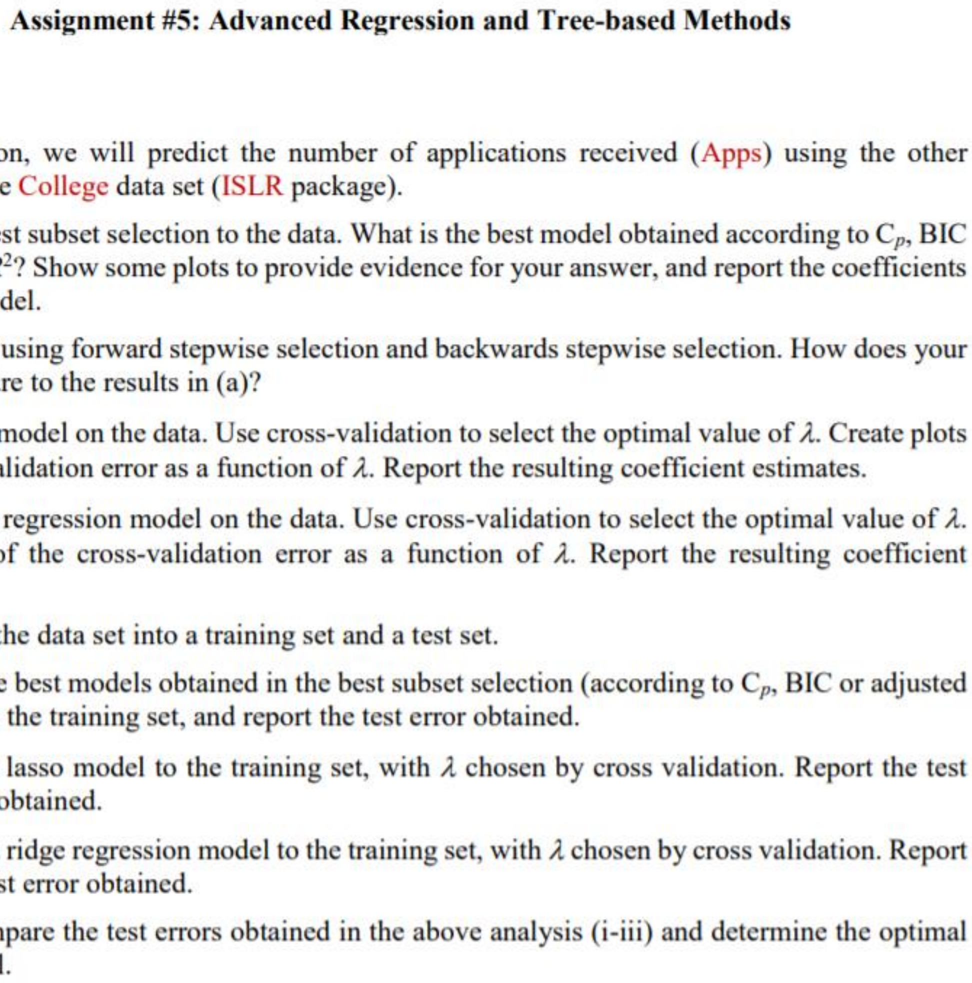


Assignment #5: Advanced Regression and Tree-based Methods

Problem 1

In this question, we will predict the number of applications received (Apps) using the other variables in the College data set (ISLR package).

(a) Perform best subset selection to the data. What is the best model obtained according to C_p, BIC and adjusted R^2? Show some plots to provide evidence for your answer, and report the coefficients of the best model.

(b) Repeat (a) using forward stepwise selection and backwards stepwise selection. How does your answer compare to the results in (a)?

(c) Fit a lasso model on the data. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates.

(d) Fit a ridge regression model on the data. Use cross-validation to select the optimal value of λ. Create plots of the cross-validation error as a function of λ. Report the resulting coefficient estimates.

(e) Now split the data set into a training set and a test set.

i. Fit the best models obtained in the best subset selection (according to C_p, BIC or adjusted R^2) to the training set, and report the test error obtained.

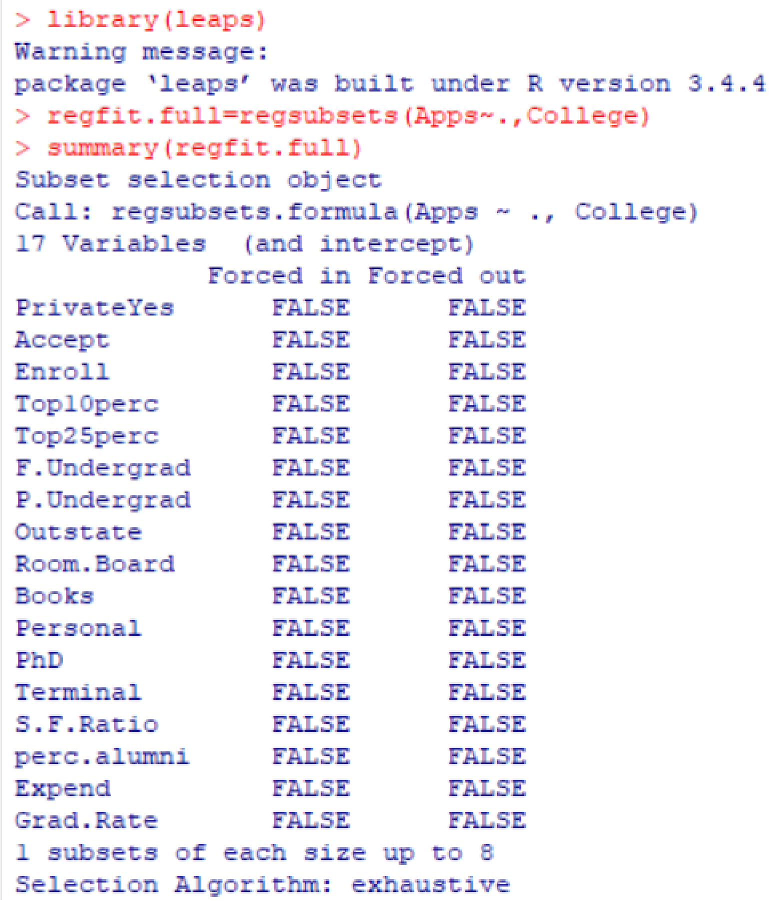
ii. Fit a lasso model to the training set, with λ chosen by cross validation. Report the test error obtained.

iii. Fit a ridge regression model to the training set, with λ chosen by cross validation. Report the test error obtained.

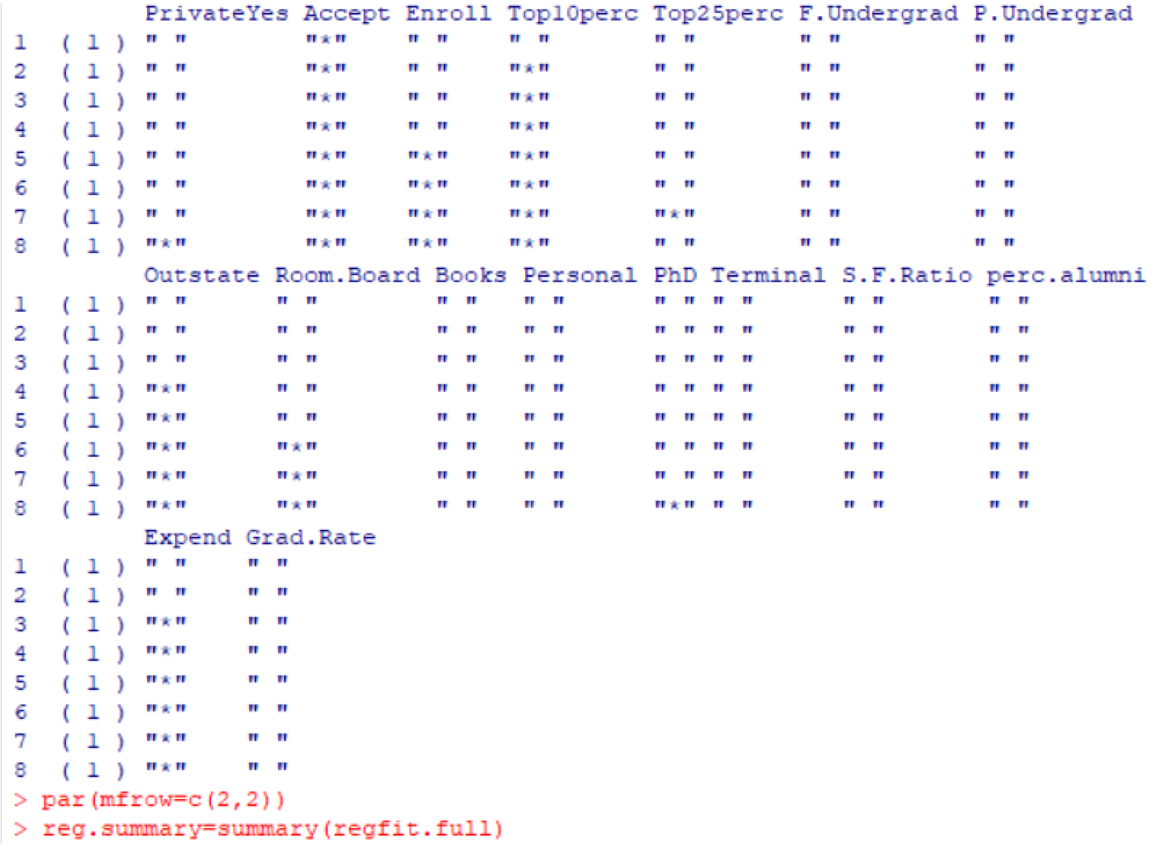
iv. Compare the test errors obtained in the above analysis (i-iii) and determine the optimal model.

Ans 1:

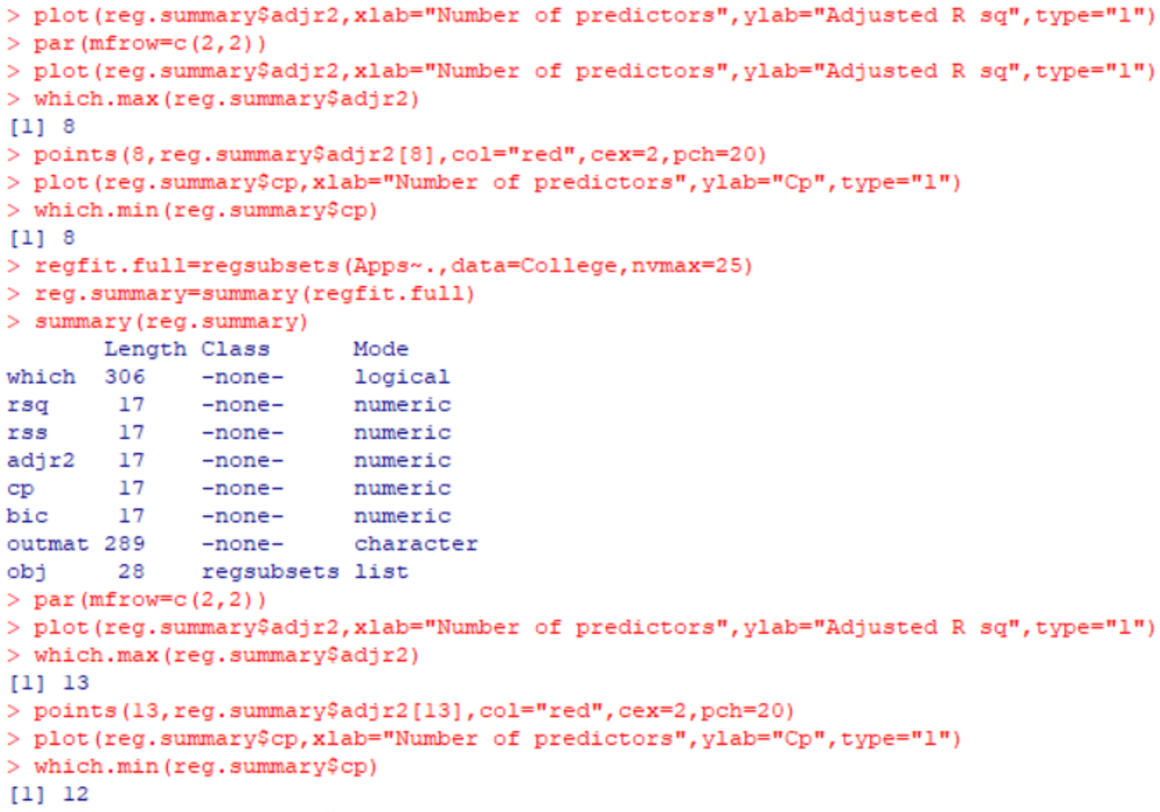
(a).



