)	*	*	*	*	*	*	*	*	*	*	*	*	*

Table 6: Best Subset Coefficient Selection (9 Predictors)

	x
(Intercept)	-0.780
zn	-0.256
indus	0.101
nox	0.395
age	-0.135
disf	-0.132
rad	1.240
ptratio	-0.109
black	-0.121
lstat	0.197

5.3.6 Code to Generate Table 6

```
# use coef function to find the coefficient
# id represents the choice for predictor number
table(coef(bss, id = 9), "Best Subset Coefficient Selection (9 Predictors)")
```

5.3.7 Code to Generate Table 7

```
## Fit a ridge regression model with idea lambda (~0.006)
ridgeFit = glmnet(X, y, alpha=0, standardize=FALSE, lambda= 0.006428073)
table(as.data.frame(as.matrix(coef(ridgeFit, s = 0.006428073))),
      "Ridge Regression Coefficients with Lambda = ~0.006")
```

5.3.8 Code to Generate Table 8

```
## Fit a LASSO regression model with idea lambda (~0.016)
lassoFit = glmnet(X, y, alpha=1, standardize=FALSE, lambda= 0.01629751)
table(as.data.frame(as.matrix(coef(lassoFit, s = 0.01629751))),
      "LASSO Coefficients with Lambda = ~0.016")
```

Table 7: Ridge Regression Coefficients with Lambda = 0.006

	1
(Intercept)	-0.780
zn	-0.253
indus	0.105
chas	-0.009
nox	0.411
rm	-0.037
age	-0.139
disf	-0.118
rad	1.230
tax	-0.003
ptratio	-0.089
black	-0.132
lstat	0.220
medv	0.078

Table 8: LASSO Coefficients with Lambda = 0.016

	1
(Intercept)	-0.780
zn	-0.231
indus	0.088
chas	0.000
nox	0.423
rm	0.000
age	-0.131
disf	-0.134
rad	1.212
tax	0.000
ptratio	-0.062
black	-0.113
lstat	0.182
medv	0.000

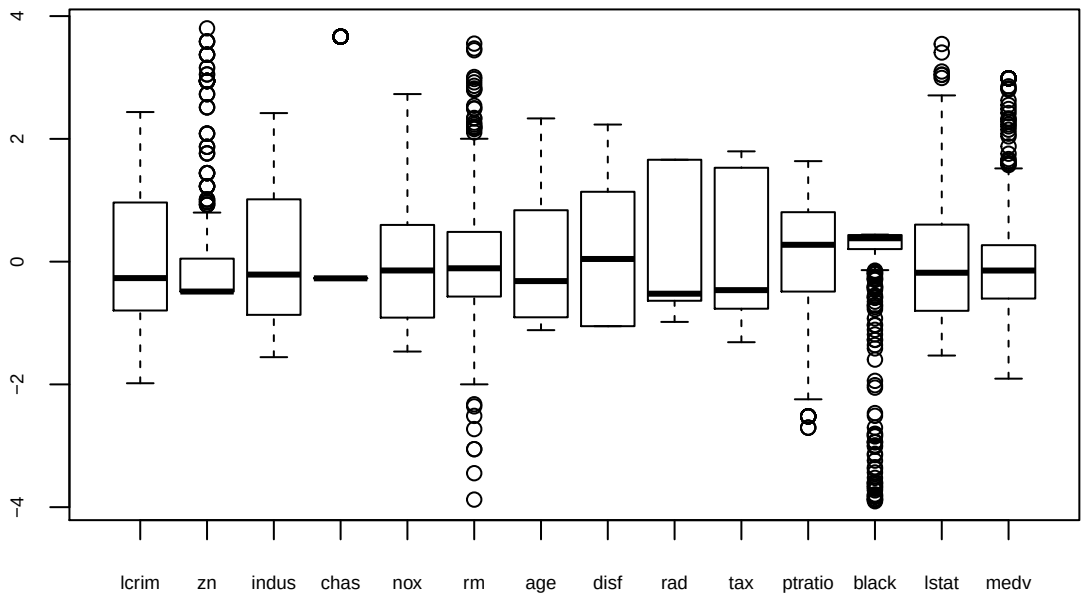

Figure 1: Box Plot

5.4 A.4 Plots

5.4.1 Code to Generate Figure 1

```
# scale transforms to deal with the variation in the nature of the measurements
boxplot(scale(Boston), cex.axis=0.6)
```

