Take Home Examination (Intro To ML)

Rohitashwa Chakraborty (rc47878)

01 Aug 2021

Exercise 2.10:

2.10.a

```
## Number of rows in the Boston DataSet   : 506
```

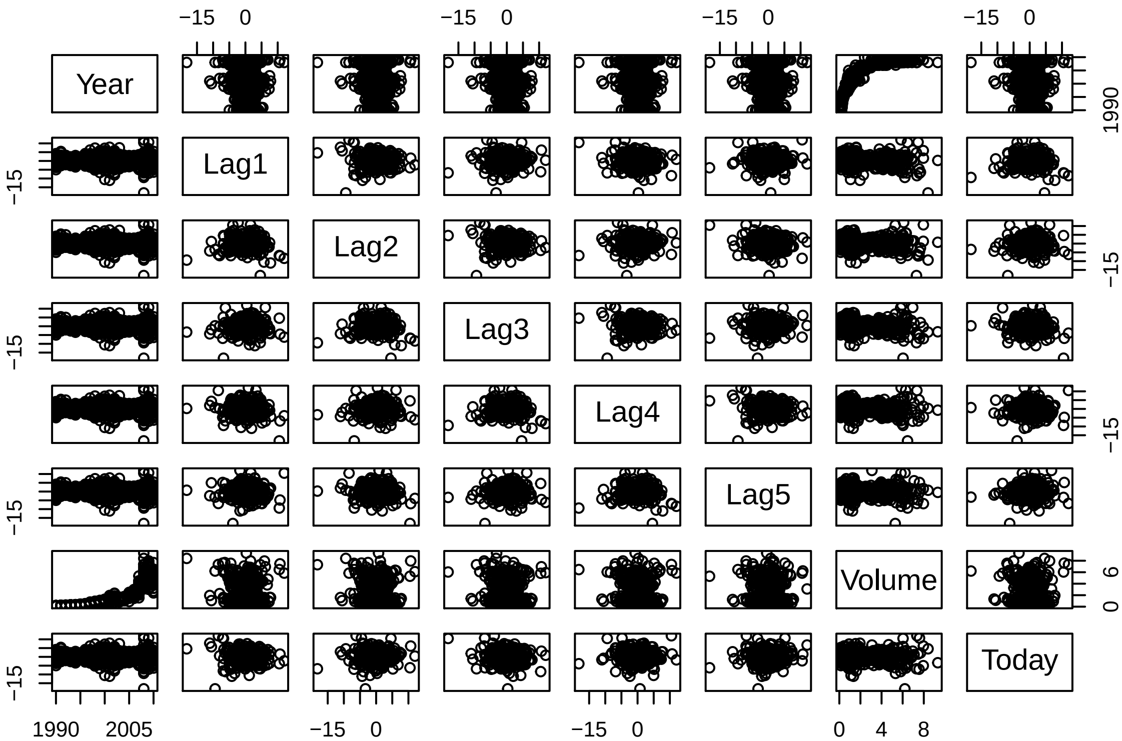
```
## Number of columns in the Boston DataSet: 14
```

The columns represent the following:

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
```

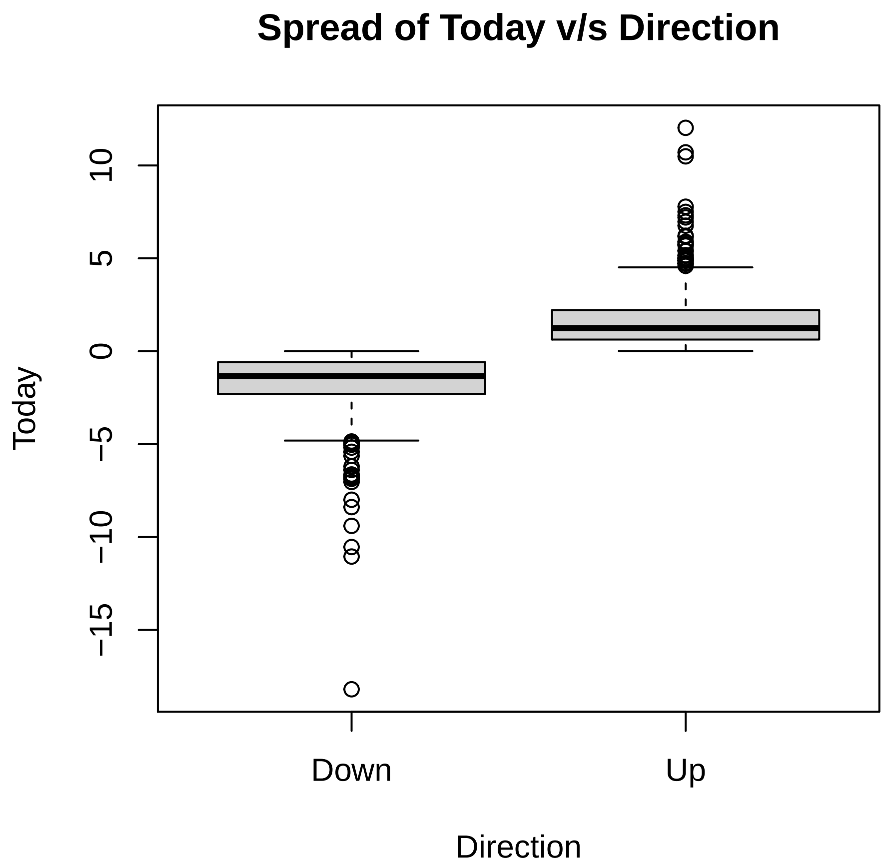
```
##
## [Predictors: Lag2 ] Accuracy :  0.62
## [Predictors: Lag2+Lag1 ] Accuracy :  0.58
## [Predictors: Lag2*Lag1 ] Accuracy :  0.58
## [Predictors: I(Lag2^2) ] Accuracy :  0.59
## [Predictors: All] Accuracy :  1
```

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
```

