Untitled

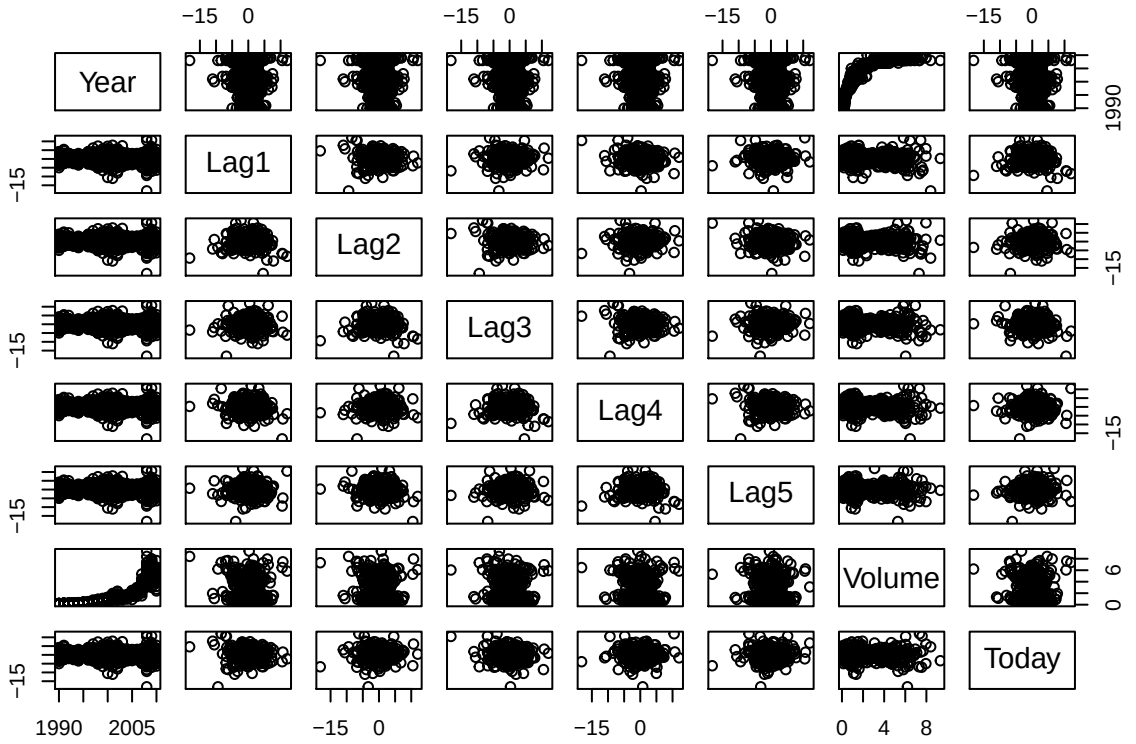
Rohitashwa Chakraborty

30/07/2021

EXERCISE 4.10:

4.10.a

```
##       Year           Lag1                Lag2                Lag3          
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       Lag4                Lag5              Volume            Today          
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540  
##  Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410  
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499  
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050  
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260  
##  Direction 
##  Down:484  
##  Up  :605  
##            
##            
##            
##
```



Positive Correlation between `Year` and `Volume` observed.

4.10.b

```
##
## Call:
## glm(formula = Direction ~ ., family = binomial, data = Weekly[,
##     c(2:7, 9)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

To check if a parameter is significant or not, we must check for its **P-Vaue**.

From the Summary, only `Lag2` has a **P-Value** < 0.05. Thus, only `Lag2` is statistically significant.

4.10.c

```
##           Reference
## Prediction Down  Up
##       Down   54  48
##       Up    430 557

## Accuracy            :  56.11 %

## Recall/Sensitivity  :  92.07 %

## Precision           :  56.43 %

## Specificity         :  11.16 %

## Up Prediction Rate  :  56.43 %

## Down Prediction Rate:  52.94 %
```

48 *"Up"* were mistaken for *"Down"*. 430 *"Down"* were mistaken for *"Up"*. 54 *"Down"*+ 557 *"Up"* were predicted accurately . Model is has higher accuracy when the prediction is *"Up"*

These results were obtained from the same set of observations the model was trained upon. Therefore, it is highly likely that the results would prove to be *overly optimistic* when tested on a new set of data.

4.10.d

```
##           Reference
## Prediction Down Up
##       Down    9  5
##       Up     34 56

## [Logistic Regression] Overall Fraction of Correct Predictions (Accuracy) :  0.62
```

4.10.g

```
##           Reference
## Prediction Down Up
##       Down   21 30
##       Up     22 31

## [KNN (k = 1)] Overall Fraction of Correct Predictions (Accuracy) :  0.5
```

4.10.h

Considering **only Accuracy** as our metric, we can conclude that *Logistic Regression* outperforms *KNN (with k = 1)*

4.10.i

Experimenting with different KNN models:

```
## Predictors: Lag2

## [KNN (k =  30 )] Accuracy :  0.53
## [KNN (k =  130 )] Accuracy :  0.57
## [KNN (k =  230 )] Accuracy :  0.59
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: Lag2, Lag1