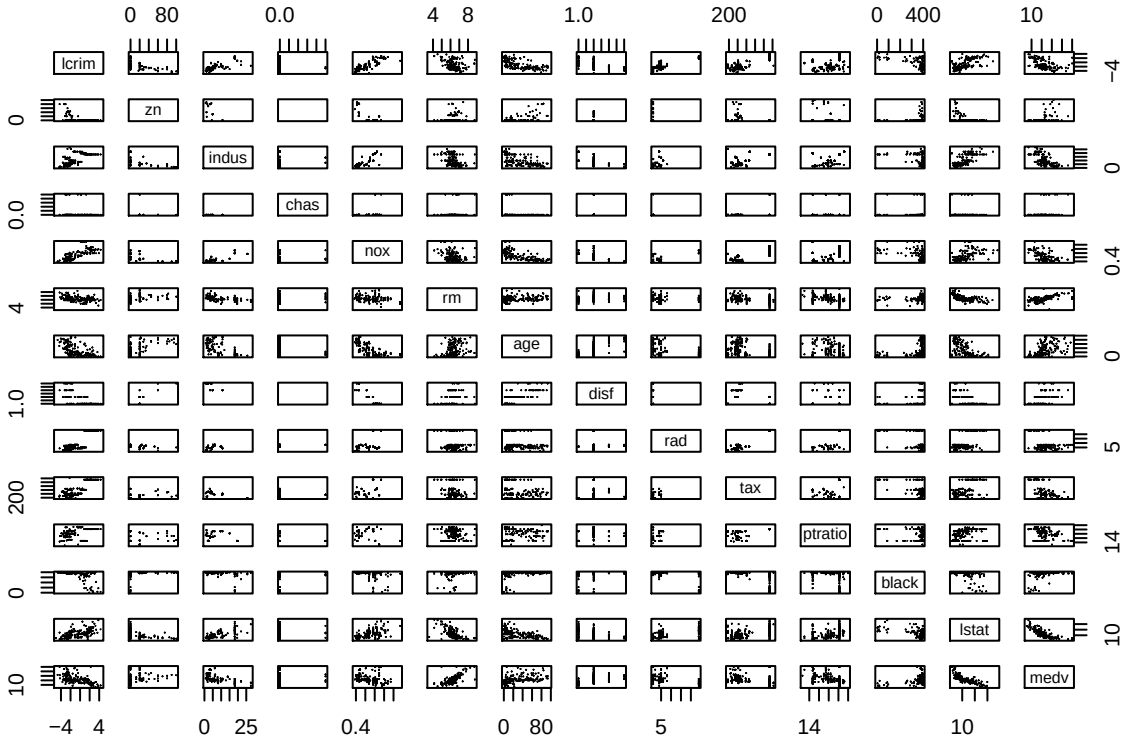

Figure 2: Pairs Plot

5.4.2 Code to Generate Figure 2

```
pairs(Boston, cex=0.0005)
```

5.4.3 Code to Generate Figure 3

```
plot(pca, type='l', main='Scree Plot for Boston Housing Values')
title(xlab='Principle Component number')
```

5.4.4 Code to Generate Figure 4

```
# Plot PCA 1 against PCA 2
plot(pca$x[,1], pca$x[,2], main ="Principle Component 1 vs 2 for Boston Housing Values",
     xlab="Component 1", ylab="Component 2")
```