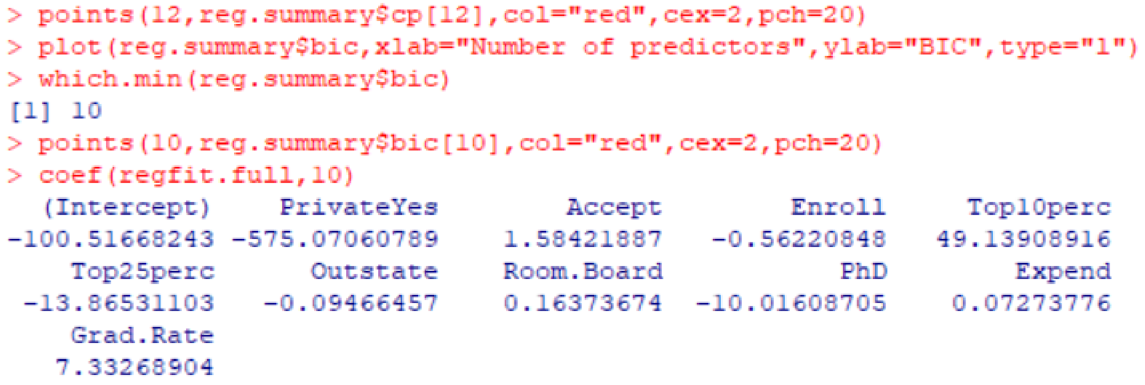
```
> library(leaps)
Warning message:
package 'leaps' was built under R version 3.4.4
> regfit.full=regsubsets(Apps~.,College)
> summary(regfit.full)
Subset selection object
Call: regsubsets.formula(Apps ~ ., College)
17 Variables  (and intercept)
            Forced in Forced out
PrivateYes      FALSE      FALSE
Accept          FALSE      FALSE
Enroll          FALSE      FALSE
Top10perc       FALSE      FALSE
Top25perc       FALSE      FALSE
F.Undergrad     FALSE      FALSE
P.Undergrad     FALSE      FALSE
Outstate        FALSE      FALSE
Room.Board      FALSE      FALSE
Books           FALSE      FALSE
Personal        FALSE      FALSE
PhD             FALSE      FALSE
Terminal        FALSE      FALSE
S.F.Ratio       FALSE      FALSE
perc.alumni     FALSE      FALSE
Expend          FALSE      FALSE
Grad.Rate       FALSE      FALSE
1 subsets of each size up to 8
Selection Algorithm: exhaustive
```



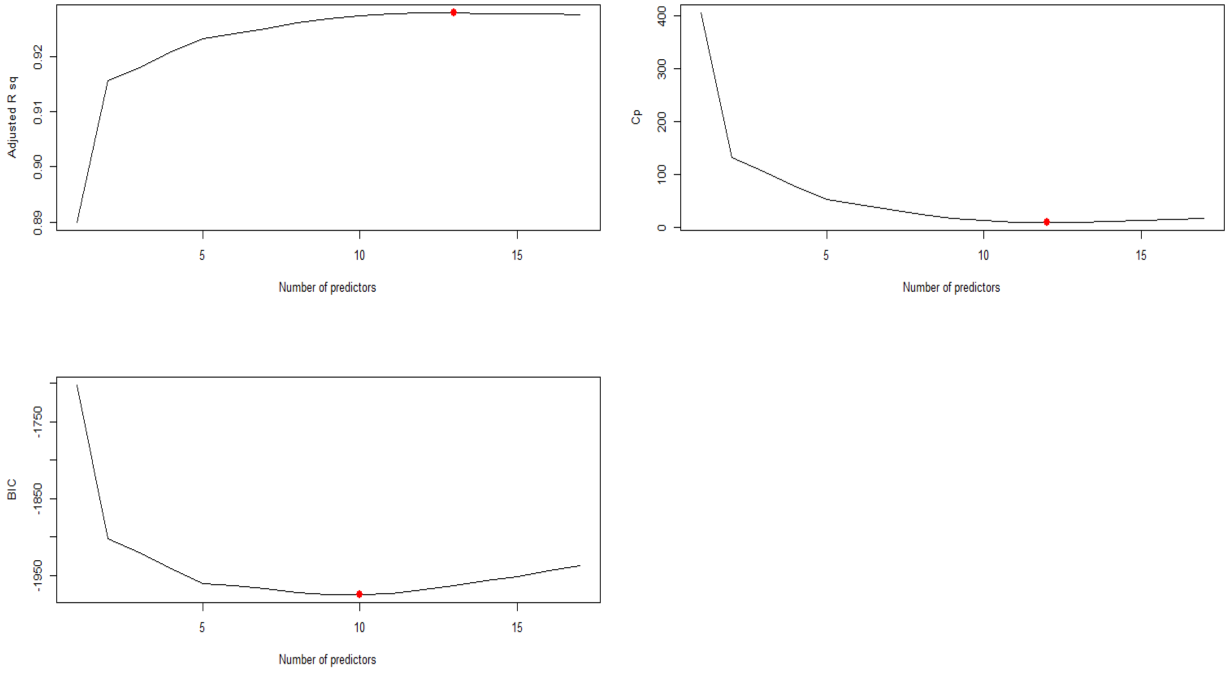
```
         PrivateYes Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad
1  ( 1 ) " "        "*"    " "    " "       " "       " "         " "        
2  ( 1 ) " "        "*"    " "    "*"       " "       " "         " "        
3  ( 1 ) " "        "*"    " "    "*"       " "       " "         " "        
4  ( 1 ) " "        "*"    " "    "*"       " "       " "         " "        
5  ( 1 ) " "        "*"    "*"    "*"       " "       " "         " "        
6  ( 1 ) " "        "*"    "*"    "*"       " "       " "         " "        
7  ( 1 ) " "        "*"    "*"    "*"       "*"       " "         " "        
8  ( 1 ) "*"        "*"    "*"    "*"       " "       " "         " "        
         Outstate Room.Board Books Personal PhD Terminal S.F.Ratio perc.alumni
1  ( 1 ) " "      " "        " "   " "      " " " "      " "       " "        
2  ( 1 ) " "      " "        " "   " "      " " " "      " "       " "        
3  ( 1 ) " "      " "        " "   " "      " " " "      " "       " "        
4  ( 1 ) "*"      " "        " "   " "      " " " "      " "       " "        
5  ( 1 ) "*"      " "        " "   " "      " " " "      " "       " "        
6  ( 1 ) "*"      "*"        " "   " "      " " " "      " "       " "        
7  ( 1 ) "*"      "*"        " "   " "      " " " "      " "       " "        
8  ( 1 ) "*"      "*"        " "   " "      "*" " "      " "       " "        
         Expend Grad.Rate
1  ( 1 ) " "    " "      
2  ( 1 ) " "    " "      
3  ( 1 ) "*"    " "      
4  ( 1 ) "*"    " "      
5  ( 1 ) "*"    " "      
6  ( 1 ) "*"    " "      
7  ( 1 ) "*"    " "      
8  ( 1 ) "*"    " "      
> par(mfrow=c(2,2))
> reg.summary=summary(regfit.full)
```



```
> plot(reg.summary$adjr2,xlab="Number of predictors",ylab="Adjusted R sq",type="l")
> par(mfrow=c(2,2))
> plot(reg.summary$adjr2,xlab="Number of predictors",ylab="Adjusted R sq",type="l")
> which.max(reg.summary$adjr2)
[1] 8
> points(8,reg.summary$adjr2[8],col="red",cex=2,pch=20)
> plot(reg.summary$cp,xlab="Number of predictors",ylab="Cp",type="l")
> which.min(reg.summary$cp)
[1] 8
> regfit.full=regsubsets(Apps~.,data=College,nvmax=25)
> reg.summary=summary(regfit.full)
> summary(reg.summary)
       Length Class      Mode     
which  306    -none-     logical  
rsq     17    -none-     numeric  
rss     17    -none-     numeric  
adjr2   17    -none-     numeric  
cp      17    -none-     numeric  
bic     17    -none-     numeric  
outmat 289    -none-     character
obj     28    regsubsets list     
> par(mfrow=c(2,2))
> plot(reg.summary$adjr2,xlab="Number of predictors",ylab="Adjusted R sq",type="l")
> which.max(reg.summary$adjr2)
[1] 13
> points(13,reg.summary$adjr2[13],col="red",cex=2,pch=20)
> plot(reg.summary$cp,xlab="Number of predictors",ylab="Cp",type="l")
> which.min(reg.summary$cp)
[1] 12
```



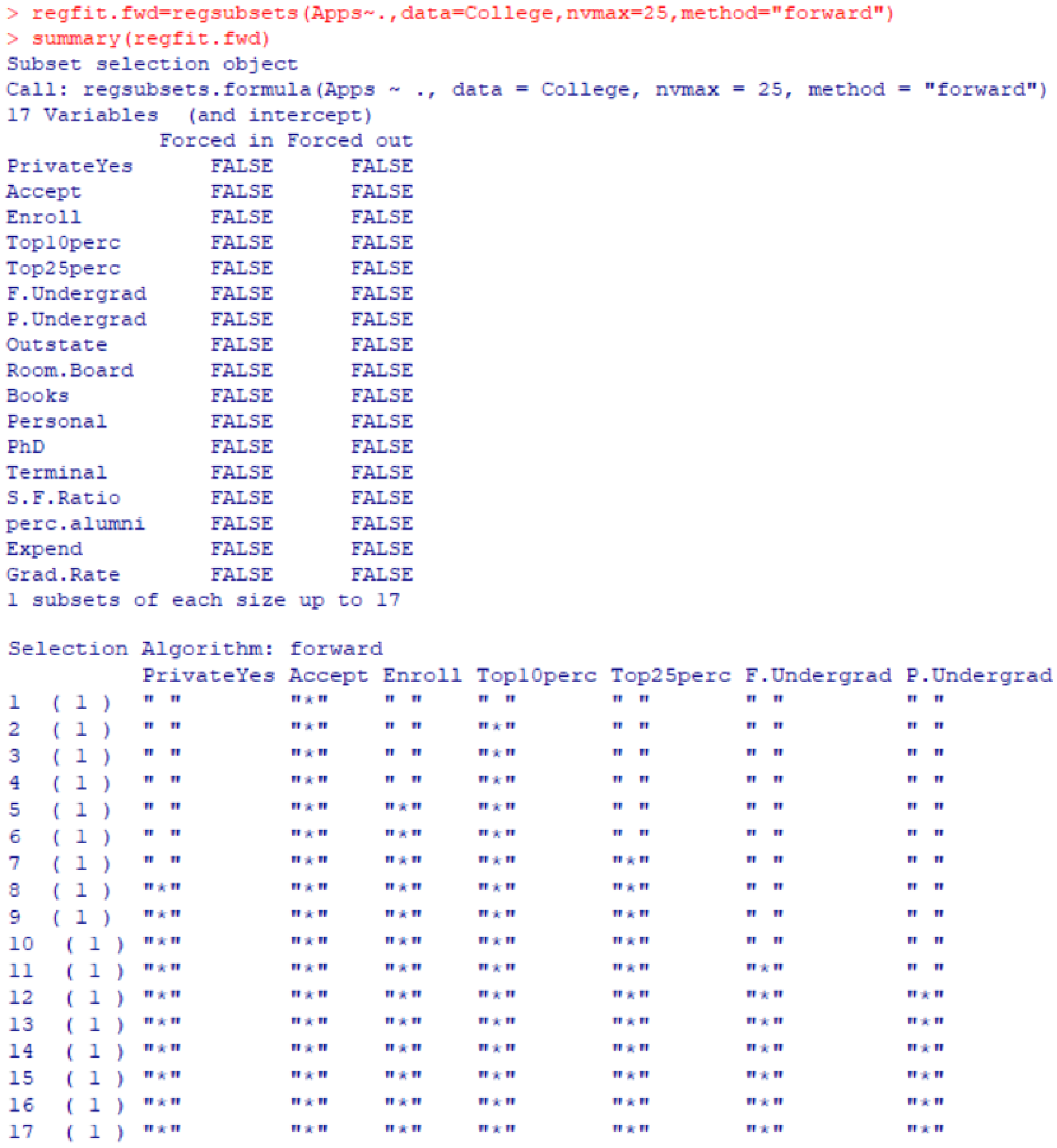
```
> points(12,reg.summary$cp[12],col="red",cex=2,pch=20)
> plot(reg.summary$bic,xlab="Number of predictors",ylab="BIC",type="l")
> which.min(reg.summary$bic)
[1] 10
> points(10,reg.summary$bic[10],col="red",cex=2,pch=20)
> coef(regfit.full,10)
  (Intercept)   PrivateYes       Accept       Enroll    Top10perc 
-100.51668243 -575.07060789   1.58421887  -0.56220848  49.13908916 
    Top25perc     Outstate   Room.Board          PhD       Expend 
 -13.86531103  -0.09466457   0.16373674 -10.01608705   0.07273776 
    Grad.Rate 
   7.33268904 
```