## [KNN (k =  30 )] Accuracy :  0.54
## [KNN (k =  130 )] Accuracy :  0.57
## [KNN (k =  230 )] Accuracy :  0.59
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: Lag2^2

## [KNN (k =  30 )] Accuracy :  0.62
## [KNN (k =  130 )] Accuracy :  0.62
## [KNN (k =  230 )] Accuracy :  0.59
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: Lag2*Lag1

## [KNN (k =  30 )] Accuracy :  0.55
## [KNN (k =  130 )] Accuracy :  0.57
## [KNN (k =  230 )] Accuracy :  0.57
## [KNN (k =  330 )] Accuracy :  0.59

##
## Predictors: All

## [KNN (k =  30 )] Accuracy :  0.89
## [KNN (k =  130 )] Accuracy :  0.86
## [KNN (k =  230 )] Accuracy :  0.79
## [KNN (k =  330 )] Accuracy :  0.75
```

Considering only **Accuracy**, we can conclude that the following models perform the best:

K= 30, Predictors: All Predictors

```
preds <- knn(as.matrix(train[,-9]),
               as.matrix(test[,-9]),
               train$Direction, k=30)
cm <- confusionMatrix(preds, Direction[!filtered_years], positive = "Up")
cat("Confusion Matrix for K= 130, Predictors: All\n")
cm$table
```

Experimenting with different Logistic Regression Models:

```
## Logistic Regression

##
## [Predictors: Lag2 ] Accuracy :  0.62
## [Predictors: Lag2+Lag1 ] Accuracy :  0.58
## [Predictors: Lag2*Lag1 ] Accuracy :  0.58
## [Predictors: I(Lag2^2) ] Accuracy :  0.59
## [Predictors: All] Accuracy :  1
```

Considering Accuracy, It seems Using **All the Parameters** gives by far the most accurate model with a **100% Accuracy**.

```
## Confusion Matrix for Linear Regression Model with All Predictors:

##           Reference
## Prediction Down Up
##       Down   43  0
##       Up      0 61
```

*NOTE: This is not surprising because one of the predictors the model trains upon is **Today**. This predictor seems to have a distinct linear boundary when plotted against **Direction***

EXERCISE 6.9:

6.9.a

Creating a **80-20** split between *Train* and *Test* set

```
## Length of College Dataset: 777

## Length of Train Dataset  : 622

## Length of Test Dataset   : 155
```

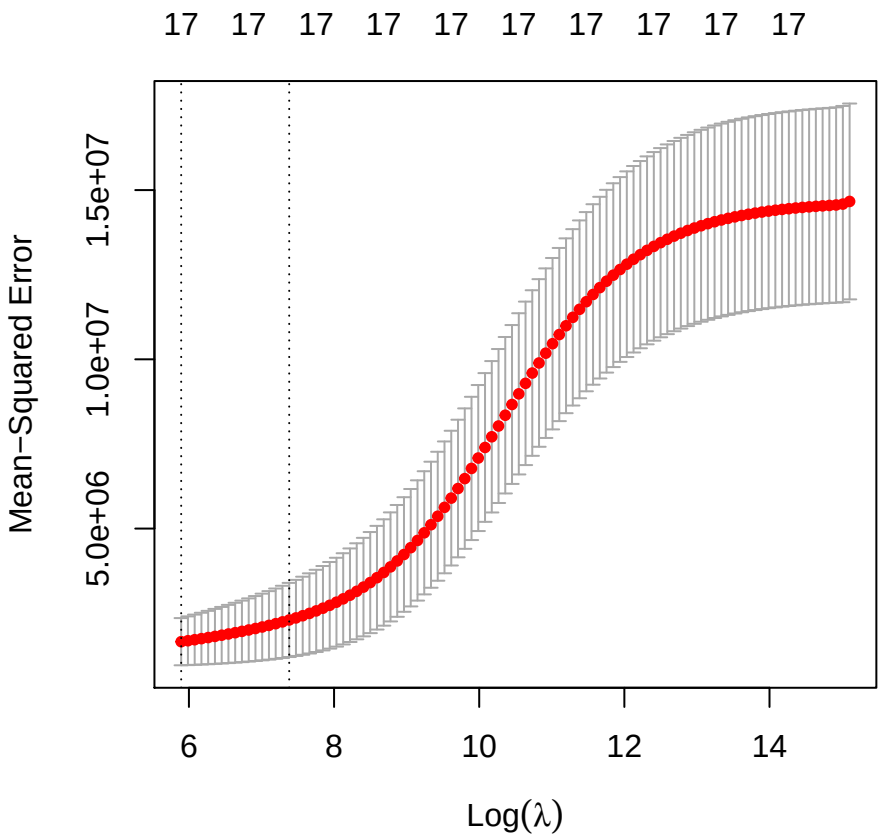
6.9.b

The test error on applying Linear Regression on a Model with All Parameters is:

```
## 1578073.167
```

6.9.c

```
## Optimal Lambda, by 10-fold cross-validation is: 362.660783476255
```

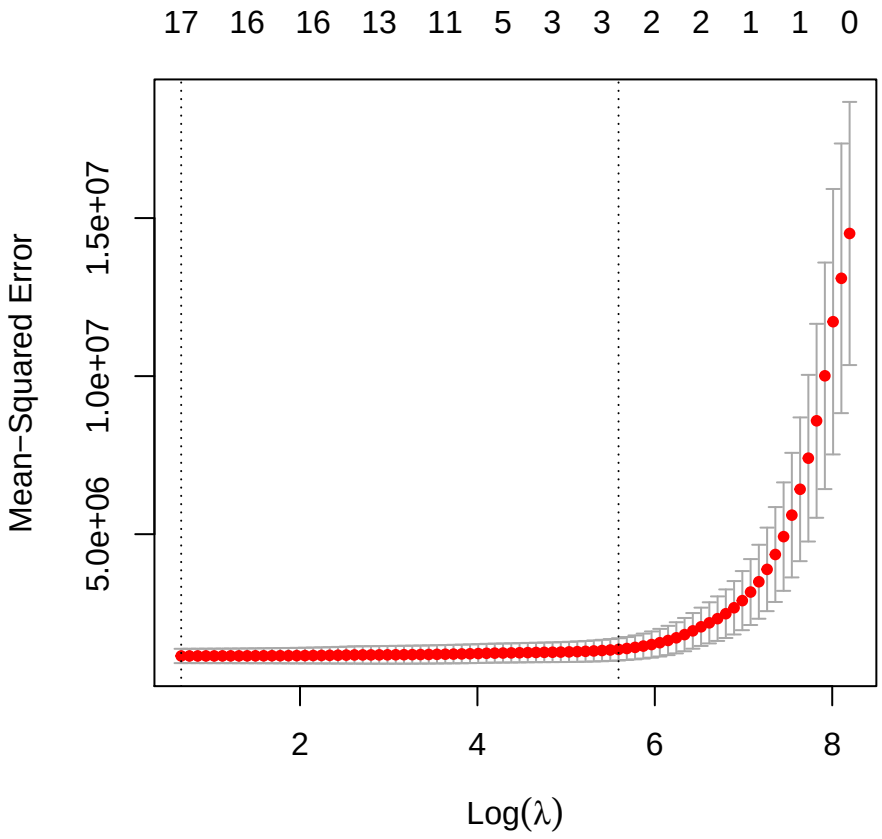


The test error on applying **Ridge-Regression** on a Model with All Parameters is:

```
## 1447148.005
```

6.9.d

```
## Optimal Lambda, by 10-fold cross-validation is: 1.93541152134794
```



The test error on applying **Lasso-Regression** on a Model with All Parameters is:

```
## 1565219.591
```

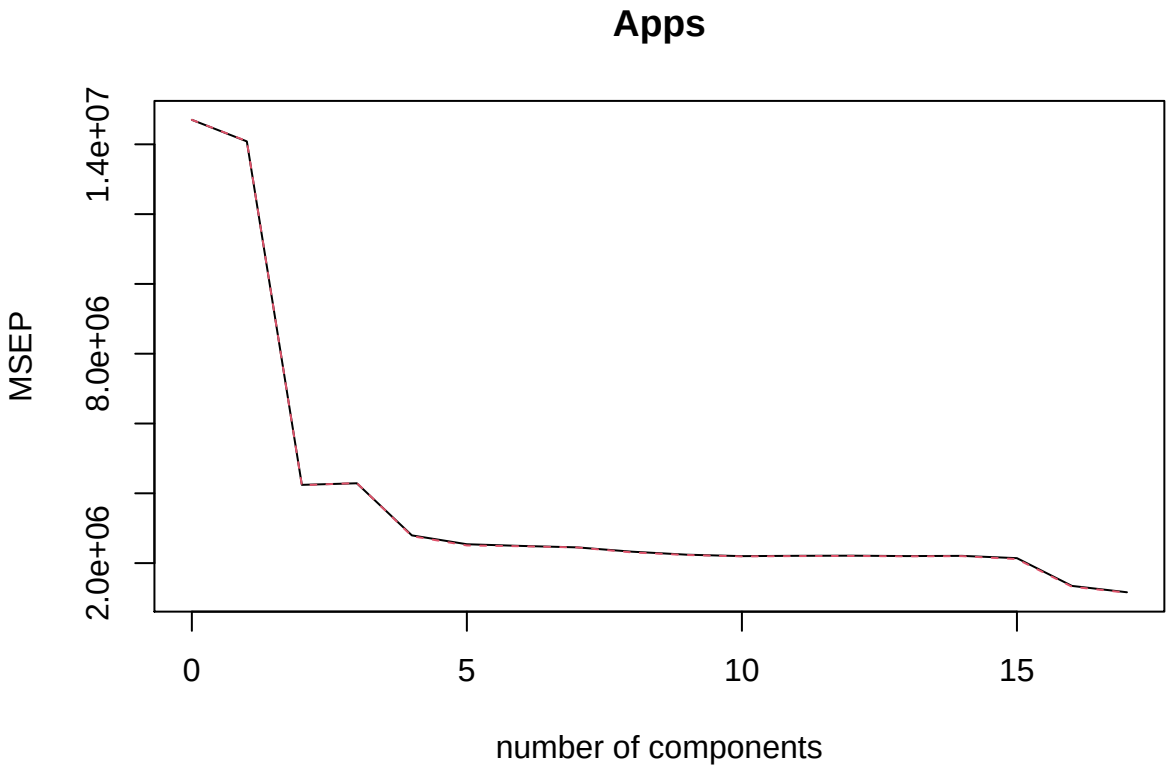
Coefficients of Predictors using the Lasso-Regression method are:

```
##   PrivateYes       Accept       Enroll    Top10perc    Top25perc  F.Undergrad
##  -381.594455  4108.880244 -1025.359690   864.928922  -247.672551   333.295597
##  P.Undergrad     Outstate   Room.Board        Books     Personal          PhD
##    98.397947  -293.010691   153.190780    32.808299    10.803342  -157.271619
##     Terminal    S.F.Ratio  perc.alumni       Expend    Grad.Rate
##    -3.363536    68.276135     7.318633   297.806071    96.507224
```

Thus, Non-Zero Coefficient Estimate Predictors are:

```
##  [1] "PrivateYes"  "Accept"      "Enroll"      "Top10perc"   "Top25perc"
##  [6] "F.Undergrad" "P.Undergrad" "Outstate"    "Room.Board"  "Books"
## [11] "Personal"    "PhD"         "Terminal"    "S.F.Ratio"   "perc.alumni"
## [16] "Expend"      "Grad.Rate"
```

6.9.e




```
## Data:    X dimension: 622 17
##  Y dimension: 622 1
## Fit method: svdpc
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3834     3753     2060     2071     1672     1594     1579
## adjCV         3834     3754     2057     2072     1665     1583     1576
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1566     1526     1497      1483      1486      1488      1484
## adjCV     1565     1519     1494      1481      1484      1485      1481
##        14 comps  15 comps  16 comps  17 comps
## CV         1486      1464      1160      1079
## adjCV      1483      1454      1150      1073
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X      32.019    57.05    64.13    70.01    75.36    80.38    84.09    87.44
## Apps    4.315    72.01    72.02    81.89    83.65    83.73    83.98    85.12
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       90.48     92.84     94.92     96.78     97.86     98.72     99.36
## Apps    85.40     85.75     85.75     85.76     85.88     85.94     89.94
```

```
##       16 comps  17 comps
## X        99.83    100.00
## Apps     92.88     93.47
```

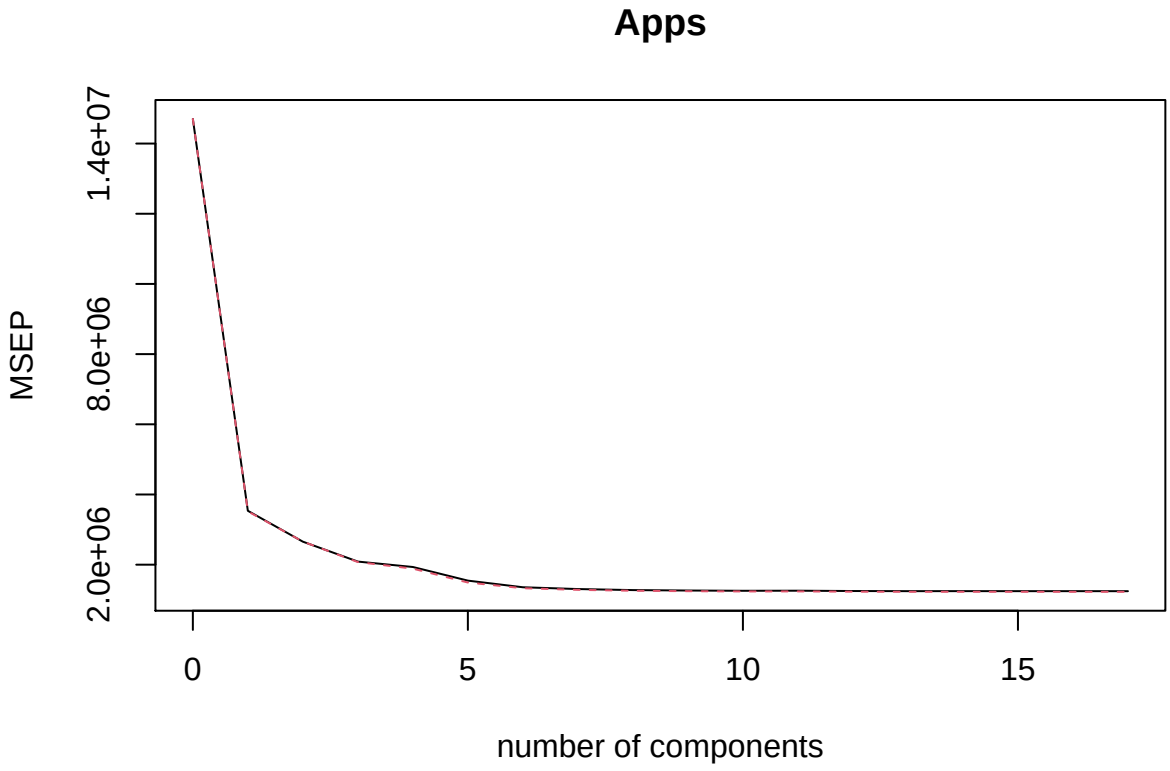
```
## Minimum CV at
```

Minimum CV at $\mathbf{M = 17}$. Thus, using *predict(. . . ,ncomp=17,. . .)*

The test error on applying **Principal Component Regression** on a Model with All Parameters is:

```
## 1578073.167
```

6.9.f




```
## Data:    X dimension: 622 17
##  Y dimension: 622 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3834     1880     1630     1445     1391     1243     1165
## adjCV         3834     1876     1630     1440     1374     1221     1153
```

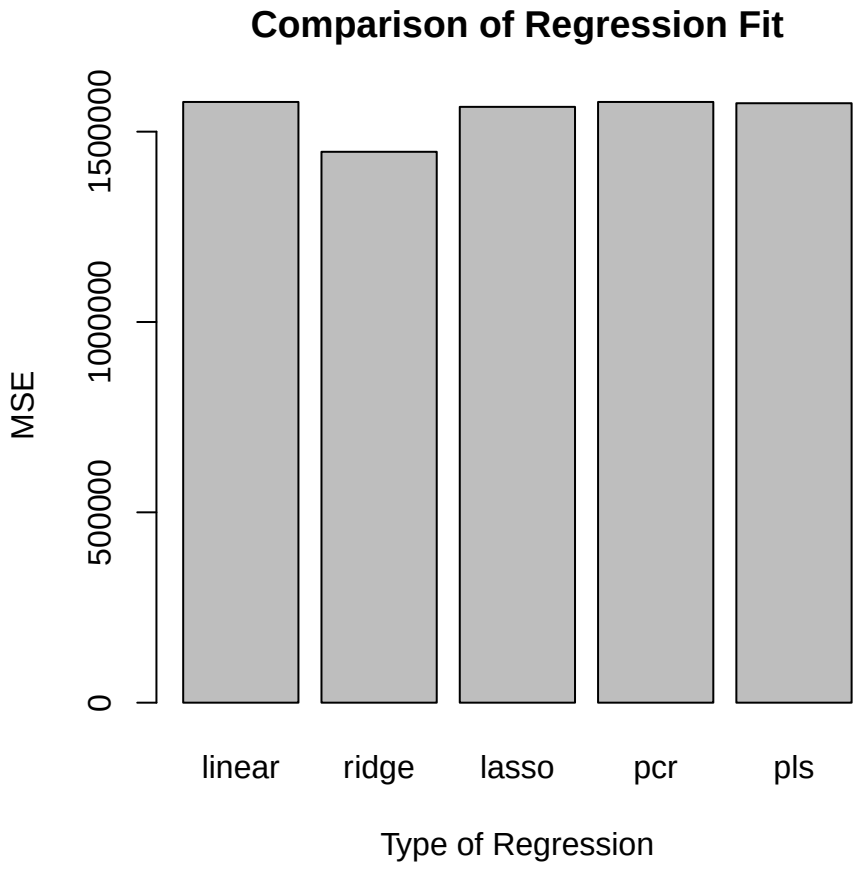
```
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1143     1130     1124      1121      1122      1118      1116
## adjCV     1133     1121     1116      1112      1113      1109      1108
##        14 comps  15 comps  16 comps  17 comps
## CV         1116      1116      1116      1116
## adjCV      1108      1108      1108      1108
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X       25.52    45.30    62.57    65.06    67.50    72.05    76.04    80.49