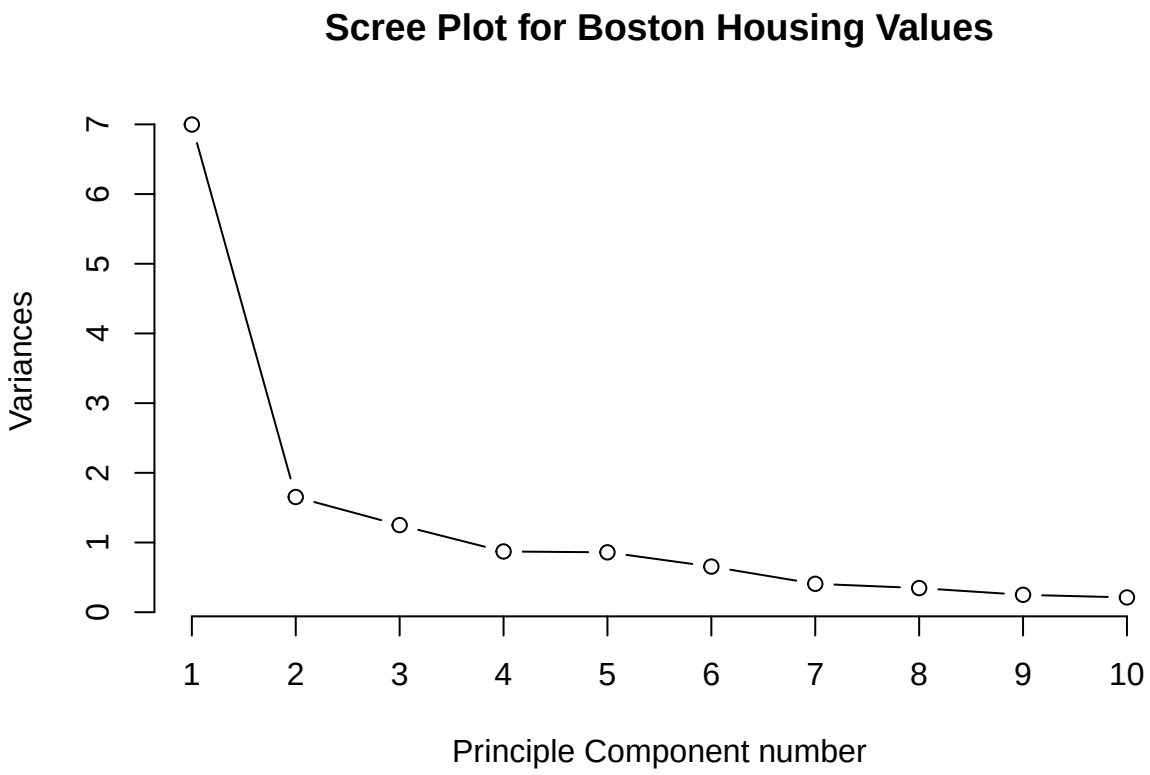

Figure 3: Scree Plot

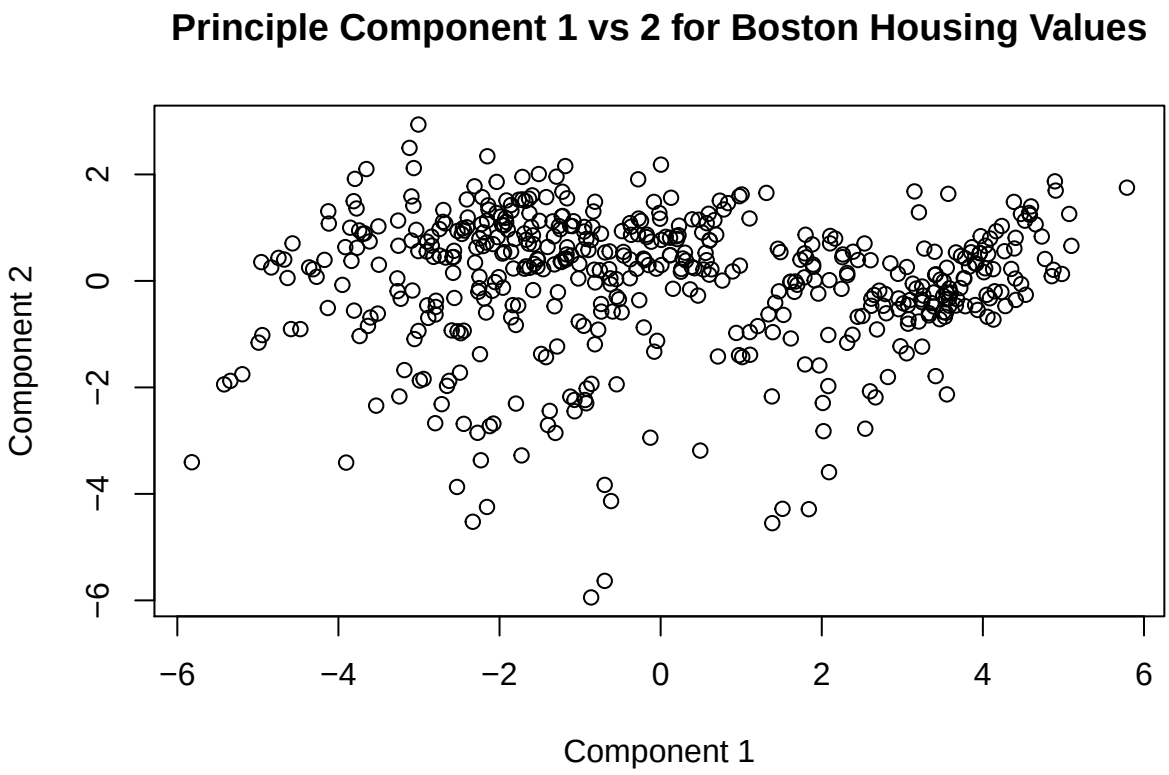

Figure 4: PCA Plot

5.4.5 Code to Generate Figure 5

```
# *** SSE Based Measurements ***

# Find best score for SSE based measurements (already done by bss summary)
bestAdjr2 = which.max(bssSummary$adjr2)
bestCp = which.min(bssSummary$cp)
bestBic = which.min(bssSummary$bic)

# *** 10 fold cross validation ***

# 10 fold cv

# Set the seed to make the analysis reproducible
set.seed(1)

# 10-fold cross validation
nFolds = 10

# Find n and p
p = ncol(Boston) - 1 # number of predictors (no lcrim)
n = nrow(Boston)

# Sample fold-assignment index
foldIndex = sample(nFolds, n, replace=TRUE)

# Hold fold sizes
foldSizes = numeric(nFolds)

# Compute fold sizes
for(k in 1:nFolds) foldSizes[k] = length(which(foldIndex==k))

# create the matrix to store the folds
cvBssErrors = matrix(NA, p, nFolds)

# Find MSEs for all fold combinations
for(k in 1:nFolds) {

  # Fit models by best-subset selection (no k-th fold)
  bssTmpFit =
    regsubsets(lcrim ~ ., data=Boston[foldIndex!=k,], method="exhaustive", nvmax=p)

  # For each model M_m where m=1,...,p:
  for(m in 1:p) {