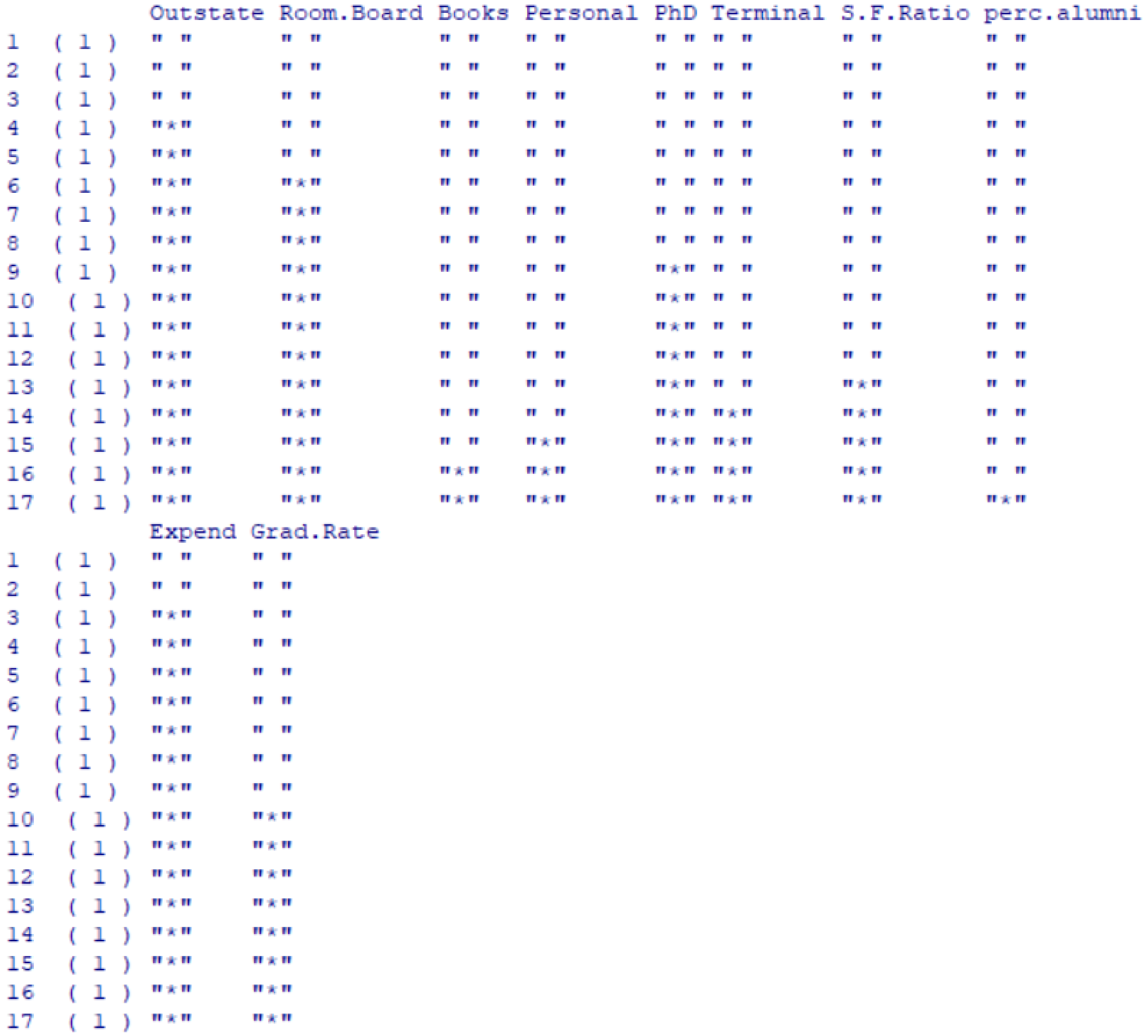
Comparing the above 3 techniques, we can see that the no of predictors used in adjusted R squared technique, Cp and BIC are 13, 12 and 10 respectively which we can see from the graph which is highlighted as the red point in the graph.

Thereby, we can state that BIC uses the least no of predictors to explain our model and thus can be stated as the best subset selection method.

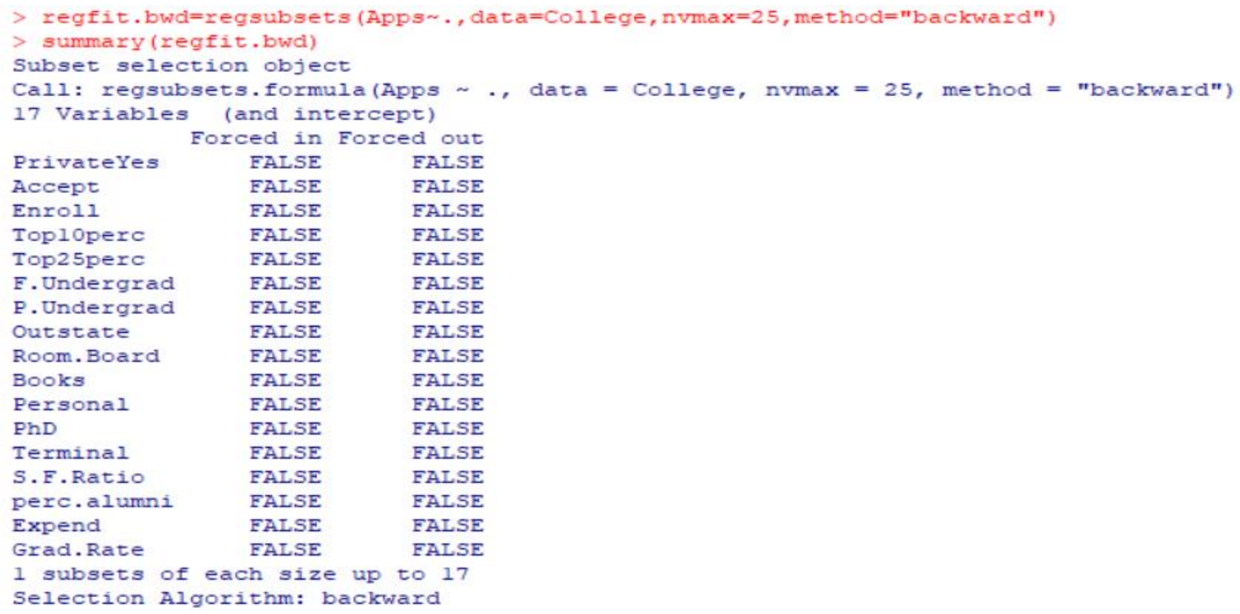
(b).



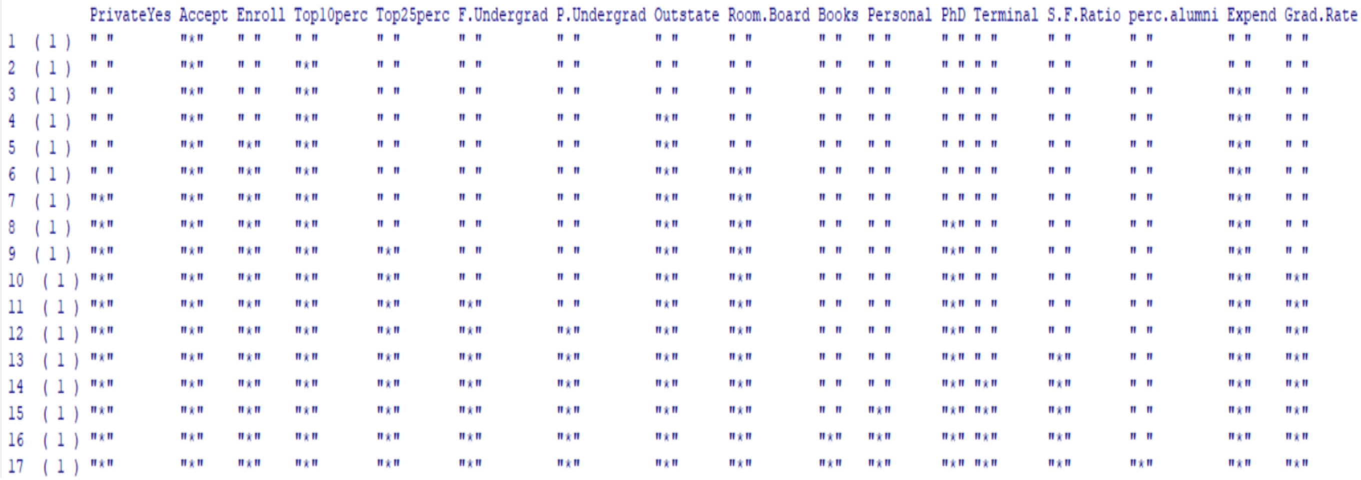
```
> regfit.fwd=regsubsets(Apps~.,data=College,nvmax=25,method="forward")
> summary(regfit.fwd)
Subset selection object
Call: regsubsets.formula(Apps ~ ., data = College, nvmax = 25, method = "forward")
17 Variables  (and intercept)
            Forced in Forced out
PrivateYes      FALSE      FALSE
Accept          FALSE      FALSE
Enroll          FALSE      FALSE
Top10perc       FALSE      FALSE
Top25perc       FALSE      FALSE
F.Undergrad     FALSE      FALSE
P.Undergrad     FALSE      FALSE
Outstate        FALSE      FALSE
Room.Board      FALSE      FALSE
Books           FALSE      FALSE
Personal        FALSE      FALSE
PhD             FALSE      FALSE
Terminal        FALSE      FALSE
S.F.Ratio       FALSE      FALSE
perc.alumni     FALSE      FALSE
Expend          FALSE      FALSE
Grad.Rate       FALSE      FALSE
1 subsets of each size up to 17

Selection Algorithm: forward
          PrivateYes Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad
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 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"
14  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"
15  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"
16  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"
17  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"
```



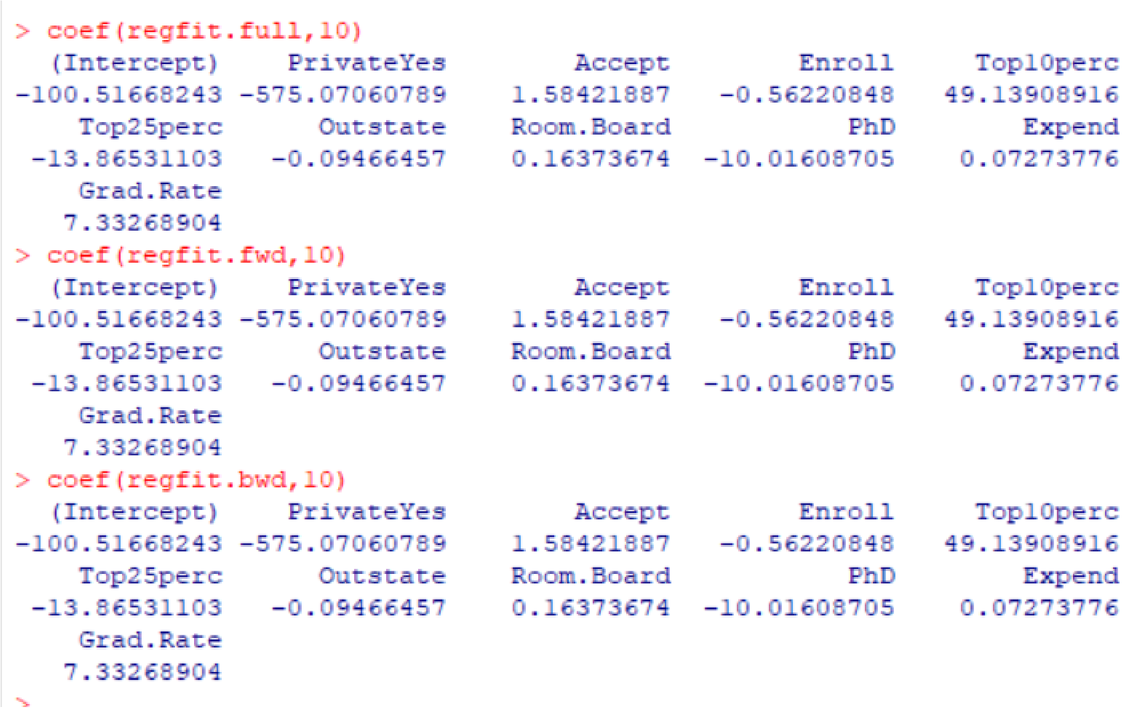
```
          Outstate Room.Board Books Personal PhD Terminal S.F.Ratio perc.alumni
1  ( 1 )  " "      " "        " "   " "      " " " "      " "       " "        
2  ( 1 )  " "      " "        " "   " "      " " " "      " "       " "        
3  ( 1 )  " "      " "        " "   " "      " " " "      " "       " "        
4  ( 1 )  "*"      " "        " "   " "      " " " "      " "       " "        
5  ( 1 )  "*"      " "        " "   " "      " " " "      " "       " "        
6  ( 1 )  "*"      "*"        " "   " "      " " " "      " "       " "        
7  ( 1 )  "*"      "*"        " "   " "      " " " "      " "       " "        
8  ( 1 )  "*"      "*"        " "   " "      " " " "      " "       " "        
9  ( 1 )  "*"      "*"        " "   " "      "*" " "      " "       " "        
10  ( 1 ) "*"      "*"        " "   " "      "*" " "      " "       " "        
11  ( 1 ) "*"      "*"        " "   " "      "*" " "      " "       " "        
12  ( 1 ) "*"      "*"        " "   " "      "*" " "      " "       " "        
13  ( 1 ) "*"      "*"        " "   " "      "*" " "      "*"       " "        
14  ( 1 ) "*"      "*"        " "   " "      "*" "*"      "*"       " "        
15  ( 1 ) "*"      "*"        " "   "*"      "*" "*"      "*"       " "        
16  ( 1 ) "*"      "*"        "*"   "*"      "*" "*"      "*"       " "        
17  ( 1 ) "*"      "*"        "*"   "*"      "*" "*"      "*"       "*"        
          Expend Grad.Rate
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 ) "*"    "*"      
14  ( 1 ) "*"    "*"      
15  ( 1 ) "*"    "*"      
16  ( 1 ) "*"    "*"      
17  ( 1 ) "*"    "*"      
```



```
> regfit.bwd=regsubsets(Apps~.,data=College,nvmax=25,method="backward")
> summary(regfit.bwd)
Subset selection object
Call: regsubsets.formula(Apps ~ ., data = College, nvmax = 25, method = "backward")
17 Variables  (and intercept)
           Forced in Forced out
PrivateYes     FALSE      FALSE
Accept         FALSE      FALSE
Enroll         FALSE      FALSE
Top10perc      FALSE      FALSE
Top25perc      FALSE      FALSE
F.Undergrad    FALSE      FALSE
P.Undergrad    FALSE      FALSE
Outstate       FALSE      FALSE
Room.Board     FALSE      FALSE
Books          FALSE      FALSE
Personal       FALSE      FALSE
PhD            FALSE      FALSE
Terminal       FALSE      FALSE
S.F.Ratio      FALSE      FALSE
perc.alumni    FALSE      FALSE
Expend         FALSE      FALSE
Grad.Rate      FALSE      FALSE
1 subsets of each size up to 17
Selection Algorithm: backward
```