## Apps    77.30    83.58    87.50    90.88    92.89    93.15    93.24    93.31
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       82.50     85.41     87.76     91.08     92.72     95.12     96.97
## Apps    93.39     93.42     93.45     93.46     93.46     93.47     93.47
##       16 comps  17 comps
## X        97.98    100.00
## Apps     93.47     93.47
```

Minimum CV at $\mathbf{M = 13}$. Thus, using *predict(. . . ,ncomp=13,. . .)*

The test error on applying **Partial Least Squares Regression** on a Model with All Parameters is:

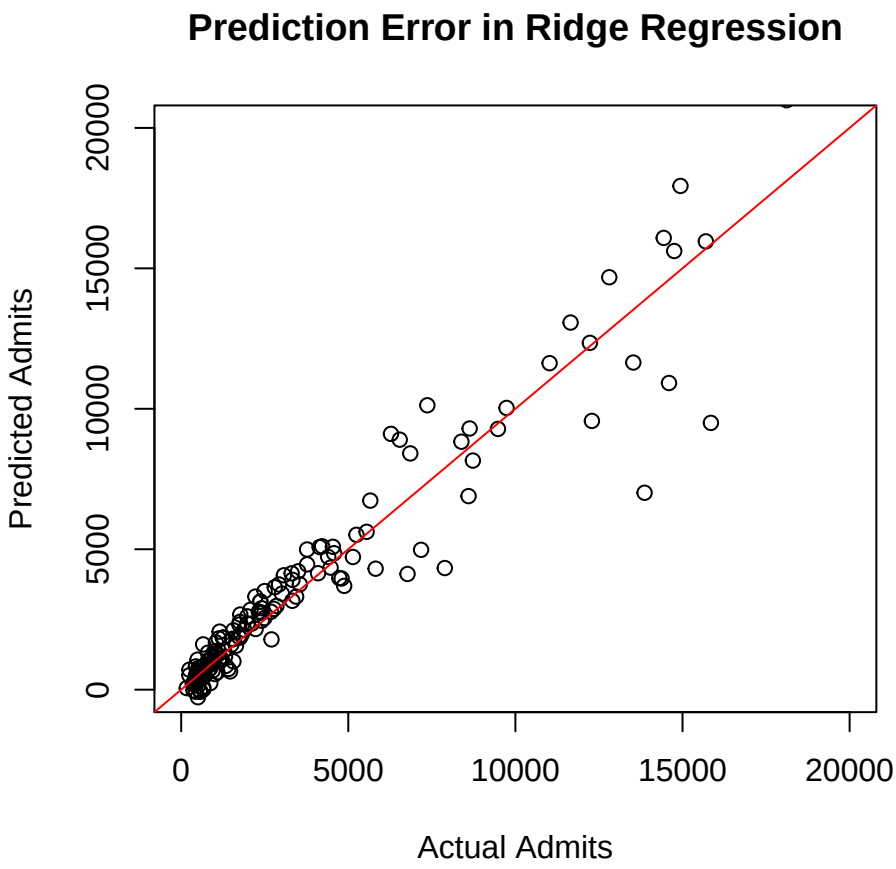
```
## 1574745.803
```

6.9.g



Most of the regression methods *("Linear", "Ridge", "Lasso", "PCR", "PLS")* have approximately the same amount of error.

The **Ridge Regression** outperforms others by a slight margin. Its **Test MSE** is: 1447148.005 *(standard deviation = 5319499)*



EXERCISE 6.11:

6.11.a

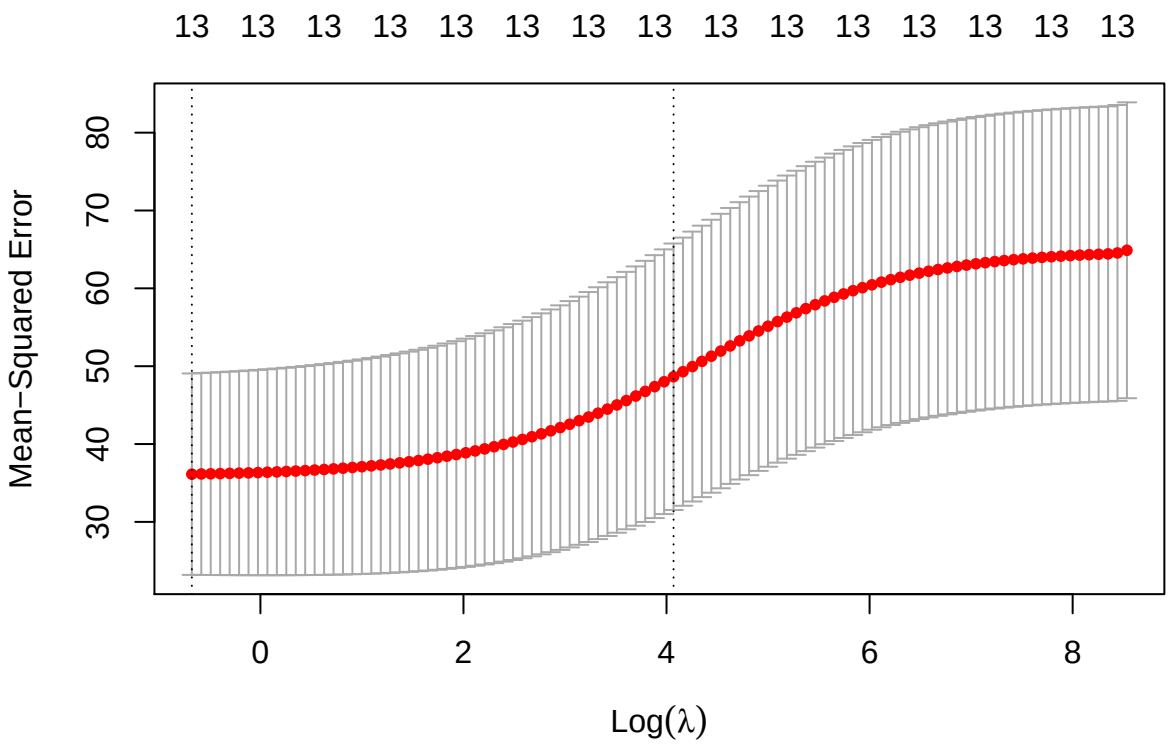
Creating a **80-20** split between *Train* and *Test* set