    # Compute fitted values for the k-th fold
    bssTmpPredict = predict(bssTmpFit, Boston[foldIndex==k,], m)
    # Work out MSE for the k-th fold
    cvBssErrors[m, k] = mean((Boston[foldIndex==k,]$lcrim - bssTmpPredict)^2)
  }
}

# Compute a weighted average MSE
bssMse = numeric(p)
```

```r
# For models M_1,...,M_p:
for(m in 1:p) {
  bssMse[m] = weighted.mean(cvBssErrors[m,], w=foldSizes)
}

# Identify model with the lowest MSE
bestCv = which.min(bssMse)

# *** Plotting ***

# Create multi-panel plotting device:
par(mfrow=c(2,2))

# Produce plots, highlighting optimal value of preductors:
plot(1:13, bssSummary$adjr2, xlab="Number of predictors", ylab="Adjusted Rsq",
type="b")
points(bestAdjr2, bssSummary$adjr2[bestAdjr2], col="red", pch=16)

plot(1:13, bssSummary$cp, xlab="Number of predictors", ylab="Cp", type="b")
points(bestCp, bssSummary$cp[bestCp], col="red", pch=16)

plot(1:13, bssSummary$bic, xlab="Number of predictors", ylab="BIC", type="b")
points(bestBic, bssSummary$bic[bestBic], col="red", pch=16)

plot(1:p, bssMse, xlab="Number of predictors", ylab="10-fold CV Error", type="b")
points(bestCv, bssMse[bestCv], col="red", pch=16)
```