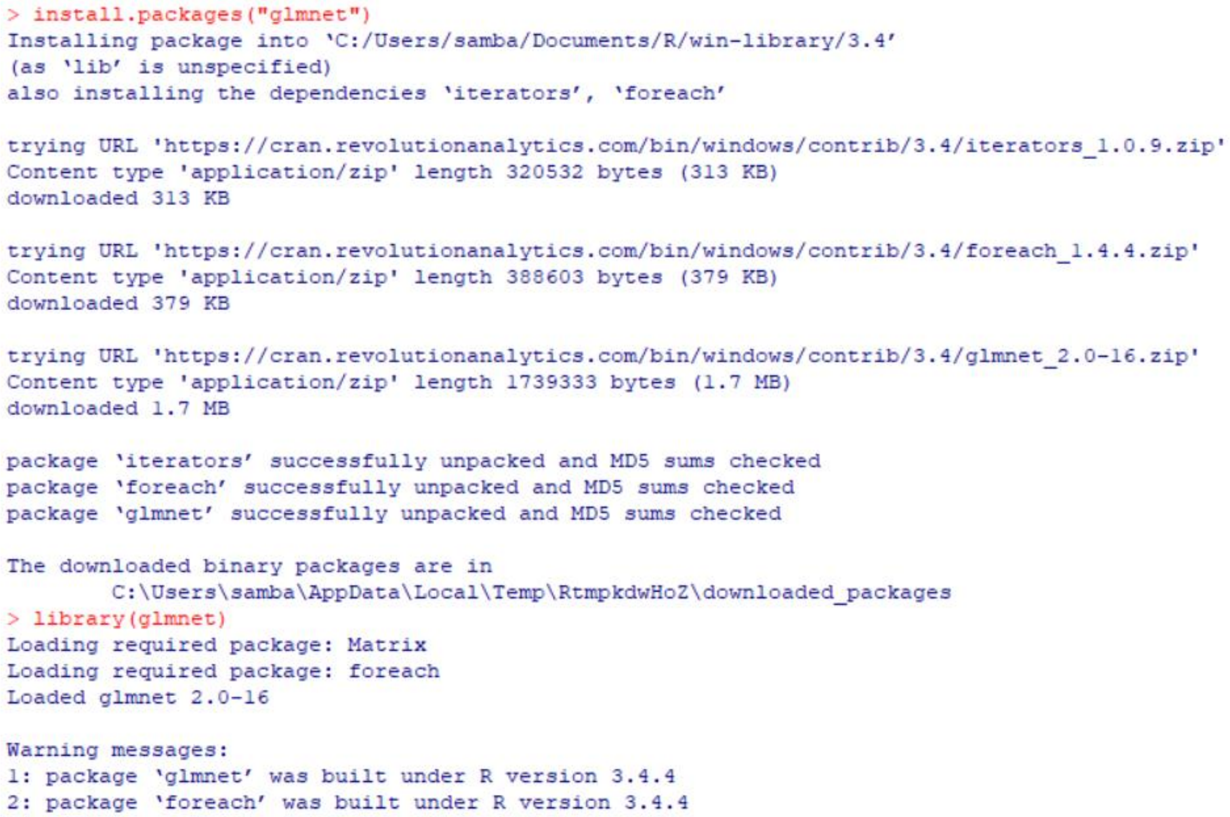
```
          PrivateYes Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad Outstate Room.Board Books Personal PhD Terminal S.F.Ratio perc.alumni Expend Grad.Rate
1  ( 1 )  " "        "*"    " "    " "       " "       " "         " "         " "      " "        " "   " "      " " " "      " "       " "         " "    " "
2  ( 1 )  " "        "*"    " "    "*"       " "       " "         " "         " "      " "        " "   " "      " " " "      " "       " "         " "    " "
3  ( 1 )  " "        "*"    " "    "*"       " "       " "         " "         " "      " "        " "   " "      " " " "      " "       " "         "*"    " "
4  ( 1 )  " "        "*"    " "    "*"       " "       " "         " "         "*"      " "        " "   " "      " " " "      " "       " "         "*"    " "
5  ( 1 )  " "        "*"    "*"    "*"       " "       " "         " "         "*"      " "        " "   " "      " " " "      " "       " "         "*"    " "
6  ( 1 )  " "        "*"    "*"    "*"       " "       " "         " "         "*"      "*"        " "   " "      " " " "      " "       " "         "*"    " "
7  ( 1 )  "*"        "*"    "*"    "*"       " "       " "         " "         "*"      "*"        " "   " "      " " " "      " "       " "         "*"    " "
8  ( 1 )  "*"        "*"    "*"    "*"       " "       " "         " "         "*"      "*"        " "   " "      "*" " "      " "       " "         "*"    " "
9  ( 1 )  "*"        "*"    "*"    "*"       "*"       " "         " "         "*"      "*"        " "   " "      "*" " "      " "       " "         "*"    " "
10  ( 1 ) "*"        "*"    "*"    "*"       "*"       " "         " "         "*"      "*"        " "   " "      "*" " "      " "       " "         "*"    "*"
11  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         " "         "*"      "*"        " "   " "      "*" " "      " "       " "         "*"    "*"
12  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"         "*"      "*"        " "   " "      "*" " "      " "       " "         "*"    "*"
13  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"         "*"      "*"        " "   " "      "*" " "      "*"       " "         "*"    "*"
14  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"         "*"      "*"        " "   " "      "*" "*"      "*"       " "         "*"    "*"
15  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"         "*"      "*"        " "   "*"      "*" "*"      "*"       " "         "*"    "*"
16  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"         "*"      "*"        "*"   "*"      "*" "*"      "*"       " "         "*"    "*"
17  ( 1 ) "*"        "*"    "*"    "*"       "*"       "*"         "*"         "*"      "*"        "*"   "*"      "*" "*"      "*"       "*"         "*"    "*"
```



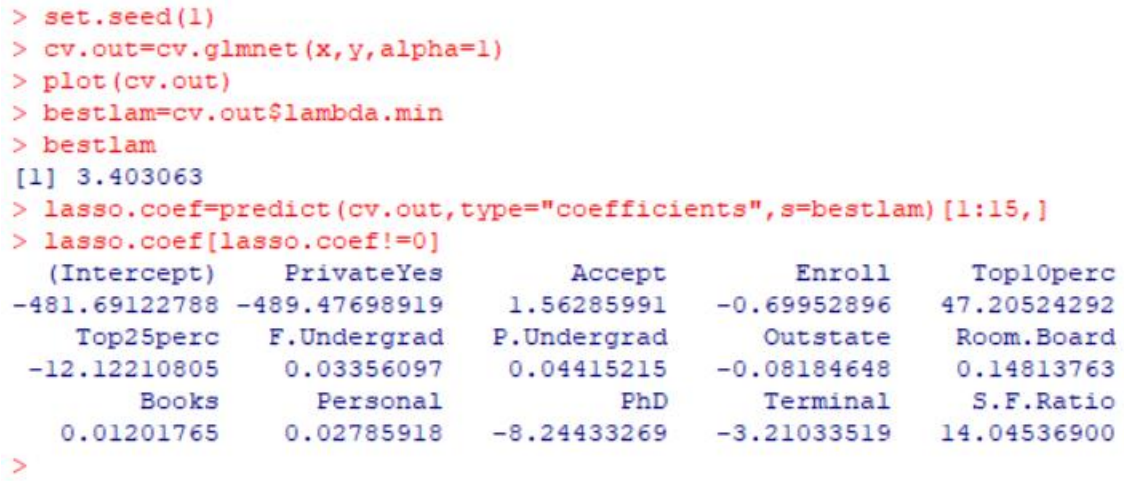
```
> coef(regfit.full,10)
  (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
-100.51668243 -575.07060789    1.58421887   -0.56220848   49.13908916 
    Top25perc      Outstate    Room.Board           PhD        Expend 
 -13.86531103   -0.09466457    0.16373674  -10.01608705    0.07273776 
    Grad.Rate 
   7.33268904 
> coef(regfit.fwd,10)
  (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
-100.51668243 -575.07060789    1.58421887   -0.56220848   49.13908916 
    Top25perc      Outstate    Room.Board           PhD        Expend 
 -13.86531103   -0.09466457    0.16373674  -10.01608705    0.07273776 
    Grad.Rate 
   7.33268904 
> coef(regfit.bwd,10)
  (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
-100.51668243 -575.07060789    1.58421887   -0.56220848   49.13908916 
    Top25perc      Outstate    Room.Board           PhD        Expend 
 -13.86531103   -0.09466457    0.16373674  -10.01608705    0.07273776 
    Grad.Rate 
   7.33268904 
>
```

In all the three cases, when we consider 10 predictors in our model, there is no change in the values of the coefficients in all the 3 approaches.

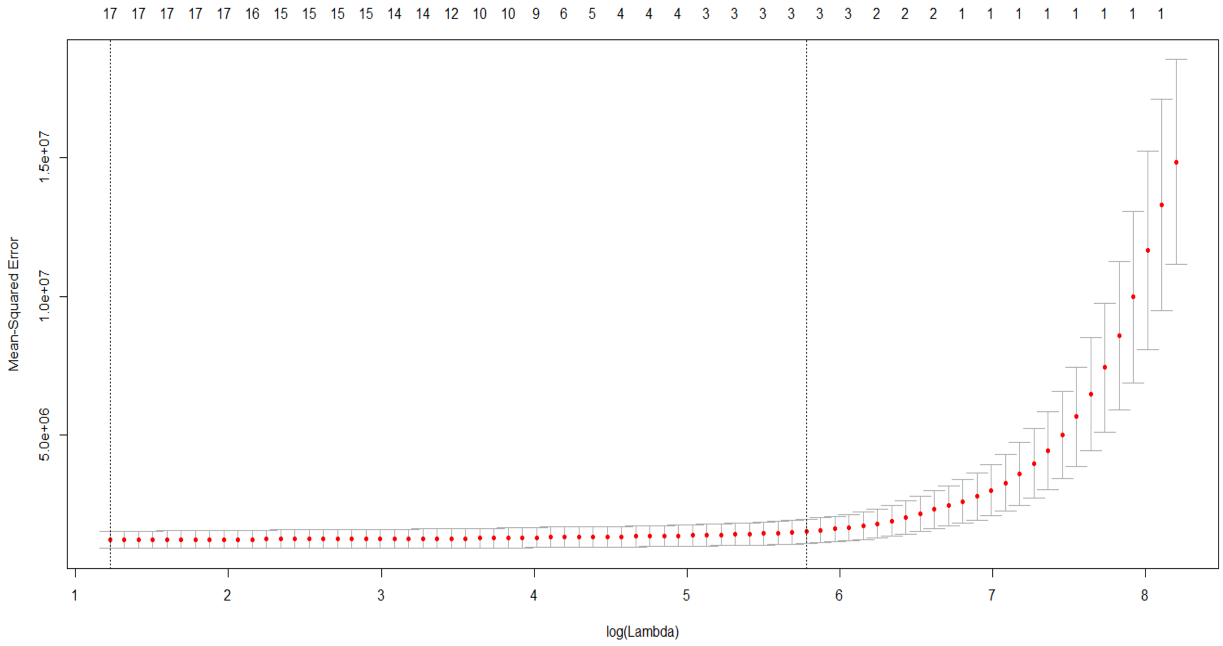
(c).



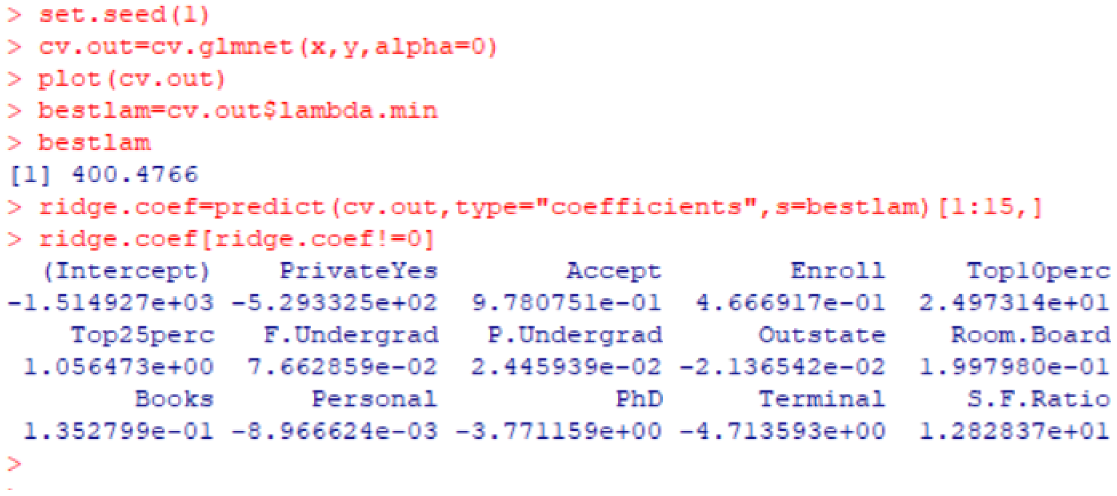
```
> install.packages("glmnet")
Installing package into 'C:/Users/samba/Documents/R/win-library/3.4'
(as 'lib' is unspecified)
also installing the dependencies 'iterators', 'foreach'

trying URL 'https://cran.revolutionanalytics.com/bin/windows/contrib/3.4/iterators_1.0.9.zip'
Content type 'application/zip' length 320532 bytes (313 KB)
downloaded 313 KB

trying URL 'https://cran.revolutionanalytics.com/bin/windows/contrib/3.4/foreach_1.4.4.zip'
Content type 'application/zip' length 388603 bytes (379 KB)
downloaded 379 KB

trying URL 'https://cran.revolutionanalytics.com/bin/windows/contrib/3.4/glmnet_2.0-16.zip'
Content type 'application/zip' length 1739333 bytes (1.7 MB)
downloaded 1.7 MB

package 'iterators' successfully unpacked and MD5 sums checked
package 'foreach' successfully unpacked and MD5 sums checked
package 'glmnet' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\samba\AppData\Local\Temp\RtmpkdwHoZ\downloaded_packages
> library(glmnet)
Loading required package: Matrix
Loading required package: foreach
Loaded glmnet 2.0-16

Warning messages:
1: package 'glmnet' was built under R version 3.4.4
2: package 'foreach' was built under R version 3.4.4
```



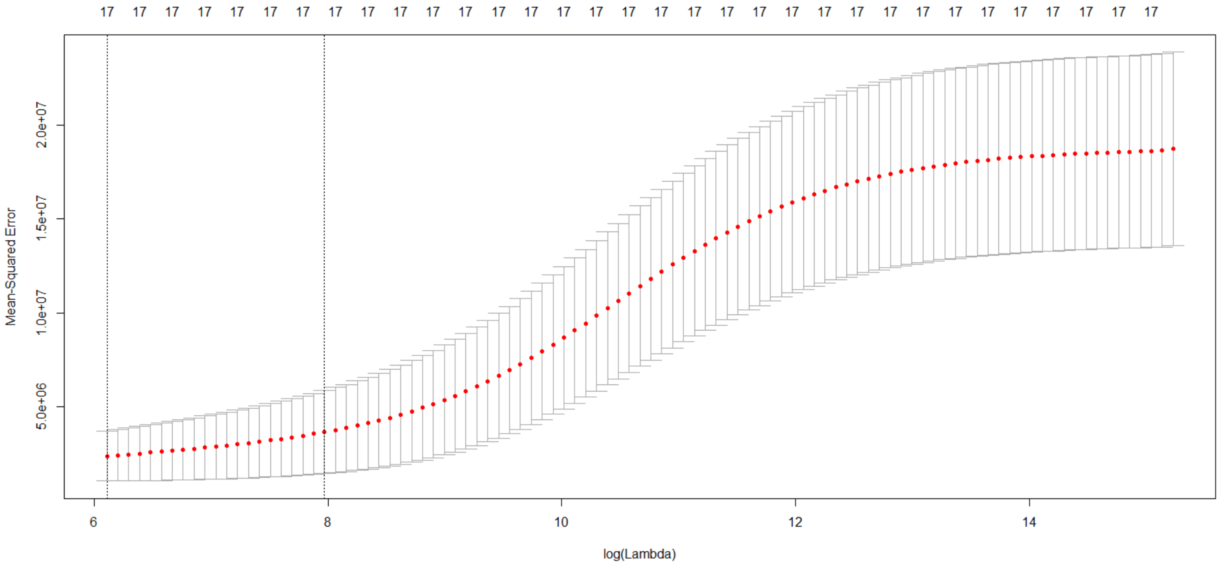
```
> set.seed(1)
> cv.out=cv.glmnet(x,y,alpha=1)
> plot(cv.out)
> bestlam=cv.out$lambda.min
> bestlam
[1] 3.403063
> lasso.coef=predict(cv.out,type="coefficients",s=bestlam)[1:15,]
> lasso.coef[lasso.coef!=0]
  (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
-481.69122788 -489.47698919    1.56285991   -0.69952896   47.20524292 
    Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board 
 -12.12210805    0.03356097    0.04415215   -0.08184648    0.14813763 
        Books      Personal           PhD      Terminal     S.F.Ratio 
   0.01201765    0.02785918   -8.24433269   -3.21033519   14.04536900 
> 
```



(d).

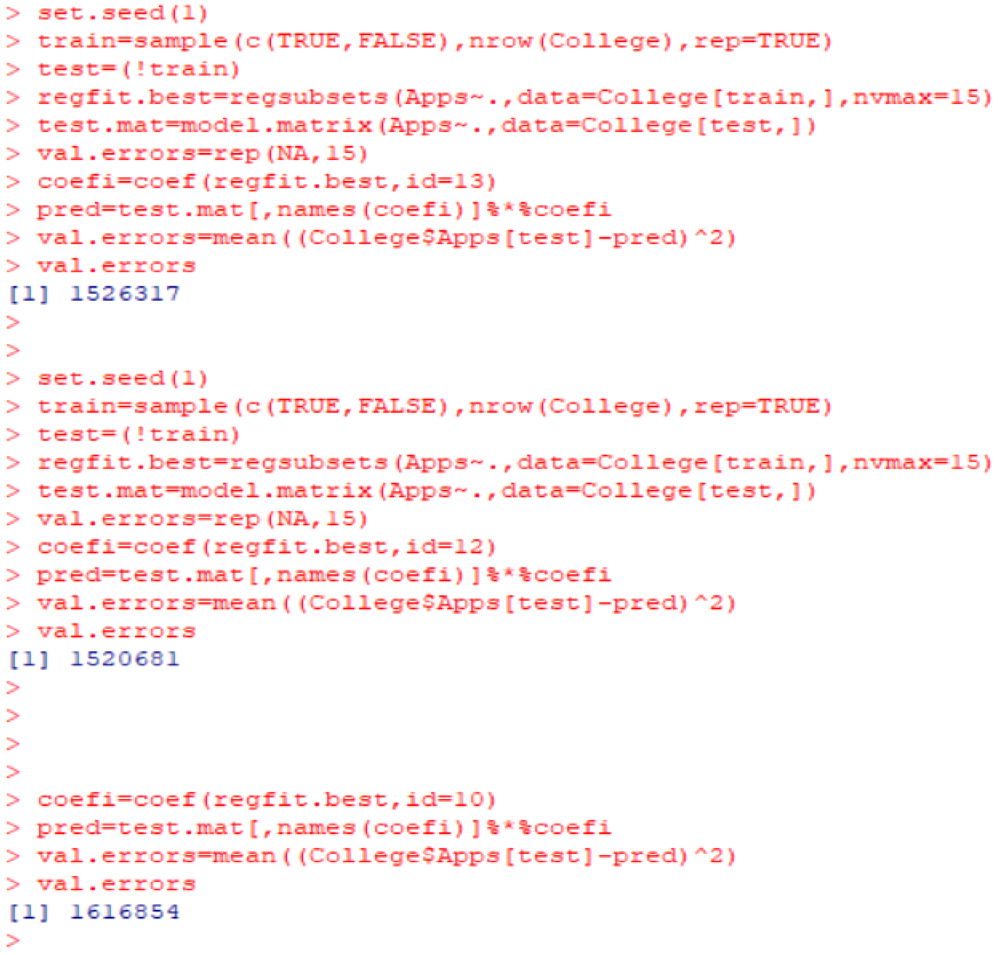


```
> set.seed(1)
> cv.out=cv.glmnet(x,y,alpha=0)
> plot(cv.out)
> bestlam=cv.out$lambda.min
> bestlam
[1] 400.4766
> ridge.coef=predict(cv.out,type="coefficients",s=bestlam)[1:15,]
> ridge.coef[ridge.coef!=0]
  (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
-1.514927e+03 -5.293325e+02  9.780751e-01  4.666917e-01  2.497314e+01 
    Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board 
 1.056473e+00  7.662859e-02  2.445939e-02 -2.136542e-02  1.997980e-01 
        Books      Personal           PhD      Terminal     S.F.Ratio 
 1.352799e-01 -8.966624e-03 -3.771159e+00 -4.713593e+00  1.282837e+01 
>
```



(e).

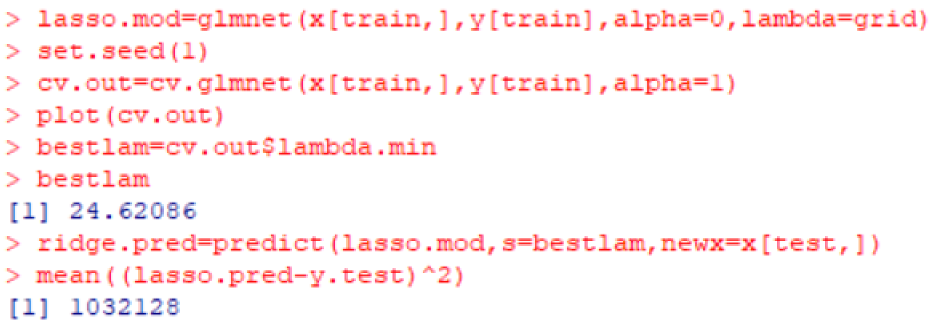
(i).



```
> set.seed(1)
> train=sample(c(TRUE,FALSE),nrow(College),rep=TRUE)
> test=(!train)
> regfit.best=regsubsets(Apps~.,data=College[train,],nvmax=15)
> test.mat=model.matrix(Apps~.,data=College[test,])
> val.errors=rep(NA,15)
> coefi=coef(regfit.best,id=13)
> pred=test.mat[,names(coefi)]%*%coefi
> val.errors=mean((College$Apps[test]-pred)^2)
> val.errors
[1] 1526317
>
>
> set.seed(1)
> train=sample(c(TRUE,FALSE),nrow(College),rep=TRUE)
> test=(!train)
> regfit.best=regsubsets(Apps~.,data=College[train,],nvmax=15)
> test.mat=model.matrix(Apps~.,data=College[test,])
> val.errors=rep(NA,15)
> coefi=coef(regfit.best,id=12)
> pred=test.mat[,names(coefi)]%*%coefi
> val.errors=mean((College$Apps[test]-pred)^2)
> val.errors
[1] 1520681
>
>
>
>
> coefi=coef(regfit.best,id=10)
> pred=test.mat[,names(coefi)]%*%coefi
> val.errors=mean((College$Apps[test]-pred)^2)
> val.errors
[1] 1616854
>
```

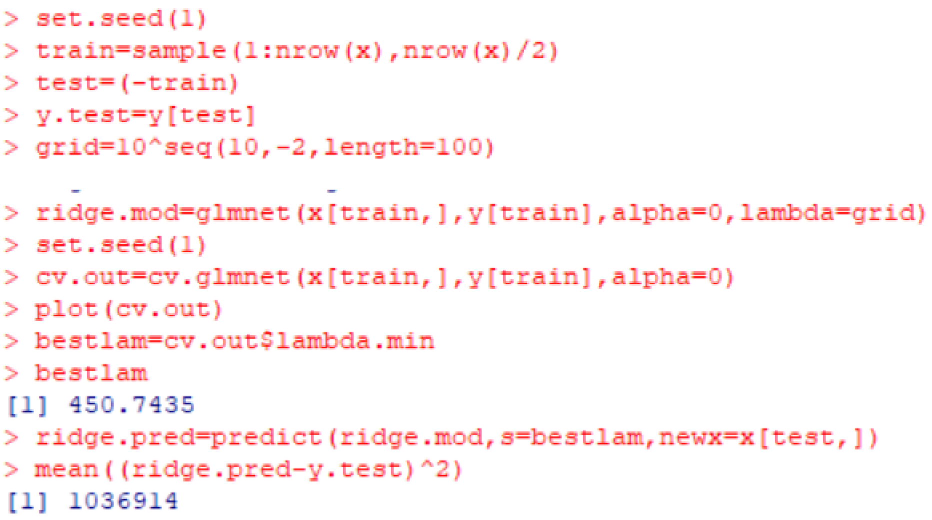
The errors for all the 3 cases after splitting our data into training and testing data have been reported (no of predictors= 13,12 and 10 respectively).