```
## Length of Boston Dataset: 506

## Length of Train Dataset : 405

## Length of Test Dataset  : 101
```

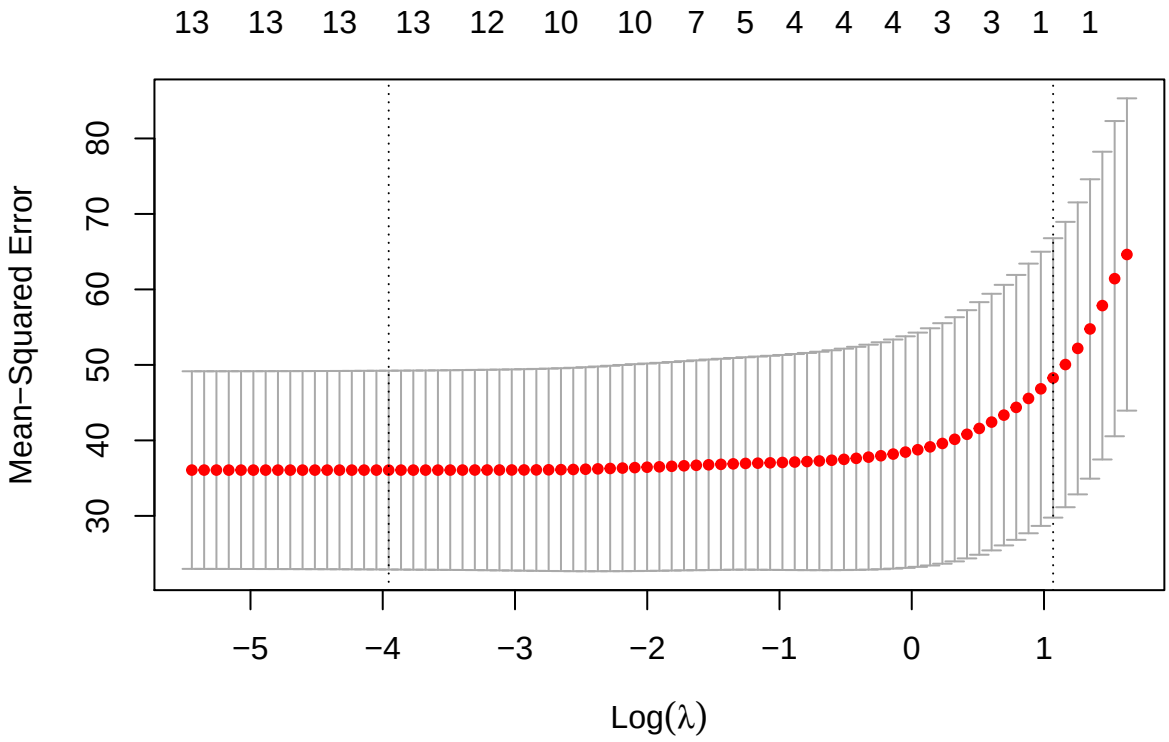
Ridge Regression




```
## Optimal Lambda, by 10-fold cross-validation is: 0.51
```

```
##
## Test Error of Ridge Regression: 71.33
```

Lasso Regression



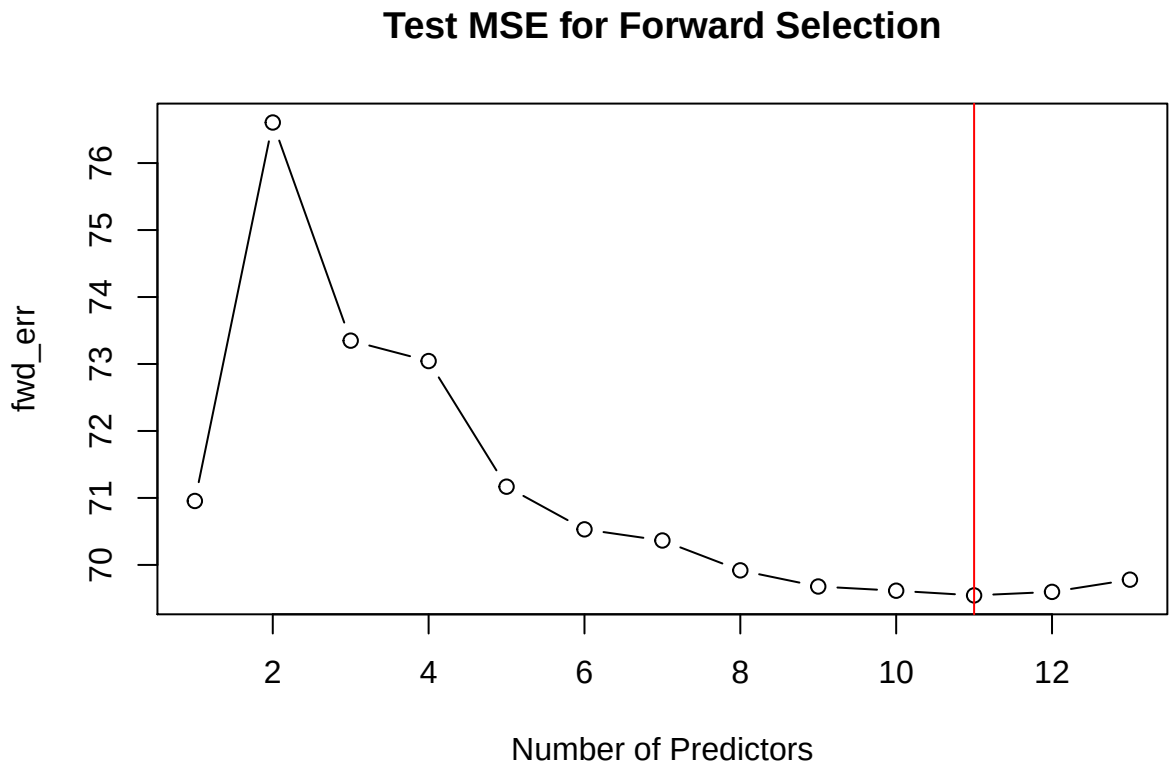