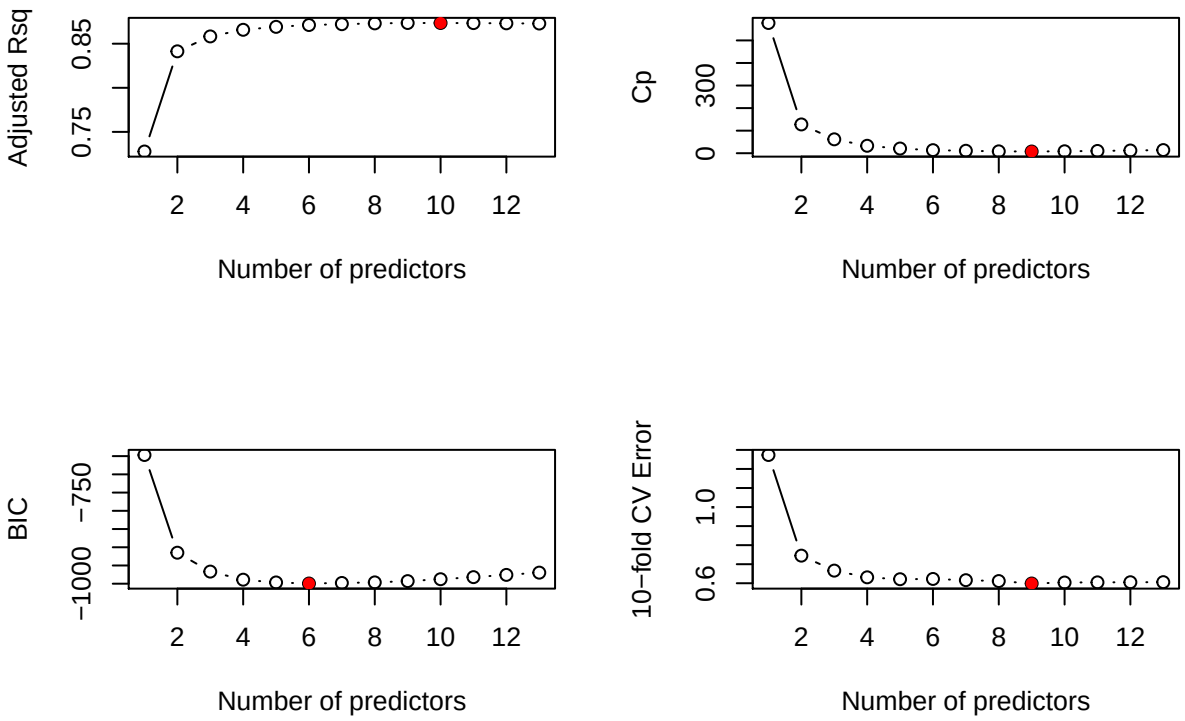

Figure 5: Best Subset Predictor Selection

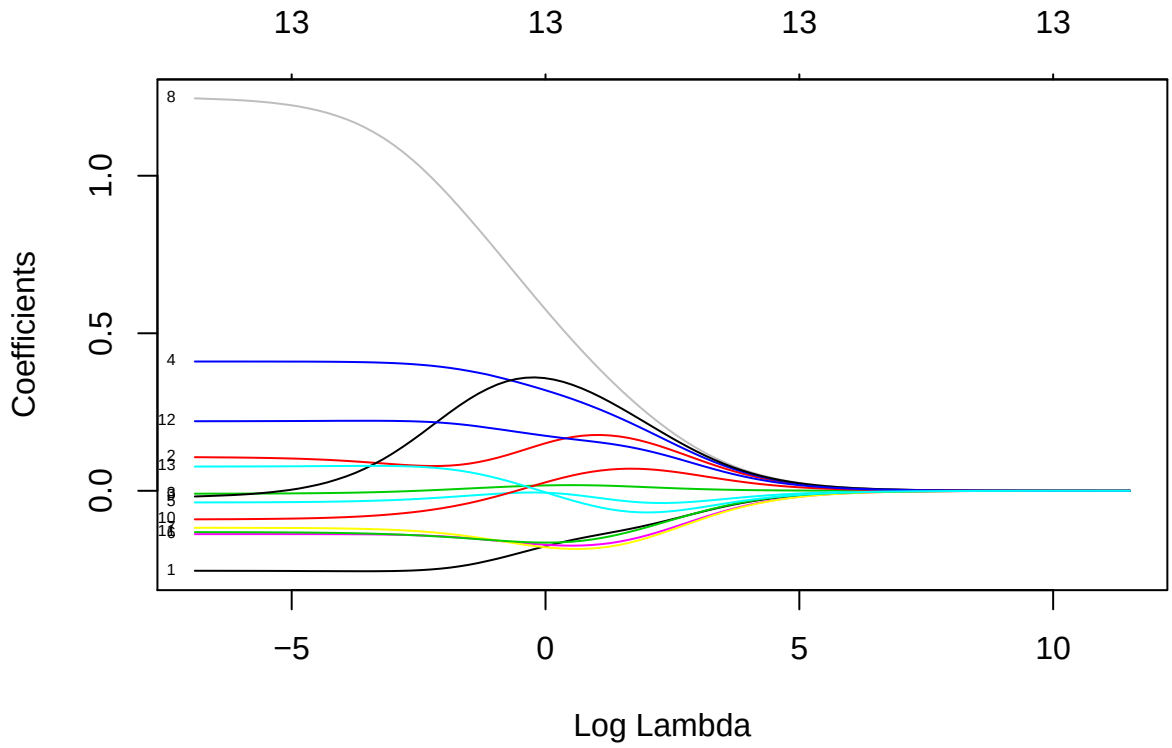

Figure 6: Ridge Fit Coefficents versus Lambdas

5.4.6 Code to Generate Figure 6

```
# grid of values for lambda (10^5 to 10^-3)
grid = 10^seq(5, -3, length=100)

## Fit a ridge regression model for each value of the tuning parameter
# alpha 0 -> Ridge regression not LASSO
ridgeFit = glmnet(X, y, alpha=0, standardize=FALSE, lambda=grid)

plot(ridgeFit, xvar="lambda", col=1:13, label=TRUE)
```