(ii).



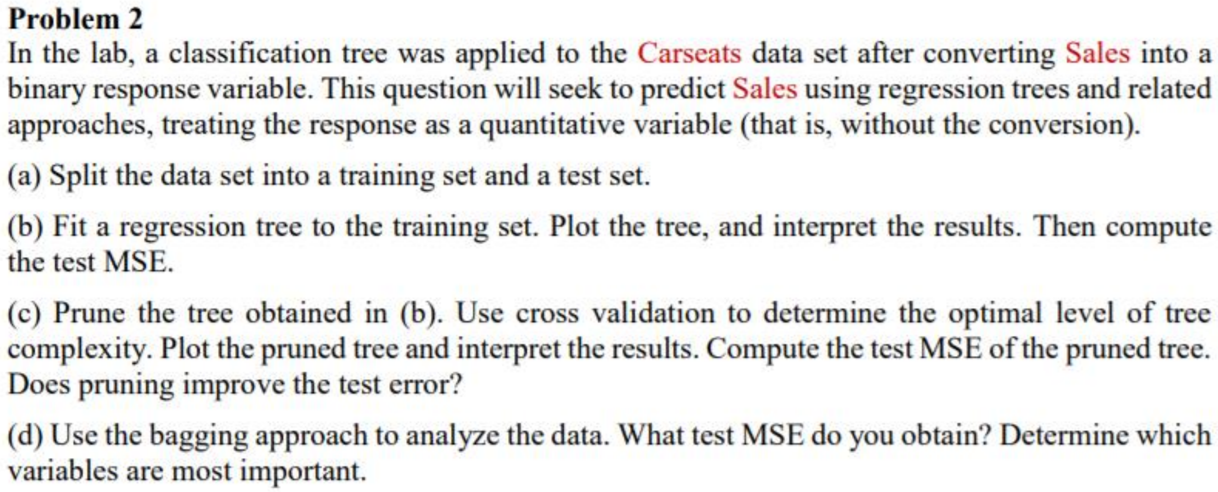
```
> lasso.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid)
> set.seed(1)
> cv.out=cv.glmnet(x[train,],y[train],alpha=1)
> plot(cv.out)
> bestlam=cv.out$lambda.min
> bestlam
[1] 24.62086
> ridge.pred=predict(lasso.mod,s=bestlam,newx=x[test,])
> mean((lasso.pred-y.test)^2)
[1] 1032128
```

(iii).



```
> set.seed(1)
> train=sample(1:nrow(x),nrow(x)/2)
> test=(-train)
> y.test=y[test]
> grid=10^seq(10,-2,length=100)

> ridge.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid)
> set.seed(1)
> cv.out=cv.glmnet(x[train,],y[train],alpha=0)
> plot(cv.out)
> bestlam=cv.out$lambda.min
> bestlam
[1] 450.7435
> ridge.pred=predict(ridge.mod,s=bestlam,newx=x[test,])
> mean((ridge.pred-y.test)^2)
[1] 1036914
```

(iv). For our model, computing the test errors, we can conclude that the LASSO model considering lambda= 24.62 is the best model to fit our model as it gives the lowest error on the test data.



Problem 2

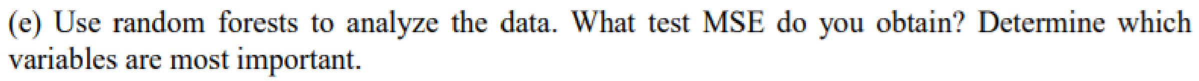
In the lab, a classification tree was applied to the Carseats data set after converting Sales into a binary response variable. This question will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable (that is, without the conversion).

(a) Split the data set into a training set and a test set.

(b) Fit a regression tree to the training set. Plot the tree, and interpret the results. Then compute the test MSE.

(c) Prune the tree obtained in (b). Use cross validation to determine the optimal level of tree complexity. Plot the pruned tree and interpret the results. Compute the test MSE of the pruned tree. Does pruning improve the test error?

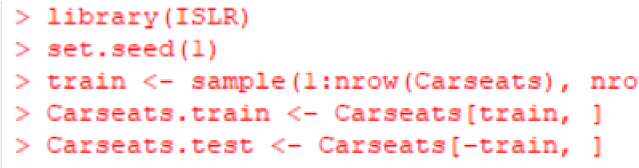
(d) Use the bagging approach to analyze the data. What test MSE do you obtain? Determine which variables are most important.



(e) Use random forests to analyze the data. What test MSE do you obtain? Determine which variables are most important.

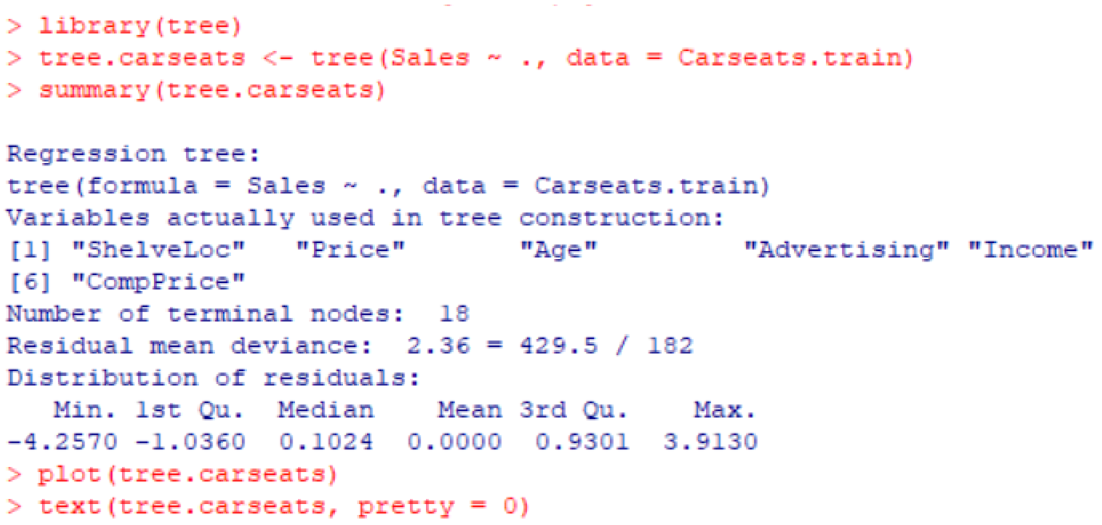
Ans2

(a).

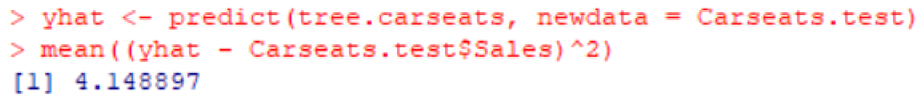


```
> library(ISLR)
> set.seed(1)
> train <- sample(1:nrow(Carseats), nrow(Carseats) / 2)
> Carseats.train <- Carseats[train, ]
> Carseats.test <- Carseats[-train, ]
```

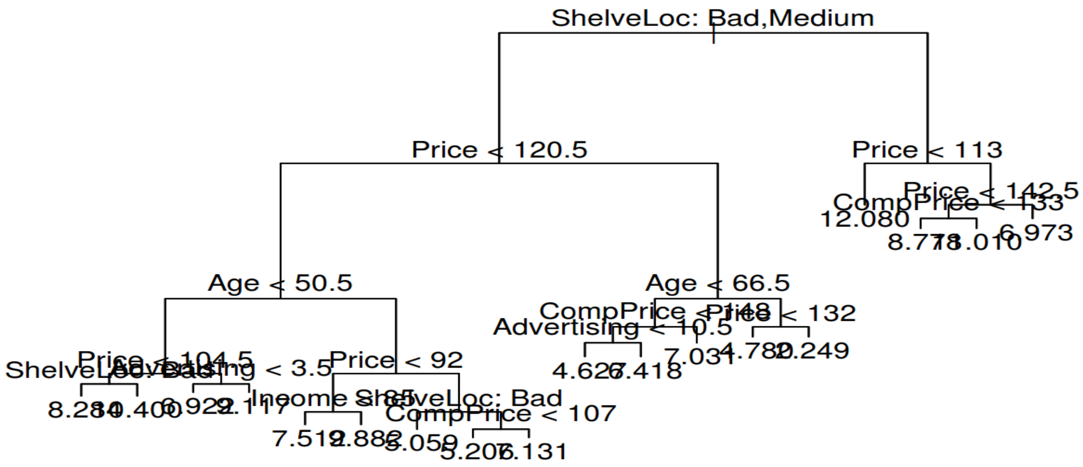
(b).



```
> library(tree)
> tree.carseats <- tree(Sales ~ ., data = Carseats.train)
> summary(tree.carseats)

Regression tree:
tree(formula = Sales ~ ., data = Carseats.train)
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"       "Age"          "Advertising" "Income"
[6] "CompPrice"
Number of terminal nodes:  18
Residual mean deviance:  2.36 = 429.5 / 182
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-4.2570 -1.0360  0.1024  0.0000  0.9301  3.9130
> plot(tree.carseats)
> text(tree.carseats, pretty = 0)
```

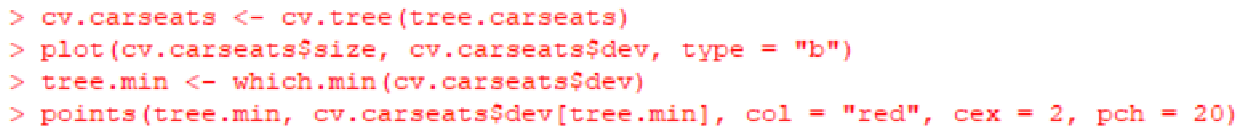


```
> yhat <- predict(tree.carseats, newdata = Carseats.test)
> mean((yhat - Carseats.test$Sales)^2)
[1] 4.148897
```

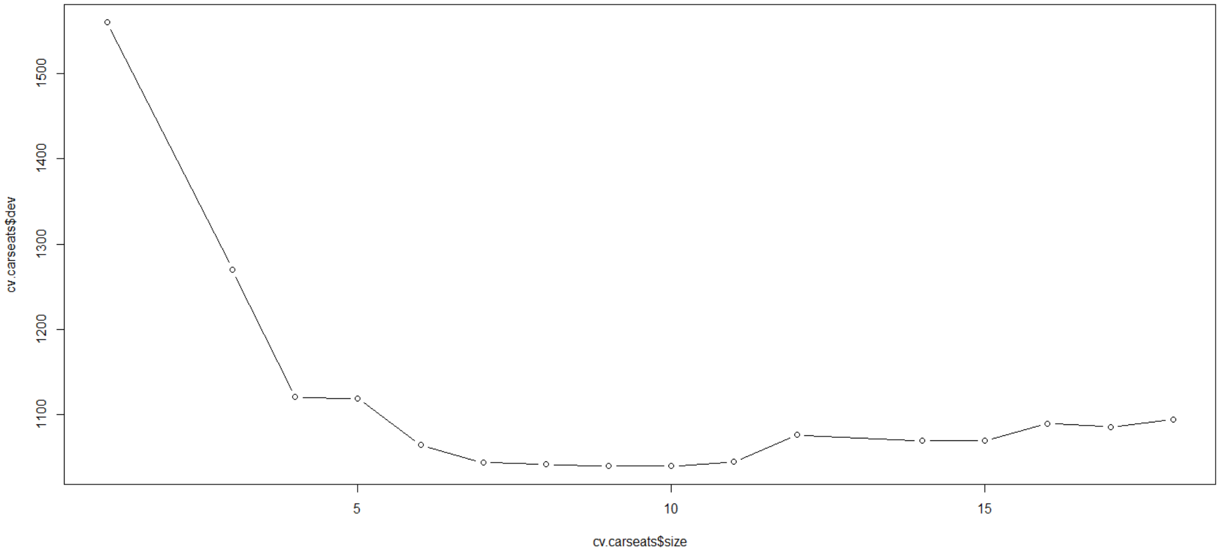


The test MSE in our case was found to be 4.148

(c).