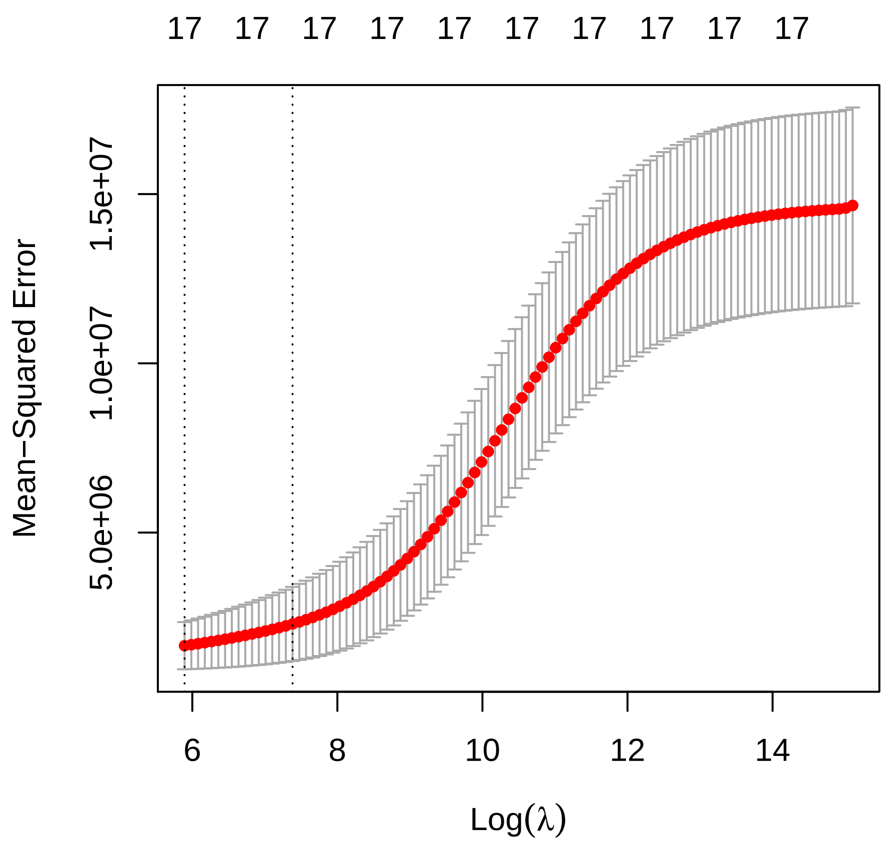
6.9.b

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
```

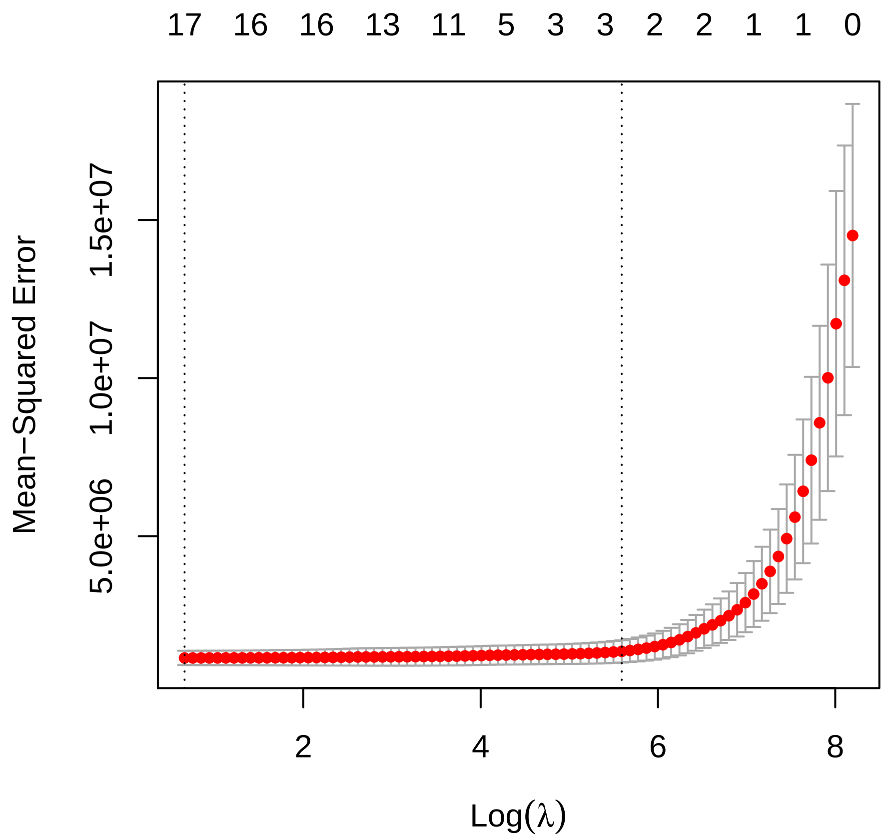


The test error on applying **Ridge-Regression** on a Model with All Parameters is:

```
## 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
```

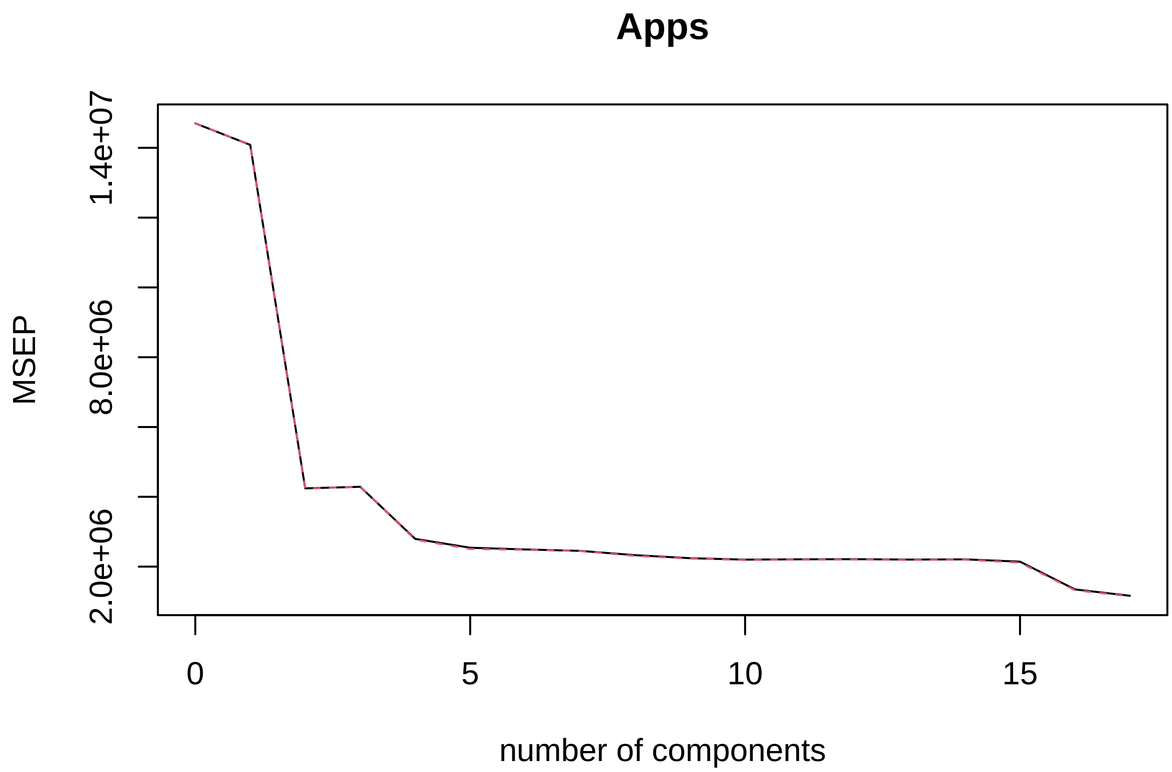
Coefficients of Predictors using the Lasso-Regression method are:

```
##   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
```

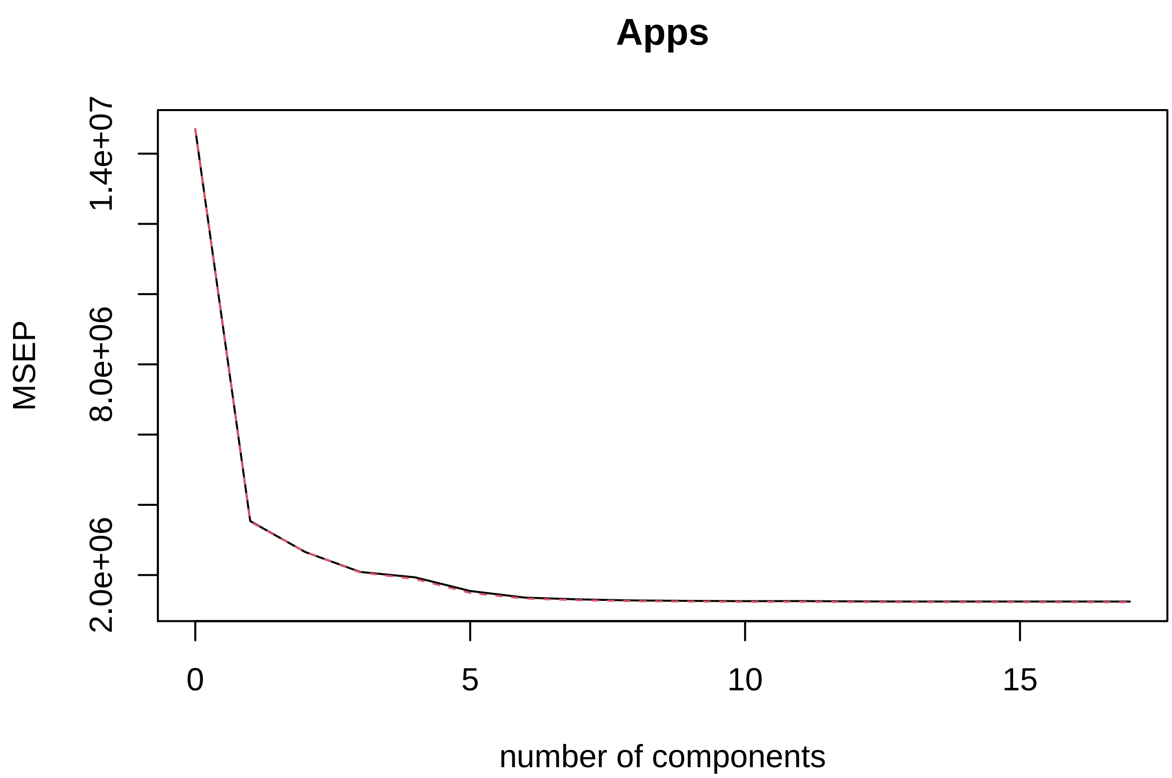
```
## Minimum CV at
```

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
```