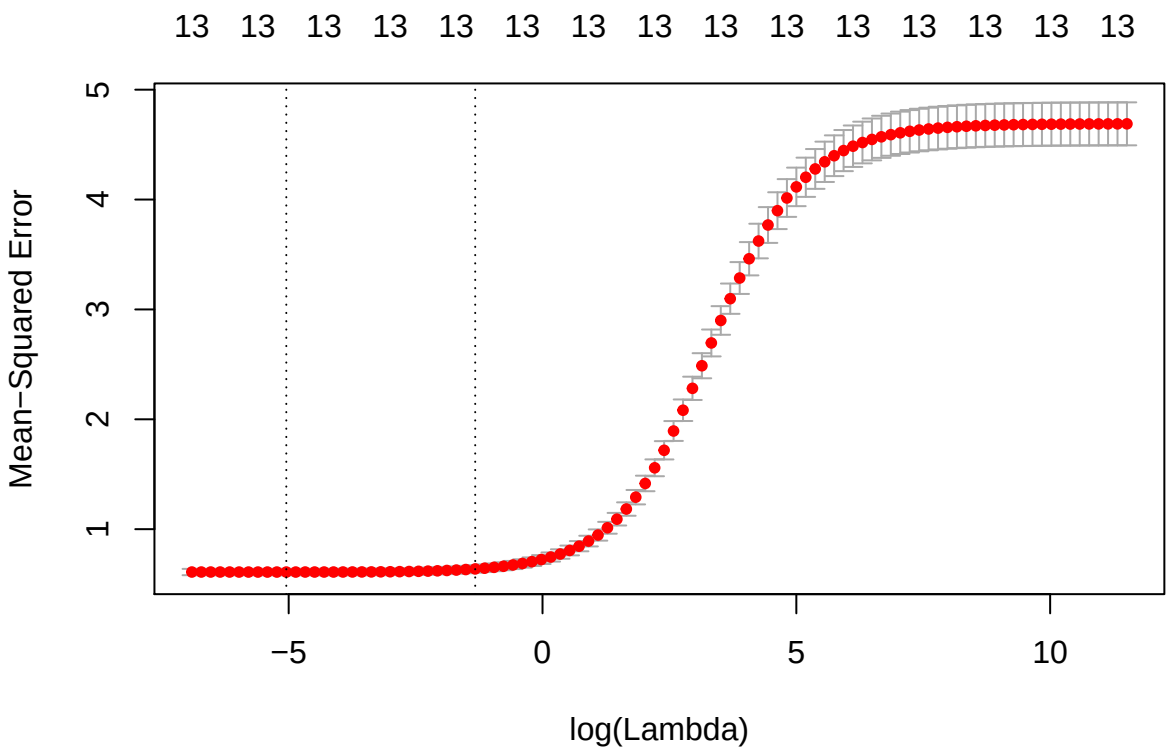

Figure 7: Ridge Fit 10 Fold Cross Validation

5.4.7 Code to Generate Figure 7 with Minimum Result

```
# Fit a ridge regression model using all lambdas with cv validation
# alpha 0 -> Ridge regression not LASSO
ridgeFitCv = cv.glmnet(X, y, alpha=0, standardize=FALSE, lambda=grid)

plot(ridgeFitCv) # plot cross validated error
```

```
# Find exact minimum
(lambdaMin = ridgeFitCv$lambda.min)
```

```
## [1] 0.006428073
```