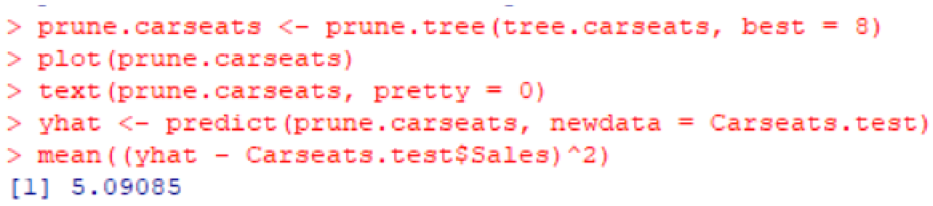


```
> cv.carseats <- cv.tree(tree.carseats)
> plot(cv.carseats$size, cv.carseats$dev, type = "b")
> tree.min <- which.min(cv.carseats$dev)
> points(tree.min, cv.carseats$dev[tree.min], col = "red", cex = 2, pch = 20)
```

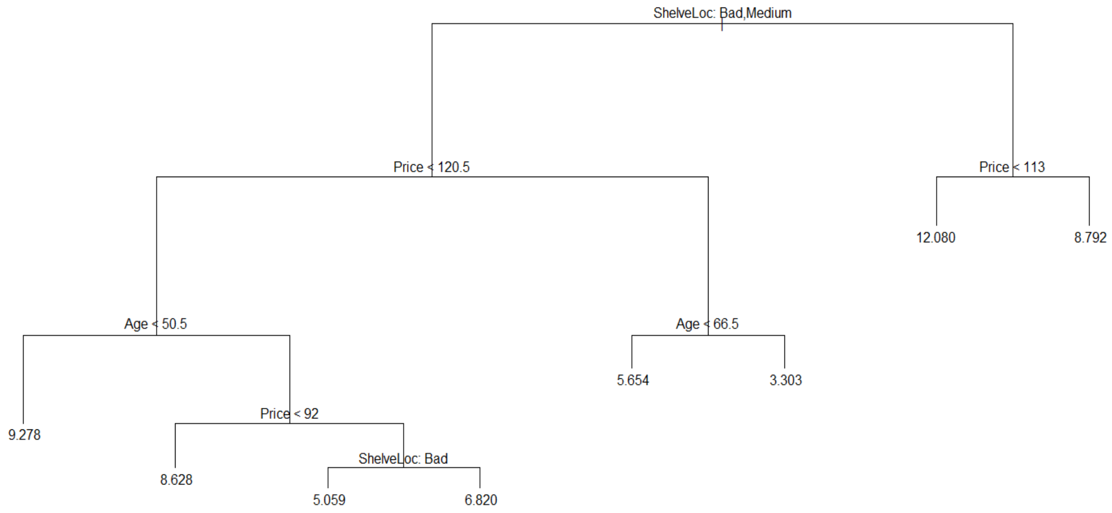


By the cross-validation method used in our approach, we can conclude that we select a tree of size=8.

So, we can prune the tree to a 8-node tree.

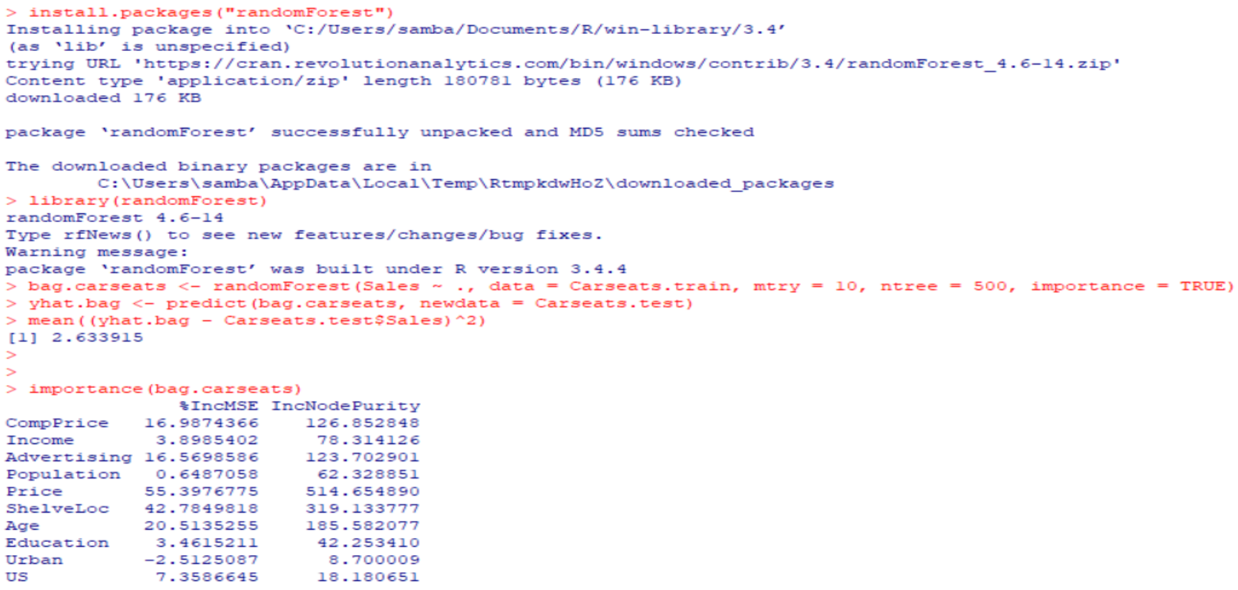


```
> prune.carseats <- prune.tree(tree.carseats, best = 8)
> plot(prune.carseats)
> text(prune.carseats, pretty = 0)
> yhat <- predict(prune.carseats, newdata = Carseats.test)
> mean((yhat - Carseats.test$Sales)^2)
[1] 5.09085
```



We can conclude that by pruning our tree, the test MSE further detoriates to 5.1 compared to 4.14 which we had in the previous case.

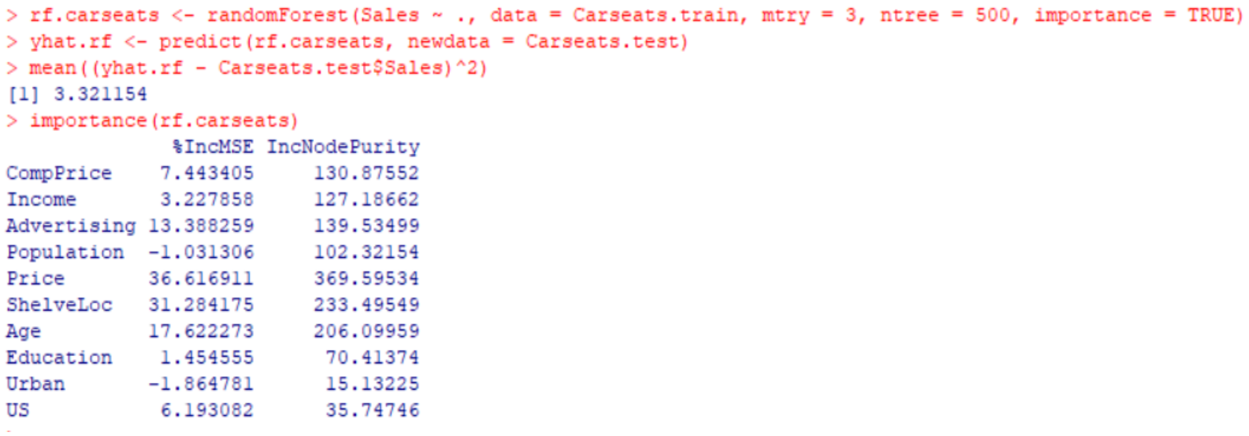
(d).



```
> install.packages("randomForest")
Installing package into 'C:/Users/samba/Documents/R/win-library/3.4'
(as 'lib' is unspecified)
trying URL 'https://cran.revolutionanalytics.com/bin/windows/contrib/3.4/randomForest_4.6-14.zip'
Content type 'application/zip' length 180781 bytes (176 KB)
downloaded 176 KB

package 'randomForest' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\samba\AppData\Local\Temp\RtmpkdwHoZ\downloaded_packages
> library(randomForest)
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
Warning message:
package 'randomForest' was built under R version 3.4.4
> bag.carseats <- randomForest(Sales ~ ., data = Carseats.train, mtry = 10, ntree = 500, importance = TRUE)
> yhat.bag <- predict(bag.carseats, newdata = Carseats.test)
> mean((yhat.bag - Carseats.test$Sales)^2)
[1] 2.633915
>
>
> importance(bag.carseats)
               %IncMSE IncNodePurity
CompPrice   16.9874366    126.852848
Income       3.8985402     78.314126
Advertising 16.5698586    123.702901
Population   0.6487058     62.328851
Price       55.3976775    514.654890
ShelveLoc   42.7849818    319.133777
Age         20.5135255    185.582077
Education    3.4615211     42.253410
Urban       -2.5125087      8.700009
US           7.3586645     18.180651
```

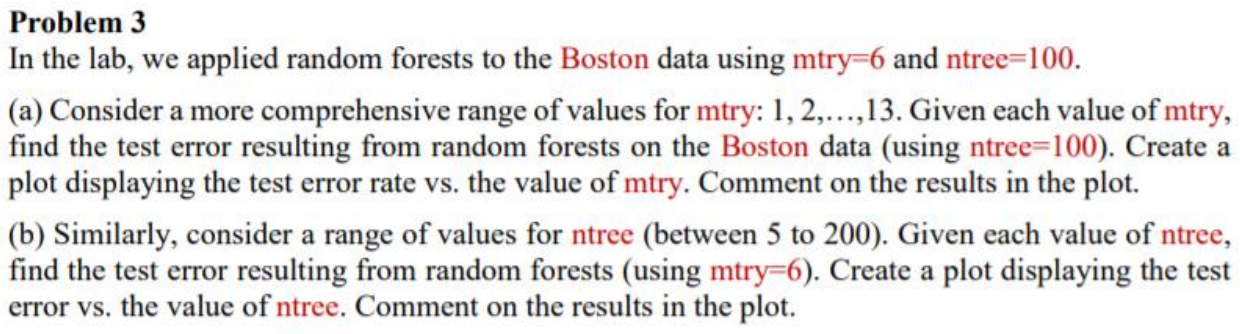
By calculating the importance parameters of the dataset, we can therefore conclude that Price and ShelveLoc are the most important predictors in the dataset since their value is the highest. (using bagging approach)

(e).



```
> rf.carseats <- randomForest(Sales ~ ., data = Carseats.train, mtry = 3, ntree = 500, importance = TRUE)
> yhat.rf <- predict(rf.carseats, newdata = Carseats.test)
> mean((yhat.rf - Carseats.test$Sales)^2)
[1] 3.321154
> importance(rf.carseats)
              %IncMSE IncNodePurity
CompPrice    7.443405     130.87552
Income       3.227858     127.18662
Advertising 13.388259     139.53499
Population  -1.031306     102.32154
Price       36.616911     369.59534
ShelveLoc   31.284175     233.49549
Age         17.622273     206.09959
Education    1.454555      70.41374
Urban       -1.864781      15.13225
US           6.193082      35.74746
```

Here, also when we calculate the importance parameters, even after using RandomForests, we get the same result. Price and ShelveLoc remain the most important predictors of the regression tree.