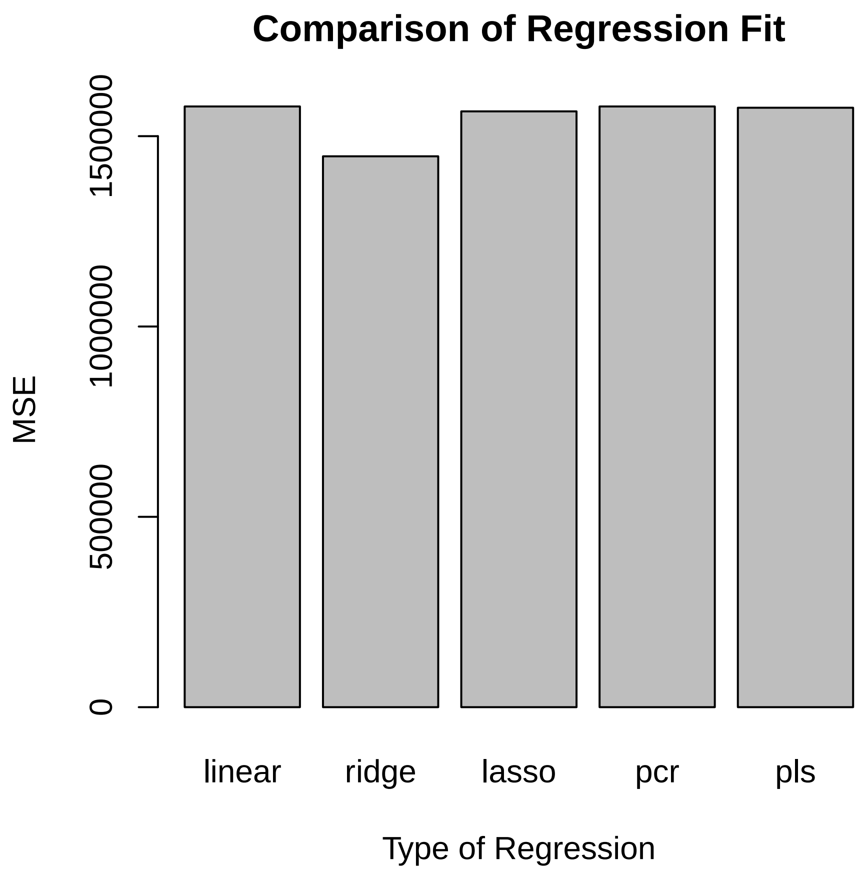
```
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1143     1130     1124      1121      1122      1118      1116
## adjCV     1133     1121     1116      1112      1113      1109      1108
##        14 comps  15 comps  16 comps  17 comps
## CV         1116      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:

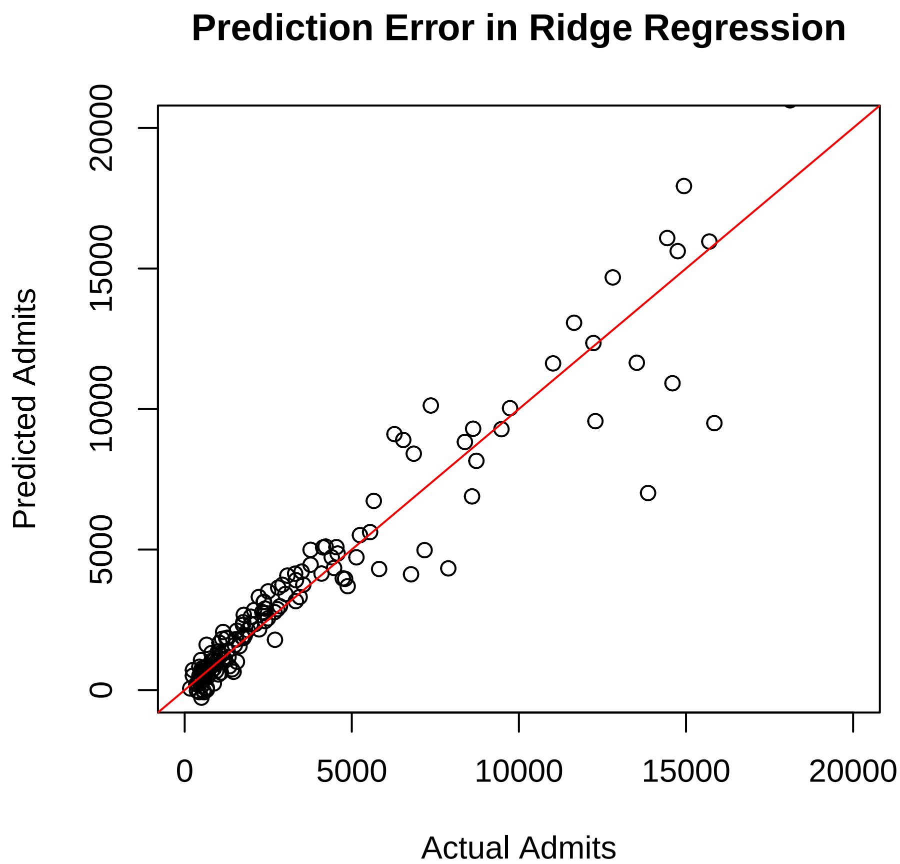
```
## 1574745.803
```

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

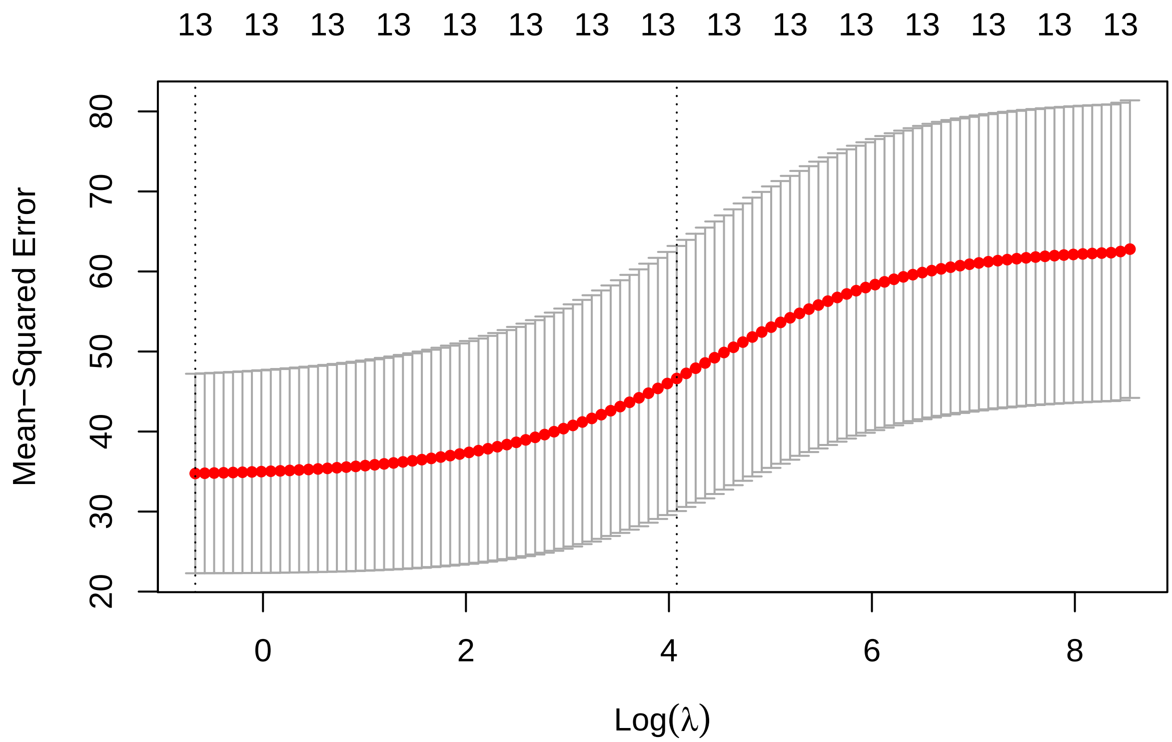
Creating a **80-20** split between *Train* and *Test* set