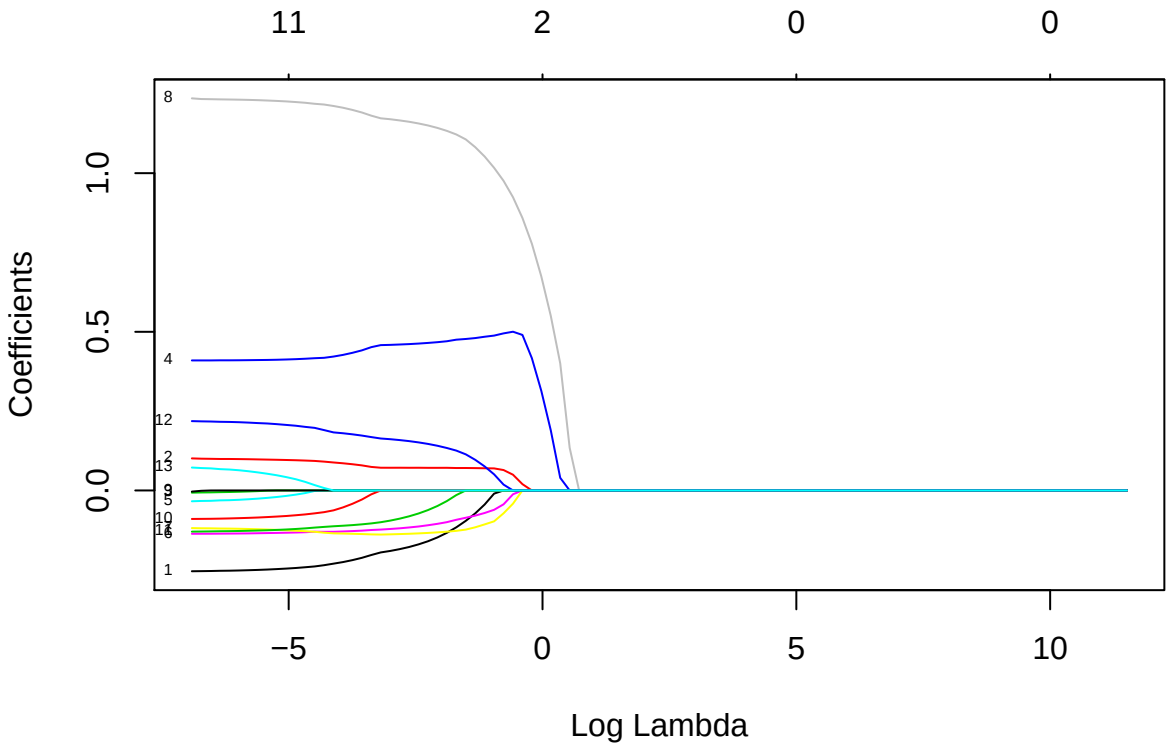

Figure 8: LASSO Coefficents versus Lambdas

5.4.8 Code to Generate Figure 8

```
# Fit a ridge regression model for each value of the tuning parameter
# alpha 1 -> LASSO
lasso = glmnet(X, y, alpha=1, standardize=FALSE, lambda=grid)

plot(lasso, xvar="lambda", col=1:13, label=TRUE)
```