Problem 3

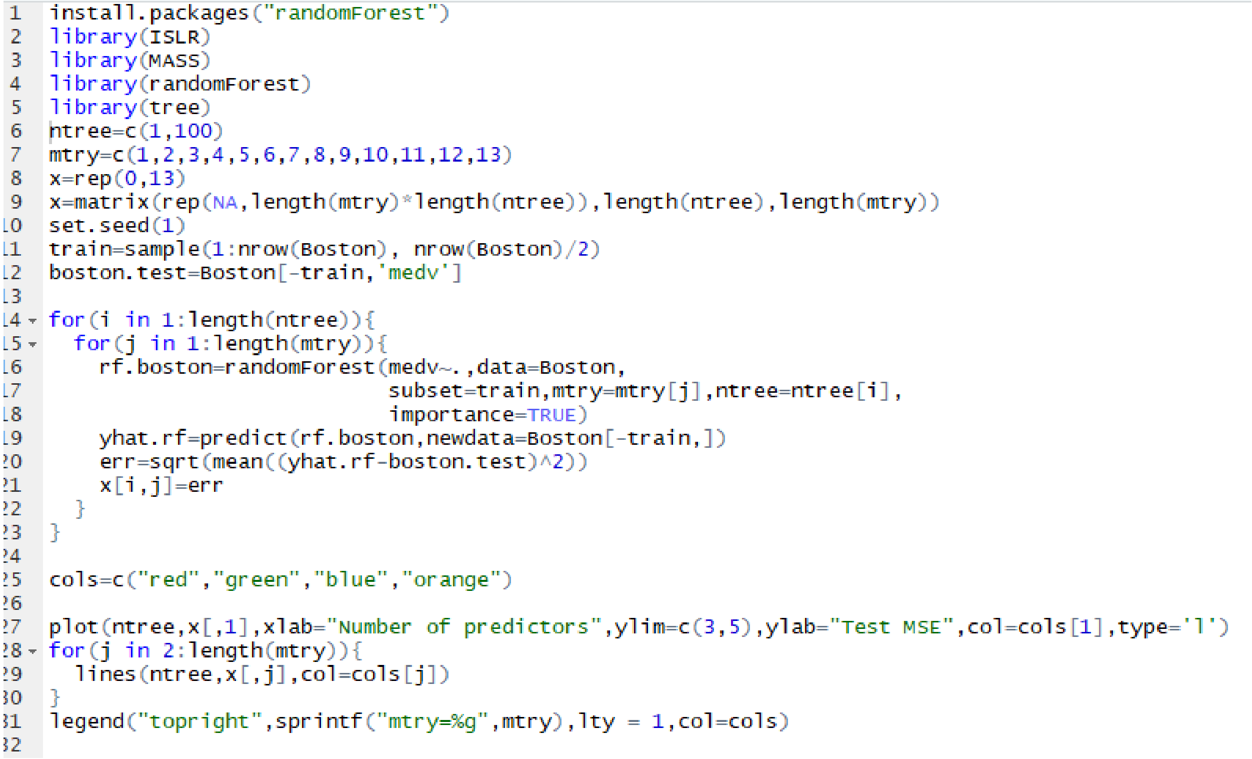
In the lab, we applied random forests to the Boston data using mtry=6 and ntree=100.

(a) Consider a more comprehensive range of values for mtry: 1, 2,…,13. Given each value of mtry, find the test error resulting from random forests on the Boston data (using ntree=100). Create a plot displaying the test error rate vs. the value of mtry. Comment on the results in the plot.

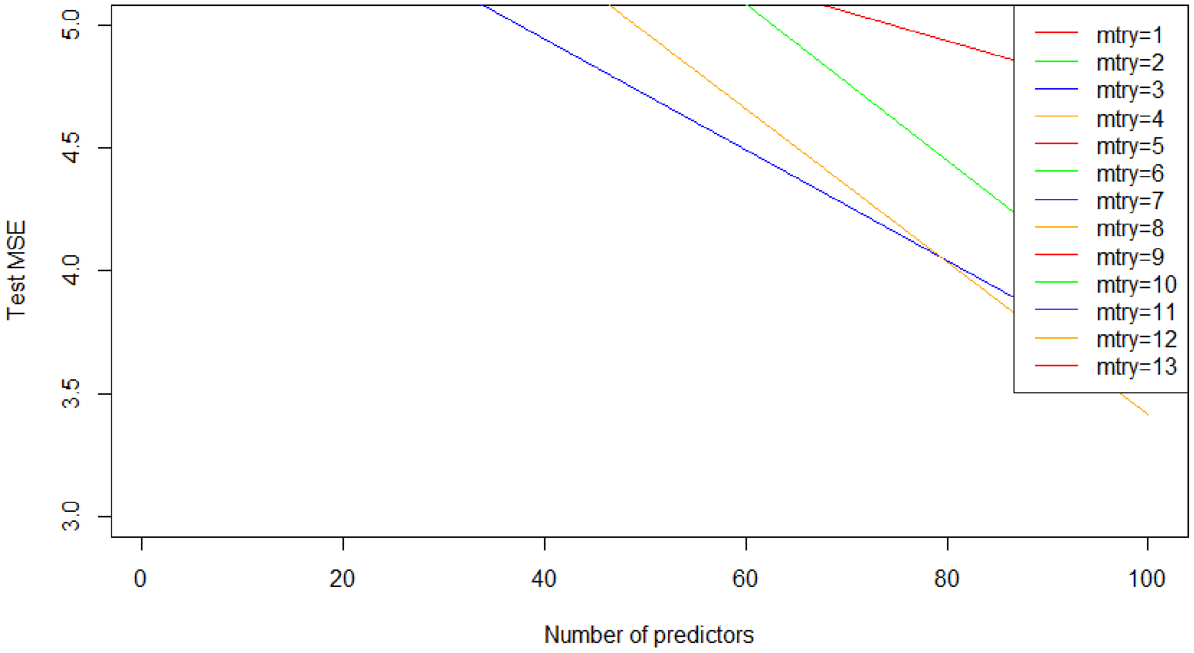
(b) Similarly, consider a range of values for ntree (between 5 to 200). Given each value of ntree, find the test error resulting from random forests (using mtry=6). Create a plot displaying the test error vs. the value of ntree. Comment on the results in the plot.

Ans 3.

(a).

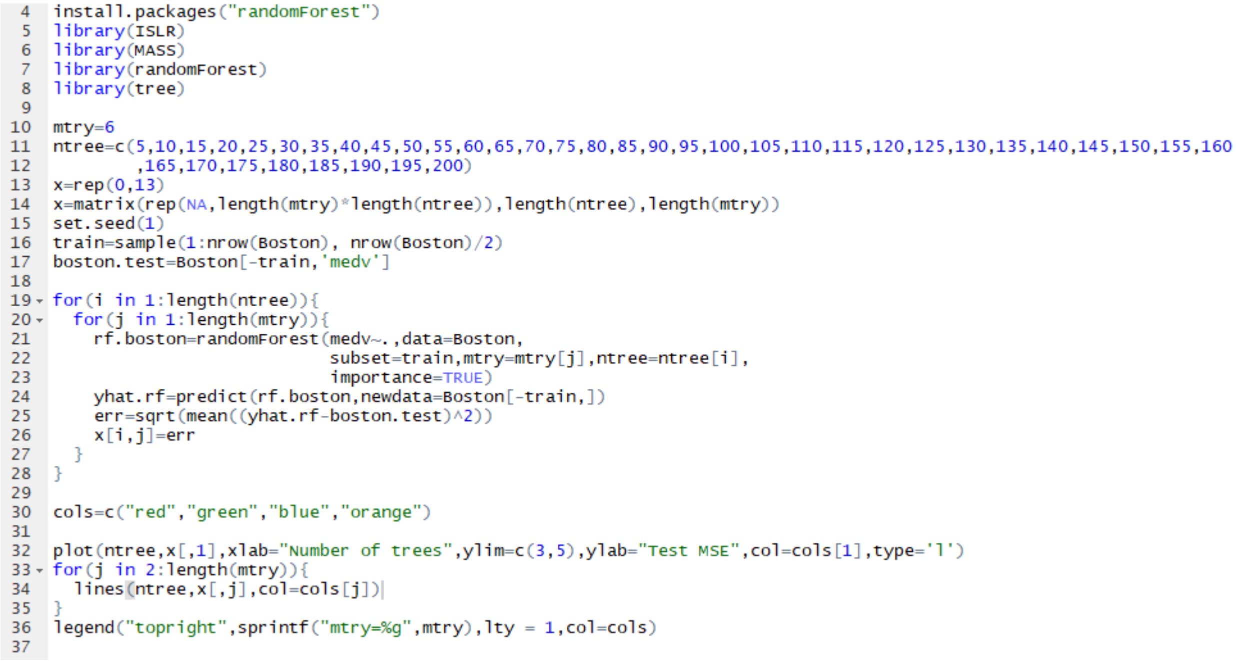


```
install.packages("randomForest")
library(ISLR)
library(MASS)
library(randomForest)
library(tree)
ntree=c(1,100)
mtry=c(1,2,3,4,5,6,7,8,9,10,11,12,13)
x=rep(0,13)
x=matrix(rep(NA,length(mtry)*length(ntree)),length(ntree),length(mtry))
set.seed(1)
train=sample(1:nrow(Boston), nrow(Boston)/2)
boston.test=Boston[-train,'medv']

for(i in 1:length(ntree)){
  for(j in 1:length(mtry)){
    rf.boston=randomForest(medv~.,data=Boston,
                           subset=train,mtry=mtry[j],ntree=ntree[i],
                           importance=TRUE)
    yhat.rf=predict(rf.boston,newdata=Boston[-train,])
    err=sqrt(mean((yhat.rf-boston.test)^2))
    x[i,j]=err
  }
}

cols=c("red","green","blue","orange")

plot(ntree,x[,1],xlab="Number of predictors",ylim=c(3,5),ylab="Test MSE",col=cols[1],type='l')
for(j in 2:length(mtry)){
  lines(ntree,x[,j],col=cols[j])
}
legend("topright",sprintf("mtry=%g",mtry),lty = 1,col=cols)
```

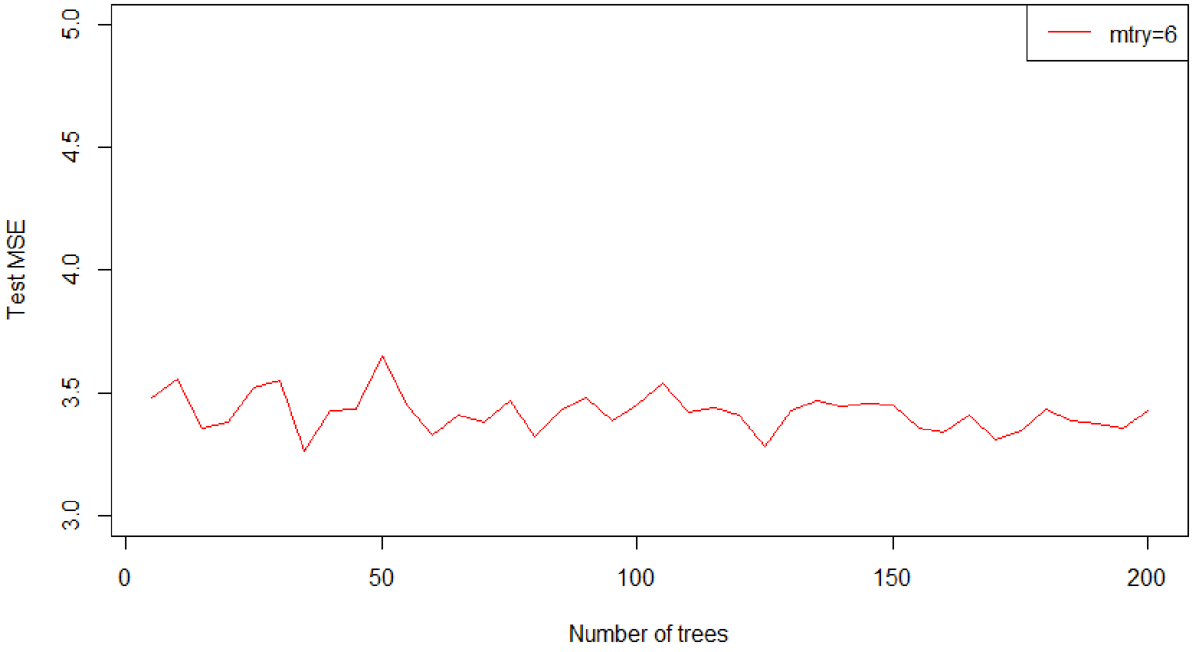


From the graph, we can observe that when the number of predictors go on to increase, the test MSE decreases. Test MSE reaches the lowest when the number of predictors is 12 and it is observed that the test MSE is maximum when the number of predictors is 1 keeping the no of decision trees constant=100.

(b).



```
install.packages("randomForest")
library(ISLR)
library(MASS)
library(randomForest)
library(tree)

mtry=6
ntree=c(5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100,105,110,115,120,125,130,135,140,145,150,155,160
        ,165,170,175,180,185,190,195,200)
x=rep(0,13)
x=matrix(rep(NA,length(mtry)*length(ntree)),length(ntree),length(mtry))
set.seed(1)
train=sample(1:nrow(Boston), nrow(Boston)/2)
boston.test=Boston[-train,'medv']

for(i in 1:length(ntree)){
  for(j in 1:length(mtry)){
    rf.boston=randomForest(medv~.,data=Boston,
                           subset=train,mtry=mtry[j],ntree=ntree[i],
                           importance=TRUE)
    yhat.rf=predict(rf.boston,newdata=Boston[-train,])
    err=sqrt(mean((yhat.rf-boston.test)^2))
    x[i,j]=err
  }
}

cols=c("red","green","blue","orange")

plot(ntree,x[,1],xlab="Number of trees",ylim=c(3,5),ylab="Test MSE",col=cols[1],type='l')
for(j in 2:length(mtry)){
  lines(ntree,x[,j],col=cols[j])
}
legend("topright",sprintf("mtry=%g",mtry),lty = 1,col=cols)
```



Keeping mtry=6, when the number of trees increase from 5 to 200, test MSE is maximum when the number of trees is 50 which is approximately 3.75 and test MSE is minimum when the number of trees considered is 125 which is approximately 3.2