```
## Length of Boston Dataset: 506
```

```
## Length of Train Dataset : 405
```

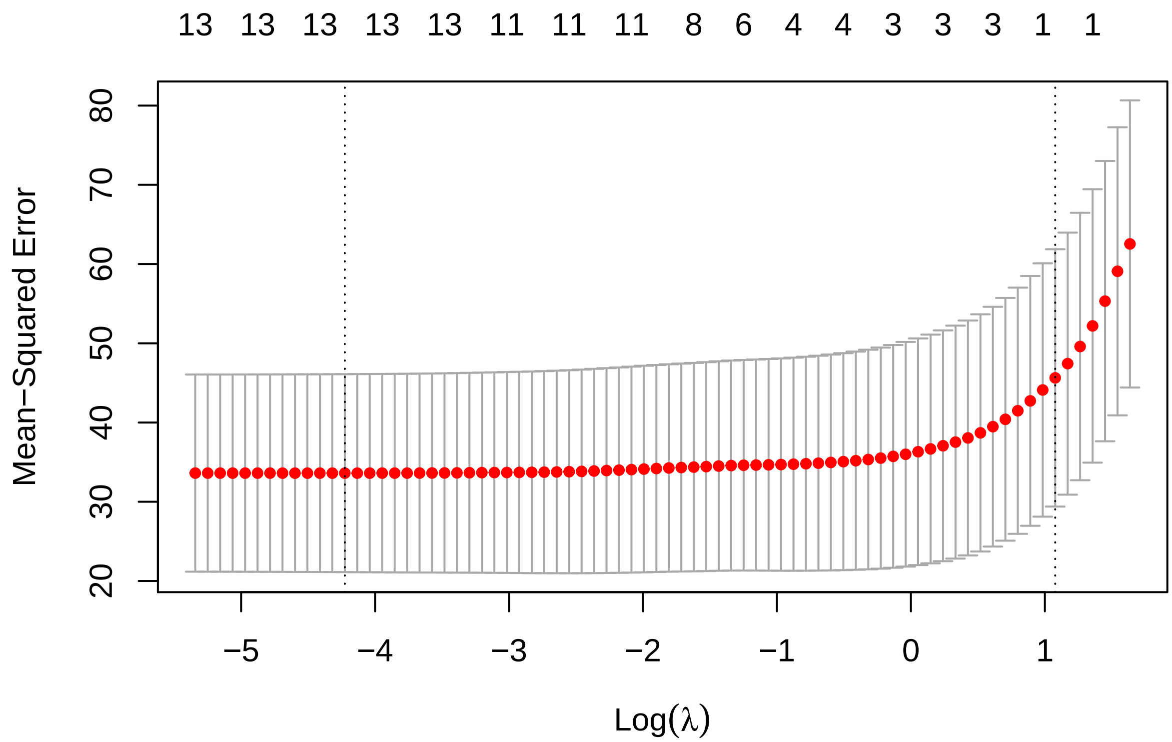
```
## Length of Test Dataset  : 101
```

Ridge Regression




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

##
## Test Error of Ridge Regression: 79.45
```

Lasso Regression

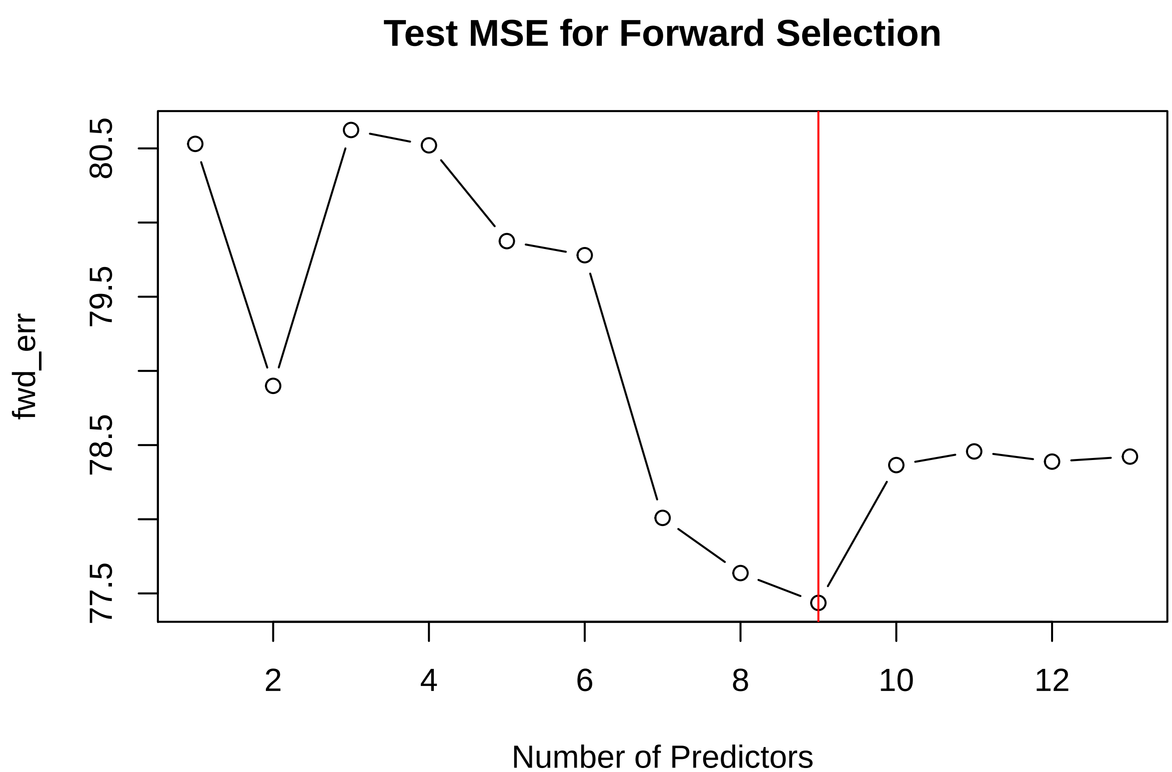



```
## Optimal Lambda, by 10-fold cross-validation is: 0.01

##
## Test Error of Lasso Regression: 78.45
```

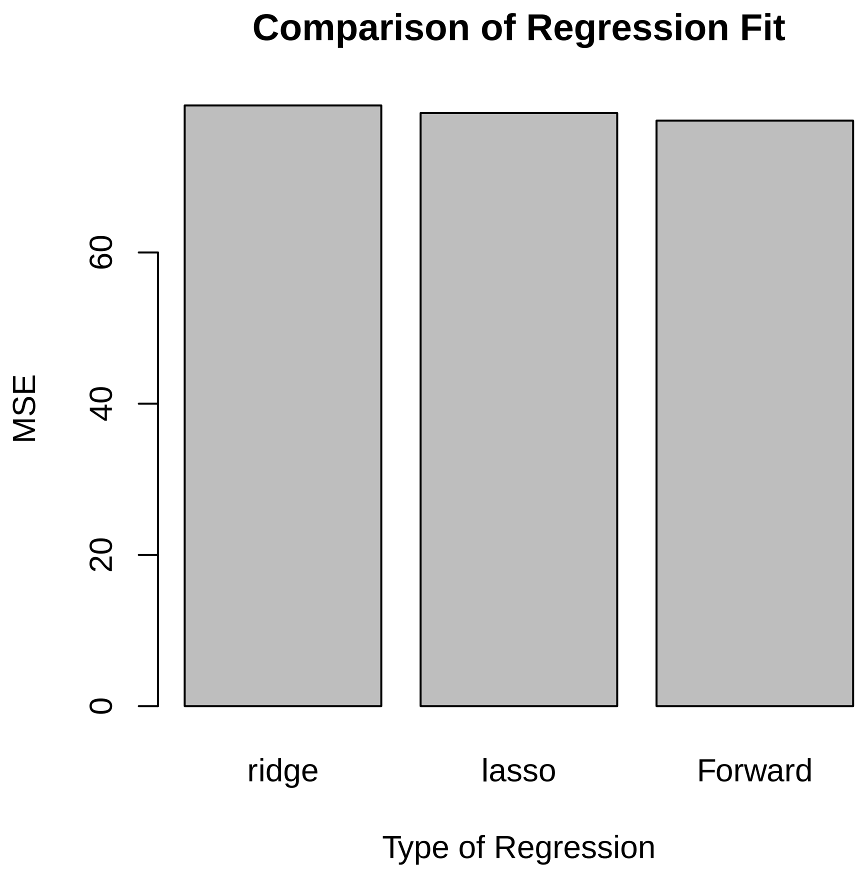
Subset Selection *(Forward Selection)*

```
## 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 ) "*" "*"   "*"  "*" "*" "*" "*" "*" "*" "*"     "*"   "*"   "*"
```