```
## Optimal Lambda, by 10-fold cross-validation is: 0.02
```

```
##
## Test Error of Lasso Regression: 69.95
```

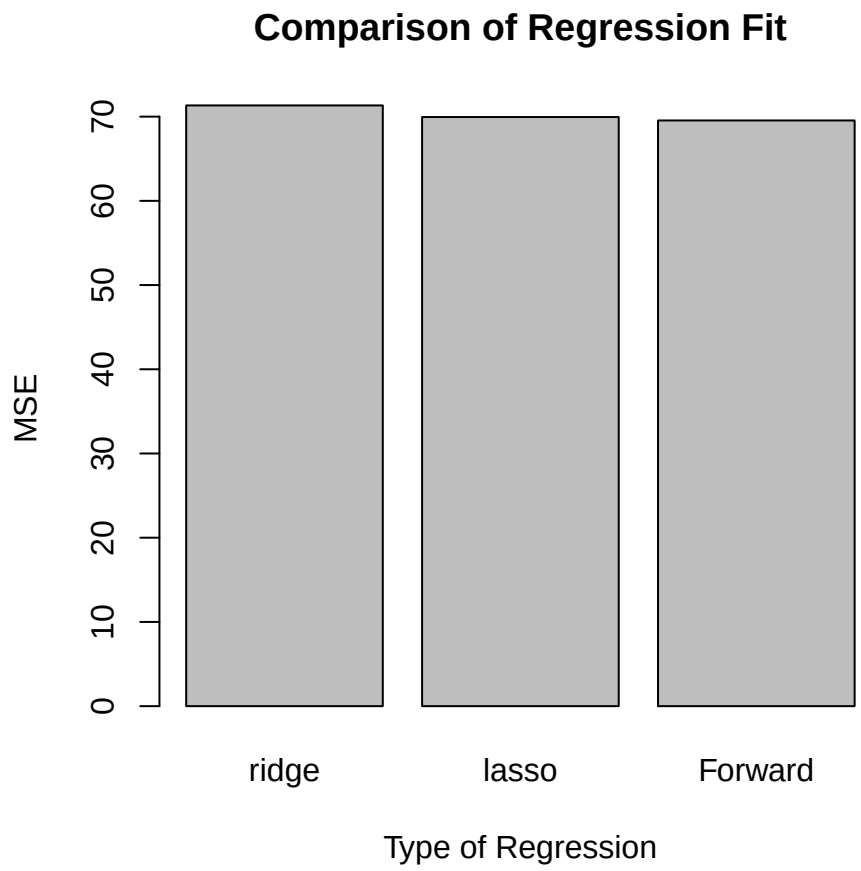
Subset Selection ***(Forward Selection)***

```
## Subset selection object
## Call: regsubsets.formula(crim ~ ., data = train, nvmax = ncol(Boston) -
##     1)
## 13 Variables  (and intercept)
##         Forced in Forced out
## zn          FALSE      FALSE
## indus       FALSE      FALSE
## chas        FALSE      FALSE
## nox         FALSE      FALSE
## rm          FALSE      FALSE
## age         FALSE      FALSE
## dis         FALSE      FALSE
## rad         FALSE      FALSE
## tax         FALSE      FALSE
## ptratio     FALSE      FALSE
## black       FALSE      FALSE
## lstat       FALSE      FALSE
## medv        FALSE      FALSE
```

```
## 1 subsets of each size up to 13
## Selection Algorithm: exhaustive
##           zn  indus chas nox rm  age dis 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 ) "*" "*"   "*"  "*" "*" "*" "*" "*" "*" "*"     "*"   "*"   "*"
```




```
##
## Min. Test Error for Forward Selection is: 69.55
```



The regression methods *("Ridge", "Lasso", "Forward Selection")* have approximately the same amount of error.

The **Subset Selection(Forward Selection) Regression** outperforms others by a slight margin. Its **Test MSE** is: 69.55

6.11.b

From the above Test Set Errors, we can reasonably conclude that the 11-parameter Forward selection model is the best fit to the Boston Dataset

6.11.c

No because not all the predictors add much value to the model. Adding more predictors makes the model more complex and computationally expensive. Thus, If a predictor does not increase the amount of variance explained by the model significantly, we can drop it. In our case, We choose the Forward Selection model, which uses just 11 Predictors.

EXERCISE 8.8:

8.8.a