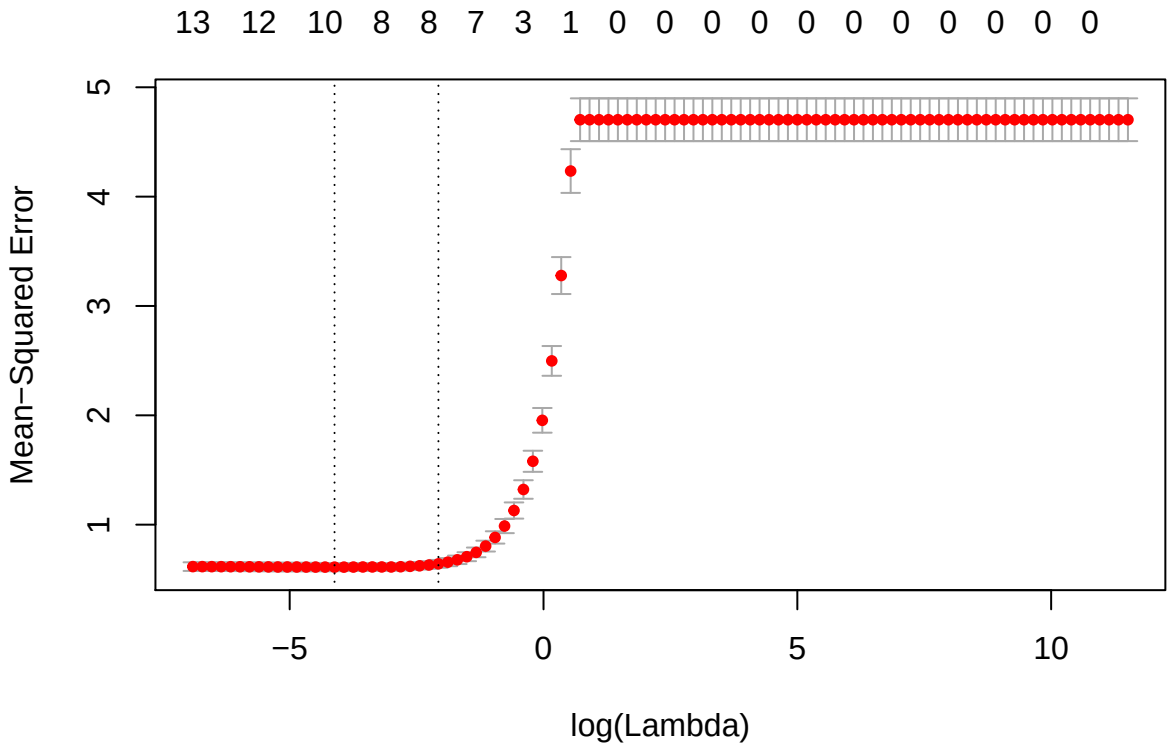

Figure 9: LASSO 10 Fold Cross Validation

5.4.9 Code to Generate Figure 9 with Minimum Result

```
# Fit a ridge regression model using all lambdas with cv validation
# aplha 1 -> LASSO
lassoCv = cv.glmnet(X, y, alpha=1, standardize=FALSE, lambda=grid)

plot(lassoCv) # plot cross validated error
```

```
# Find exact minimum
(lambdaMin = lassoCv$lambda.min)
```

```
## [1] 0.01629751
```

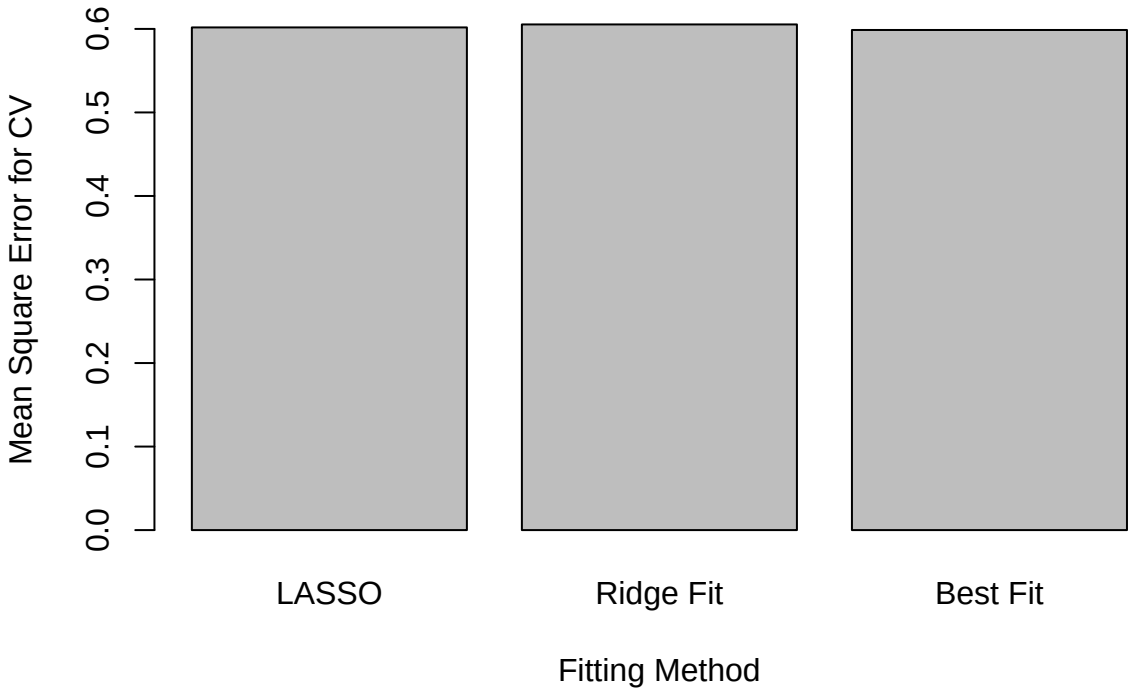

Figure 10: MSE for Different Fitting Methods

5.4.10 Code to Generate Figure 10 with Minimum Results

```
# Pass fold index used by Best Fit Cross Validation to ensure similar folds
lassoCv = cv.glmnet(X, y, alpha=1, standardize=FALSE, lambda=grid, foldid = foldIndex)
ridgeFitCv = cv.glmnet(X, y, alpha=0, standardize=FALSE, lambda=grid, foldid = foldIndex)

i = which(ridgeFitCv$lambda == ridgeFitCv$lambda.min)
ridgeFitError = ridgeFitCv$cvm[i]

j = which(lassoCv$lambda == lassoCv$lambda.min)
lassoError = lassoCv$cvm[j]

# make bar plot of minimum MSEs for selected predictors and lambdas
errors = c(lassoError, ridgeFitError, bssMse[9])
names(errors) = c("LASSO", "Ridge Fit", "Best Fit")
barplot(errors, xlab = "Fitting Method", ylab = "Mean Square Error for CV")
```

```
# Exact values
lassoError # LASSO MSE
```

```
## [1] 0.6017774
```

```
ridgeFitError # Ridge Fit MSE
```

```
## [1] 0.6053586
```

```
bssMse[9] # Best Fit MSE
```

```
## [1] 0.5986935
```