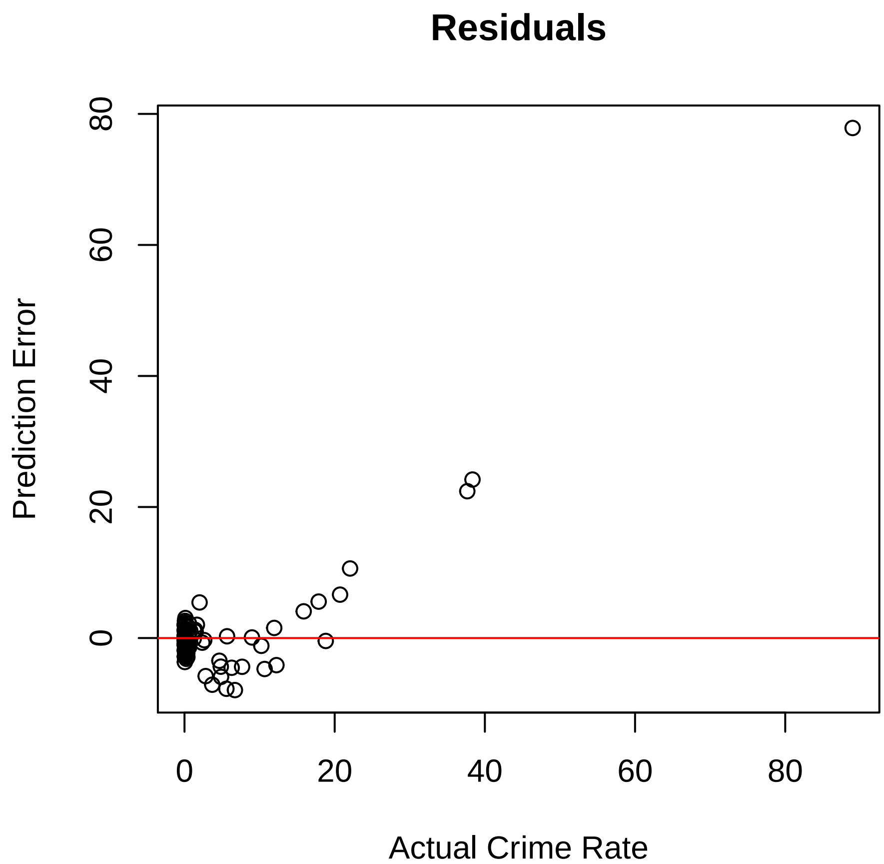



```
##
## Min. Test Error for Forward Selection is: 77.44
```



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

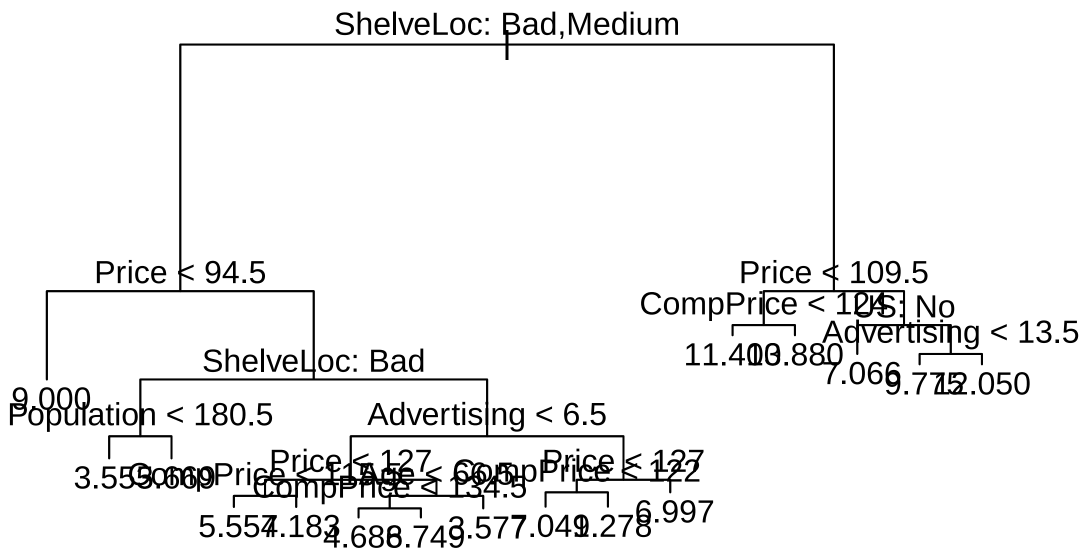
```
## Length of Carseats Dataset: 400
```

```
## Length of Train Dataset : 280
```

```
## Length of Test Dataset  : 120
```

8.8.b

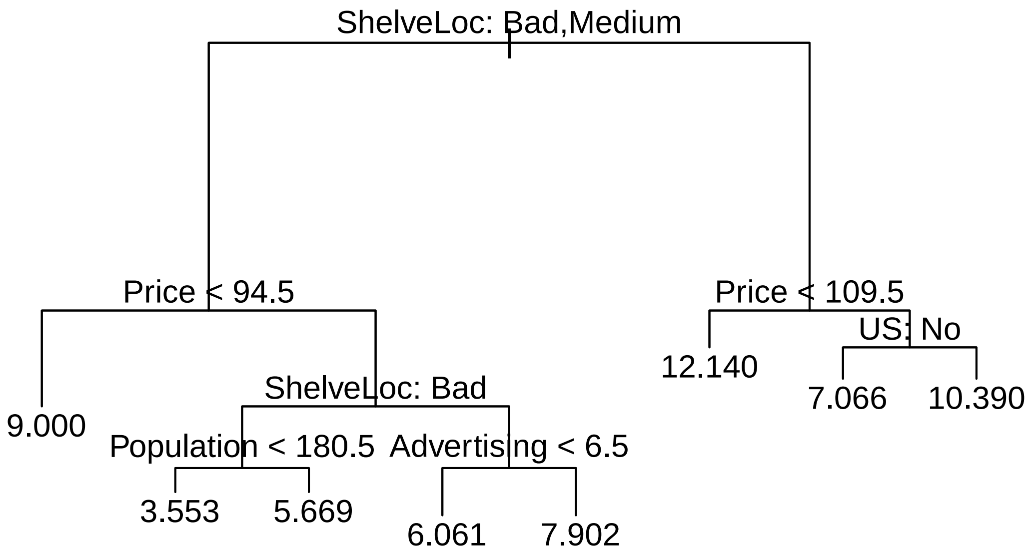
```
##
## Regression tree:
## tree(formula = Sales ~ ., data = carseats_train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Population"  "Advertising" "CompPrice"
## [6] "Age"         "US"
## Number of terminal nodes:  16
## Residual mean deviance:  2.575 = 679.8 / 264
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -3.5770 -1.0600 -0.2128  0.0000  1.0250  5.1330
```




```
## Test MSE for tree is: 4.27687725775323
```

8.8.c

Therefore, Cross-Validation suggests minimum Deviation at tree size of 8. Thus, Pruning the tree to size of 8




```
## Test MSE for Pruned tree is: 5.08898603003723
```

We observe that Test MSE has **increased** to 5.09 after pruning.

8.8.d

For Bagging, Implementing Random Forest with each tree considering all parameters *(mtry = ncol(Carseats) -1)*

NOTE*: Subtracting 1 from ncol(Carseats) because one of the columns in the DataSet is the target column itself*

```
## Test MSE for Bagged Model is: 2.44628424130643
```

```
##                %IncMSE IncNodePurity
## ShelveLoc   78.6084580    748.557677
## Price       62.4450684    591.976064
## CompPrice   32.5703925    217.430548
## Advertising 28.2289858    210.788050
## Age         16.6401794    166.092738
## Income       5.2444968     95.395989
## Population  -0.1787145     82.961943
## Education    3.3273281     63.948412
## US           6.7573125     15.904009
## Urban       -2.5443548      8.609084
```

From the above table, we see that **ShelveLoc** and **Price** are the most important predictors for *Sales*.

8.8.e

For Random Forest, each tree will consider a subset of parameters

$$m = \sqrt(p)$$

Thus: *(mtry = floor(sqrt(ncol(Carseats) -1)))*

NOTE*: Subtracting 1 from ncol(Carseats) because one of the columns in the DataSet is the target column itself*

```
## Test MSE for Random Forest Model is: 2.90107946940021
```

We observe that reducing $\boldsymbol{m}$ from 10 *(Bagging)* to 3 *(Random Forest)* increases the Test MSE a little bit.

This increase in MSE can be explained by the fact that in Random Forest, unlike Bagging (wherein each tree is exposed to all the predictors), each tree is exposed to a subset of the predictors.

```
##                %IncMSE IncNodePurity
## ShelveLoc   48.4045567     549.59174
## Price       40.2843961     508.51985
## Age         16.7054144     232.84697
## Advertising 19.8861191     211.35524
## CompPrice   12.4875929     185.69329
## Income       1.4748646     152.24336
## Population   0.1294146     150.96879
## Education    1.8102926      99.19403
## US           4.4167507      27.62612
## Urban       -1.7114750      17.57537
```

From the above table, we see that **ShelveLoc** and **Price** are the still most important predictors for *Sales*.

EXERCISE 8.11:

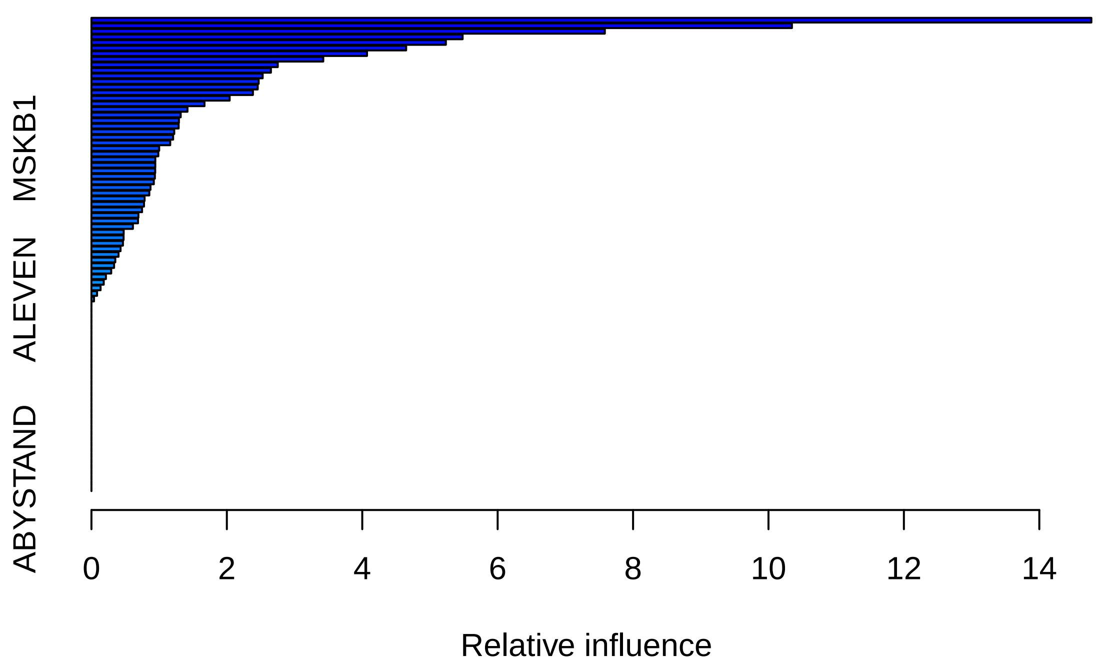
8.11.a

```
## Length of Caravan Dataset: 5822
```

```
## Length of Train Dataset : 1000
```

```
## Length of Test Dataset  : 4822
```

8.11.b




```
##                 var     rel.inf
## PPERSAUT PPERSAUT 14.76897821
## MKOOPKLA MKOOPKLA 10.34608977
## MOPLHOOG MOPLHOOG  7.58276819
## MBERMIDD MBERMIDD  5.48219989
## PBRAND     PBRAND  5.23496596
## ABRAND     ABRAND  4.64846137
## MGODGE     MGODGE  4.07035536
## MINK3045 MINK3045  3.42388385
## MAUT1       MAUT1  2.75186778
## MGODPR     MGODPR  2.65078748
## MOSTYPE   MOSTYPE  2.52899159
## MAUT2       MAUT2  2.47005442
## PWAPART   PWAPART  2.45561187
## MINKGEM   MINKGEM  2.38496124
## MBERARBG MBERARBG  2.04151595
## MSKC         MSKC  1.66946684
## MINK7512 MINK7512  1.41843892
## MRELGE     MRELGE  1.31818732
## MSKA         MSKA  1.29174003
## PBYSTAND PBYSTAND  1.28796768
## MFWEKIND MFWEKIND  1.22285160
## MBERHOOG MBERHOOG  1.20643199
## MGODOV     MGODOV  1.16347453
```

```
## MSKB1       MSKB1  1.00060171
## MHHUUR     MHHUUR  0.98767774
## MBERBOER MBERBOER  0.94327732
## MFGEKIND MFGEKIND  0.94278746
## MAUTO       MAUTO  0.94186003
## MGODRK     MGODRK  0.93716929
## MINK4575 MINK4575  0.92108809
## MRELOV     MRELOV  0.87214668
## MSKB2       MSKB2  0.85468878
## MOPLMIDD MOPLMIDD  0.78499519
## MOSHOOFD MOSHOOFD  0.77643271
## APERSAUT APERSAUT  0.74746748
## MSKD         MSKD  0.69351351
## MINKM30   MINKM30  0.68778644
## PLEVEN     PLEVEN  0.61251083
## MBERARBO MBERARBO  0.47629718
## PMOTSCO   PMOTSCO  0.47452037
## MZPART     MZPART  0.46517256
## MGEMLEEF MGEMLEEF  0.43011772
## MGEMOMV   MGEMOMV  0.40006320
## MZFONDS   MZFONDS  0.35216196
## MHKOOP     MHKOOP  0.33449542
## MINK123M MINK123M  0.29175841
## MFALLEEN MFALLEEN  0.21368001
## MRELSA     MRELSA  0.18161892
## MBERZELF MBERZELF  0.13498287
## MOPLLAAG MOPLLAAG  0.08308058
## ALEVEN     ALEVEN  0.03799572
## MAANTHUI MAANTHUI  0.00000000
## PWABEDR   PWABEDR  0.00000000
## PWALAND   PWALAND  0.00000000
## PBESAUT   PBESAUT  0.00000000
## PVRAAUT   PVRAAUT  0.00000000
## PAANHANG PAANHANG  0.00000000
## PTRACTOR PTRACTOR  0.00000000
## PWERKT     PWERKT  0.00000000
## PBROM       PBROM  0.00000000
## PPERSONG PPERSONG  0.00000000
## PGEZONG   PGEZONG  0.00000000
## PWAOREG   PWAOREG  0.00000000
## PZEILPL   PZEILPL  0.00000000
## PPLEZIER PPLEZIER  0.00000000
## PFIETS     PFIETS  0.00000000
## PINBOED   PINBOED  0.00000000
## AWAPART   AWAPART  0.00000000
## AWABEDR   AWABEDR  0.00000000
## AWALAND   AWALAND  0.00000000
## ABESAUT   ABESAUT  0.00000000
## AMOTSCO   AMOTSCO  0.00000000
## AVRAAUT   AVRAAUT  0.00000000
## AAANHANG AAANHANG  0.00000000
## ATRACTOR ATRACTOR  0.00000000
## AWERKT     AWERKT  0.00000000
## ABROM       ABROM  0.00000000
```

```
## APERSONG APERSONG  0.00000000
## AGEZONG   AGEZONG  0.00000000
## AWAOREG   AWAOREG  0.00000000
## AZEILPL   AZEILPL  0.00000000
## APLEZIER APLEZIER  0.00000000
## AFIETS     AFIETS  0.00000000
## AINBOED   AINBOED  0.00000000
## ABYSTAND ABYSTAND  0.00000000
```

From the Table and Graph, we can conclude that the top **Most-Important Predictors** are:

- PPERSAUT
- MKOOPKLA
- MOPLHOOG

8.11.c

Boosting

```
##           Reference
## Prediction    0    1
##          0 4412  255
##          1  121   34
```

```
## [Boosting] Fraction of the people predicted to make a purchase who in fact make one: 0.22
```

This value is also called the ***Precision***

Logistic Regression

```
##           Reference
## Prediction    0    1
##          0 4183  231
##          1  350   58
```

```
## [Logistic Regression]Fraction of the people predicted to make a purchase who in fact make one: 0.14
```

KNN

```
##                           2 102 202 302 402
## out_precision_knn 0.08362369  NA  NA  NA  NA
```

Simply applying KNN does not work because the training data is highly skewed. The following is the Summary of **Purchase** in Training Set:

```
##   0   1
## 941  59
```

Thus, the model ends up predicting everything as class-0 or *Not Purchased* Whole this achieves a High accuracy, the precision is low.

In order to improve the model, we probably need to *undersample* the majority class in our training data.

Summary: ***Boosting*** outperforms ***Logistic Regression***.

Problem-1 Beauty Pays:

Upon reading the *"BeautyData.csv"* file, we find that the following categorical columns are read as integers:

- female
- lower
- nonenglish
- tenuretrack

Thus converting the same into **factors**.

Summary:

```
##   CourseEvals     BeautyScore       female  lower   nonenglish tenuretrack
##  Min.   :1.944   Min.   :-1.53884   0:268   0:306   0:435      0:102
##  1st Qu.:3.326   1st Qu.:-0.74462   1:195   1:157   1: 28      1:361
##  Median :3.682   Median :-0.15636
##  Mean   :3.689   Mean   :-0.08835
##  3rd Qu.:4.067   3rd Qu.: 0.45725
##  Max.   :5.000   Max.   : 1.88167
```

1.1

In order to estimate the effect of *"Beauty"* on a professor's *"CourseEvaluation"*, taking into account the other determinants, we will fit a Multiple regression model > Model Summary

```
##
## Call:
## lm(formula = CourseEvals ~ ., data = Beauty)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.31385 -0.30202  0.01011  0.29815  1.04929
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.06542    0.05145  79.020  < 2e-16 ***
## BeautyScore   0.30415    0.02543  11.959  < 2e-16 ***
## female1      -0.33199    0.04075  -8.146 3.62e-15 ***
## lower1       -0.34255    0.04282  -7.999 1.04e-14 ***
## nonenglish1  -0.25808    0.08478  -3.044  0.00247 **
## tenuretrack1 -0.09945    0.04888  -2.035  0.04245 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4273 on 457 degrees of freedom
## Multiple R-squared:  0.3471, Adjusted R-squared:  0.3399
## F-statistic: 48.58 on 5 and 457 DF,  p-value: < 2.2e-16
```

From the above model summary, we see that a person's beauty actually has a statistically significant impact on their course evaluation.

Holding other determinants constant, the ***CourseEval*** is directly proportional to ***BeautyScore*** by a factor of **0.30415**

1.2

Problem-2 Housing Price Structure:

Upon reading the *"MidCity.csv"* file, we find the following:

- **Home:** This is the ID and is same as the row number and hence, can be dropped.
- **Bricks:** This is a categorical Variable and thus need to convert it
- **Nbhd:** This is the Neighbourhood. This is a Categorical Value too and needs to be converted into factors
- **Sqft:** This is the Square Feet area of the House. While the other predictors are in a range of 0-6, Sqft. is way outside. Thus scaling this parameter down will allow for a better regression fit.

  Summary:

```
##  Nbhd       Offers            SqFt.V1         Brick       Bedrooms
##  1:44   Min.   :1.000   Min.   :-2.6040137   No :86   Min.   :2.000
##  2:45   1st Qu.:2.000   1st Qu.:-0.5716128   Yes:42   1st Qu.:3.000
##  3:39   Median :3.000   Median :-0.0044311            Median :3.000
##         Mean   :2.578   Mean   : 0.0000000            Mean   :3.023
##         3rd Qu.:3.000   3rd Qu.: 0.6572808            3rd Qu.:3.000
##         Max.   :6.000   Max.   : 2.7842120            Max.   :5.000
##    Bathrooms         Price
##  Min.   :2.000   Min.   : 69100
##  1st Qu.:2.000   1st Qu.:111325
##  Median :2.000   Median :125950
##  Mean   :2.445   Mean   :130427
##  3rd Qu.:3.000   3rd Qu.:148250
##  Max.   :4.000   Max.   :211200
```

Each of the Predictors *(Nbhd,Offers,Brick,Bedrooms,Bathrooms)* have similar range, therefore no scaling required.

2.1

To understand the impact of the being a Brick house on Price, all other factors considered, we will fit a Multiple regression model

  Model Summary:

```
##
## Call:
## lm(formula = Price ~ ., data = MidCity)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27337.3  -6549.5    -41.7   5803.4  27359.3
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   108197       6689  16.175  < 2e-16 ***
## Nbhd2          -1561       2397  -0.651  0.51621
## Nbhd3          20681       3149   6.568 1.38e-09 ***
## Offers         -8268       1085  -7.621 6.47e-12 ***
## SqFt           11212       1213   9.242 1.10e-15 ***
## BrickYes       17297       1982   8.729 1.78e-14 ***
## Bedrooms        4247       1598   2.658  0.00894 **
## Bathrooms       7883       2117   3.724  0.00030 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10020 on 120 degrees of freedom
## Multiple R-squared:  0.8686, Adjusted R-squared:  0.861
## F-statistic: 113.3 on 7 and 120 DF,  p-value: < 2.2e-16
```

From the above model summary, we see that the parameter: **BrickYes** is statistically significant *(assuming a 95% confidence interval for the hypothesis* $\beta_{BrickYes} = 0$ *)* and has a positive impact on the Price. Thus, holding other factors constant, being a Brick House increases the Price of a house by $17,297.

In other words, **Premium for a Brick House is estimated to be $17,297.**

2.2

Yes, There is a premium for a House in Neighbourhood 3

All other factors held constant, a house in Neighbouhood 3 fetches:

- $20,681 more than the same house in Neighbourhood 1
- $22,242 more than the same house in Neighbourhood 2

2.3

In order to account for plausible interactions between predictors, a new feature: **Brick_Nbhd3** has to be added. which will be 1 iff the house is a brick house in Neighbourhood 3

```
##
## Call:
## lm(formula = Price ~ ., data = MidCity)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -26939.1  -5428.7   -213.9   4519.3  26211.4
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      111190       6668  16.675  < 2e-16 ***
## Nbhd2              -673       2376  -0.283  0.77751
## Nbhd3             17241       3391   5.084 1.39e-06 ***
## Offers            -8401       1064  -7.893 1.62e-12 ***
## SqFt              11439       1192   9.593  < 2e-16 ***
## BrickYes          13826       2406   5.748 7.11e-08 ***
```

```
## Bedrooms           4718       1578   2.991  0.00338 **
## Bathrooms          6463       2154   3.000  0.00329 **
## Brick_Nbhd3Yes    10182       4165   2.444  0.01598 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9817 on 119 degrees of freedom
## Multiple R-squared:  0.8749, Adjusted R-squared:  0.8665
## F-statistic:   104 on 8 and 119 DF,  p-value: < 2.2e-16
```

From the above model summary, we see that the parameter: **Brick_Nbhd3No** is statistically significant and has a negative impact on the Price. Thus, holding other factors constant, being a Brick House in Neighbourhood 3 increases the Price of a house by $10,182.

In other words, **Extra Premium for a Brick House in Neighbourhood 3 is estimated to be $10,182.**

NOTE: *This interaction is only statistically significant if we assume a 95% confidence interval. A less restrictive confidence interval, say 99%, might make the interaction statistically insignificant since the p-value will lie within the non-rejection region of the hypothesis* $\beta_{Brick\_Nbhd3No} = 0$

2.4

Yes, we may merge Neighbourhood 2 and 1 into a common "Old" class.

In our models, both **Nbhd1** and **Nbhd2** are statistical significant *(considering a 95% confidence interval for the hypothesis)* $\beta_{Nbhd1} = 0$ *and* $\beta_{Nbhd2} = 0$ Thus, we fail to reject the hypothesis that Nbhd2 has a 0 coefficient. This implies that belonging to Neighbourhood2 specifically has a significant impact on house price. So, merging Neighbourhood 2 and 1 into a common class would be unwise.

Problem-3 What Causes What??:

3.1

For our models to perform well, we need to ensure that our datasets do not have any inherent biases.

At first sight, it might seem that *"getting data from a few different cities and running the regression of Crime on Police to understand how more cops in the streets affect crime"* is a good idea. However, we need to consider that the density of Police in a city is a policy decision which is taken in response to existing Crime Rates.

Thus, our model will end up reinforcing our bias: *"More Cops reduce Crime."*

However, to get a good dataset, free of biases, we must **conduct experiments randomly.** One way to achieve this would be by deploying different number of cops across different cities and alert levels over multiple days.

3.2

To obtain their dataset, the researchers from UPENN were able to get the data for an experiment that saw a high volume of police on the streets on high terror alert days.

They observed a low crime rate on high alert days, when more police were deployed on the streets. However, higher police activity was observed on those days to monitor and curb terrorist activities; not for controlling petty crimes.

Similarly, as shown in the table 2, the UPENN researchers considered the metro ridership information, to confirm that low crime rates was not a result of lesser movement of possible criminals or victims.

Nevertheless, there exists an alternate explanation for crime rates being be low on high alert days. High alert implies higher probability of a terrorist incident, vis-a-vis being the victim of one. Thus the criminals, like the common man, avoid exposing themselves to this threat by staying indoors. This would mean that the observed lower crime rate are police activity are not directly related but the high alert is an underlying *confounding variable.*

For the purpose of this experiment, as indicated by the table-2, if the ridership is kept constant on high alert days, *more police would reduce crime rate.*

3.3

As mentioned above, METRO ridership had to be controlled to ensure that it was not affecting crime rates. This would help the UPENN researchers establish a more clear relationship between "Police" and "Crime". Consequently, according to the experiment, it was observed that keeping the metro ridership fixed for high alert days, **more police reduced crime rates**.

3.4

Table 4 has further refined the analysis to check if the effect of high alert days on crime was the same in all areas of town.

Using interactions between *location* and *high alert days* it was observed the effect is only statistically significant *(with a confidence interval of 95%)* in district 1. The effect in the 5 other districts, though still negative, is not statistically significant.

Problem-4 Neural Nets:

Step-1:

First, we split the data-set into **Train, Cross-Validation** and **Test** sets

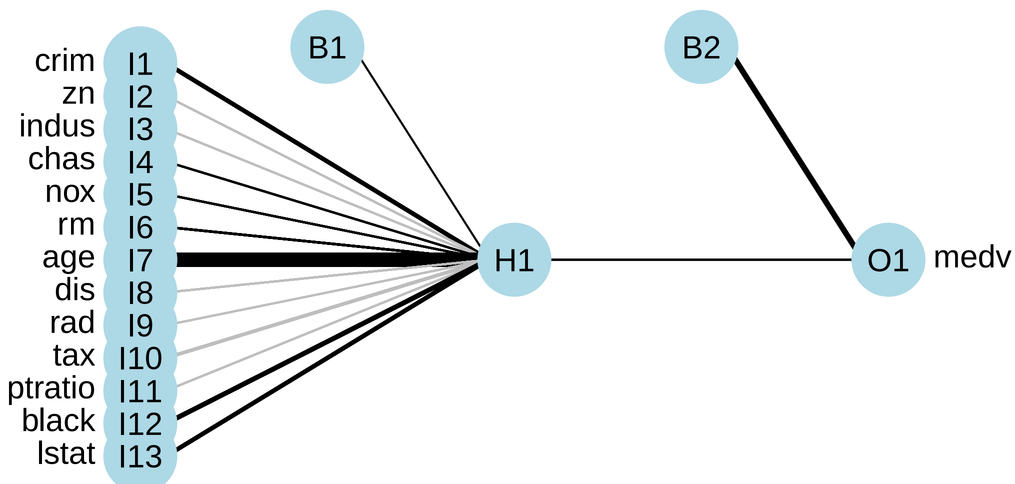
```
## Length of Boston Dataset           : 506

## Length of Train Dataset            : 404

## Length of Cross Validation Dataset : 51

## Length of Test Dataset             : 51
```

Step-2:

Fitting a single layer neural net to predict **medv** based on all other variables in Boston dataset as predictors:

```
## # weights:  29
## initial  value 25703914.918253
## iter  10 value 666560.910142
## iter  20 value 8716.661658
## final  value 8704.559128
## converged
```

```
## a 13-1-1 network with 29 weights
## inputs: crim zn indus chas nox rm age dis rad tax ptratio black lstat
## output(s): medv
## options were - skip-layer connections  linear output units
```




```
## RMSE of a single-layer neural network is:  4.852
```

Step-3:

Cross-Validating over different **Size** and **Decay** Parameters:

```
##           0      0.1      0.3     0.01     0.03    0.001    0.003    1e-04
```

```
## 1  5.385722 5.390818 5.445314 5.423088 5.375968 5.361723 5.362722 5.361274
## 2  5.361227 5.430607 4.937380 5.364228 5.629962 5.361531 5.362722 5.149887
## 5  5.379487 5.180211 5.361882 5.266550 4.577101 5.378594 5.528865 5.369206
## 10 5.368180 5.332621 5.606176 5.085475 5.247146 5.458245 5.324589 5.365822
## 20 5.173832 5.316855 5.250776 4.905512 5.336179 8.050774 5.417361 5.378064
## 30 5.938562 5.228055 5.965700 5.746113 5.354573 7.178072 5.363293 5.377685
##       3e-04    1e-05    3e-05
## 1  5.361374 5.361230 5.281073
## 2  4.518851 5.362142 5.298272
## 5  5.685748 5.361231 5.353238
## 10 5.379146 5.284663 5.349404
## 20 5.206584 5.347391 5.449923
## 30 4.916482 5.160412 5.503535
```

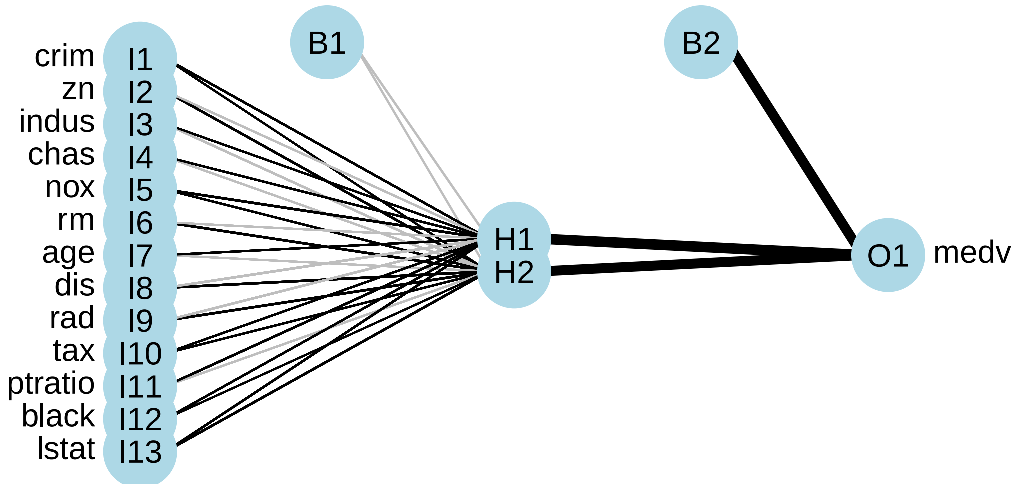
```
## Minimum Cross-Validation RMSE of 4.51885087058788 achieved with a Neural Net with the following specs
## Size: 2
## Decay: 3e-04
```

Step-4:

Test Error on the Test Set with the Neural Net chosen from the Cross-Validation set

```
## # weights:  44
## initial  value 35011181.757967
## iter  10 value 1778150.833349
## iter  20 value 8704.832493
## final  value 8704.798073
## converged
```

```
## a 13-2-1 network with 44 weights
## inputs: crim zn indus chas nox rm age dis rad tax ptratio black lstat
## output(s): medv
## options were - skip-layer connections  linear output units  decay=3e-04
```




```
##
##
## RMSE of a Neural Net of Size: 2 and Decay: 3e-04 is-
##  4.358
```

Problem-5 Final Project:
