# Assignment 5

*Mayank Jaggi*

*April 12, 2018*

## Problem 1

### a)

```
library(ISLR)
fix(College)
sum(is.na(College)) # to determine missing elements
```

```
## [1] 0
```

```
library (leaps)
```

```
## Warning: package 'leaps' was built under R version 3.4.4
```

```
regit.full=regsubsets(Apps~., College,nvmax=19)
reg.summary=summary(regit.full)
reg.summary
```

```
## Subset selection object
## Call: regsubsets.formula(Apps ~ ., College, nvmax = 19)
## 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: exhaustive
##           PrivateYes Accept Enroll Top10perc Top25perc F.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 ) "*"        "*"    "*"    "*"       "*"       "*"
##             P.Undergrad Outstate Room.Board Books Personal PhD Terminal
## 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 ) "*"         "*"      "*"        "*"   "*"      "*" "*"
##             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 ) "*"       "*"         "*"    "*"
```

```r
par(mfrow=c(2,2))

plot(reg.summary$rss,xlab="Number of Variables",ylab="RSS",type = "l",col="red")
b=which.min(reg.summary$rss)
points(b,reg.summary$rss[b],col="black", pch=20)
```

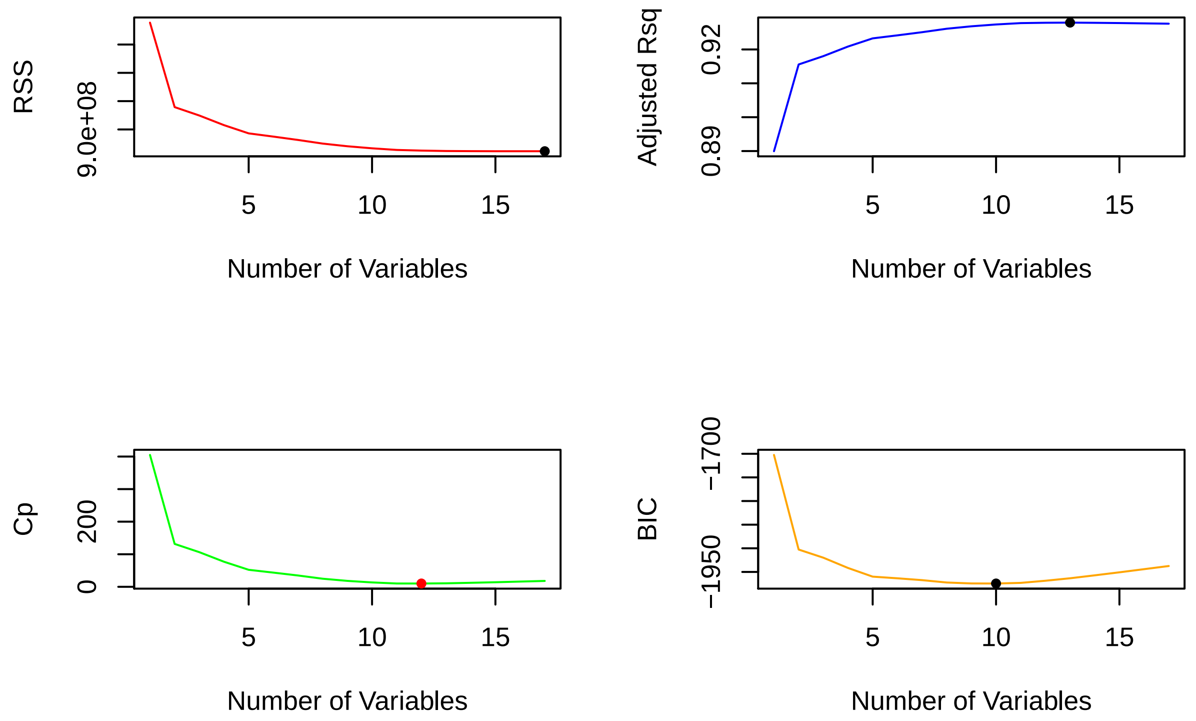
```
plot(reg.summary$adjr2,xlab="Number of Variables",ylab="Adjusted Rsq",type = "l",col="blue")
a=which.max(reg.summary$adjr2)
points(a,reg.summary$adjr2[a],col="black", pch=20)
print(a)
```

```
## [1] 13
```

```
plot(reg.summary$cp,xlab="Number of Variables",ylab="Cp",type = "l",col="green")
c=which.min(reg.summary$cp)
points(c,reg.summary$cp[c],col="red", pch=20)
print(c)
```

```
## [1] 12
```

```
plot(reg.summary$bic,xlab="Number of Variables",ylab="BIC",type = "l",col="orange")
d=which.min(reg.summary$bic)
points(d,reg.summary$bic[d],col="black", pch=20)
```



```
print(d)
```

```
## [1] 10
```

```
coef(regit.full,a)     #coeffecients of Adjusted R^2
```

```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
## -440.74148270 -484.77261885    1.58542302   -0.87824288   50.41461998 
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board 
##  -14.63667155    0.05762769    0.04642270   -0.08823311    0.14696204 
##           PhD     S.F.Ratio        Expend     Grad.Rate 
##  -10.91804823   15.15475056    0.07786425    8.58578735 
```

```
coef(regit.full,c)    #coeffecients of Cp
```

```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
## -157.28685883 -511.78760196    1.58691470   -0.88265385   50.41131660 
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board 
##  -14.74735373    0.05945481    0.04593068   -0.09017643    0.14776586 
##           PhD        Expend     Grad.Rate 
##  -10.70502848    0.07246655    8.63961002 
```

```
coef(regit.full,d)    #coeffecients of BIC
```

```
##   (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 
```

As per the above analysis for:

1. Cp 12 variables are selected which include PrivateYes, Accept, Enroll, Top10perc, Top25perc, F.Undergrad, P.Undergrad, Outstate, Room.Board, PhD, Expend, Grad.Rate
2. BIC 10 variables are selected which include PrivateYes, Accept, Enroll, Top10perc, Top25perc, Outstate, Room.Board, PhD, Expend, Grad.Rate
3. Adj R^2 13 variables are selected which include PrivateYes, Accept, Enroll, Top10perc, Top25perc, F.Undergrad, P.Undergrad, Outstate, Room.Board, PhD, S.F.Ratio, Expend, Grad.Rate

## b) Forward Stepwise Selection

```
regit.fwd=regsubsets(Apps~., College,nvmax=19, method = "forward")
reg.summary.fwd=summary(regit.fwd)

par(mfrow=c(2,2))
plot(reg.summary.fwd$adjr2,xlab="Number of Variables",ylab="Adjusted Rsq",type = "l",col="blue")
a=which.max(reg.summary.fwd$adjr2)
points(a,reg.summary.fwd$adjr2[a],col="black", pch=20)
print(a)
```

```
## [1] 13
```

```
plot(reg.summary.fwd$cp,xlab="Number of Variables",ylab="Cp",type = "l",col="green")
c=which.min(reg.summary.fwd$cp)
points(c,reg.summary.fwd$cp[c],col="red", pch=20)
print(c)
```

```
## [1] 12
```

```
plot(reg.summary.fwd$bic,xlab="Number of Variables",ylab="BIC",type = "l",col="orange")
d=which.min(reg.summary.fwd$bic)
points(d,reg.summary.fwd$bic[d],col="black", pch=20)
print(d)
```

```
## [1] 10
```

```
coef(regit.full,a)     #coeffecients of Adjusted R^2
```

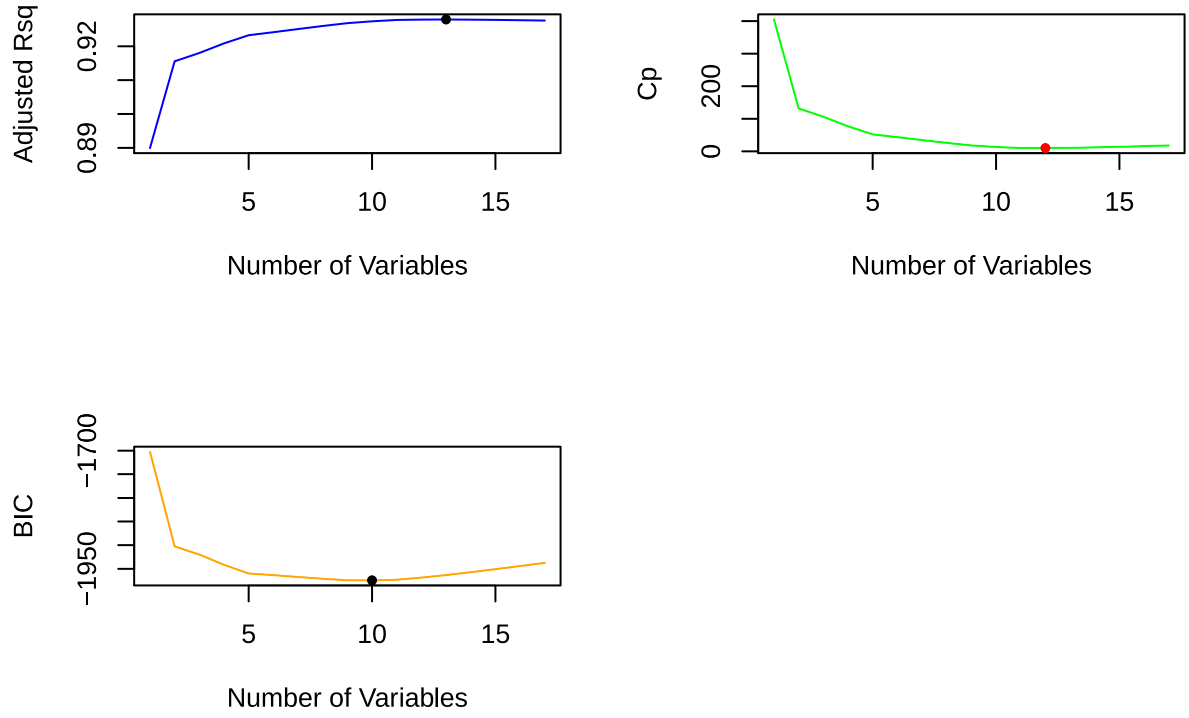
```
##    (Intercept)     PrivateYes         Accept         Enroll      Top10perc
## -440.74148270 -484.77261885     1.58542302    -0.87824288    50.41461998
##      Top25perc     F.Undergrad    P.Undergrad       Outstate     Room.Board
##   -14.63667155     0.05762769     0.04642270    -0.08823311     0.14696204
##            PhD      S.F.Ratio         Expend      Grad.Rate
##   -10.91804823    15.15475056     0.07786425     8.58578735
```

```
coef(regit.full,c)     #coeffecients of Cp
```

```
##    (Intercept)     PrivateYes         Accept         Enroll      Top10perc
## -157.28685883 -511.78760196     1.58691470    -0.88265385    50.41131660
##      Top25perc     F.Undergrad    P.Undergrad       Outstate     Room.Board
##   -14.74735373     0.05945481     0.04593068    -0.09017643     0.14776586
##            PhD          Expend      Grad.Rate
##   -10.70502848     0.07246655     8.63961002
```

```
coef(regit.full,d)     #coeffecients of BIC
```

```
##    (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
```



## b) Backward stepwise Selection

```r
regit.bwd=regsubsets(Apps~., College,nvmax=19, method="backward")
reg.summary.bwd=summary(regit.bwd)

par(mfrow=c(2,2))
plot(reg.summary.bwd$adjr2,xlab="Number of Variables",ylab="Adjusted Rsq",type = "l",col="blue")
a=which.max(reg.summary.bwd$adjr2)
points(a,reg.summary.bwd$adjr2[a],col="black", pch=20)
print(a)
```

```
## [1] 13
```

```r
plot(reg.summary.bwd$cp,xlab="Number of Variables",ylab="Cp",type = "l",col="green")
c=which.min(reg.summary.bwd$cp)
points(c,reg.summary.bwd$cp[c],col="red", pch=20)
print(c)
```

```
## [1] 12
```

```r
plot(reg.summary.bwd$bic,xlab="Number of Variables",ylab="BIC",type = "l",col="orange")
d=which.min(reg.summary.bwd$bic)
points(d,reg.summary.bwd$bic[d],col="black", pch=20)
print(d)
```

```
## [1] 10
```

```r
coef(regit.full,a)    #coeffecients of Adjusted R^2
```

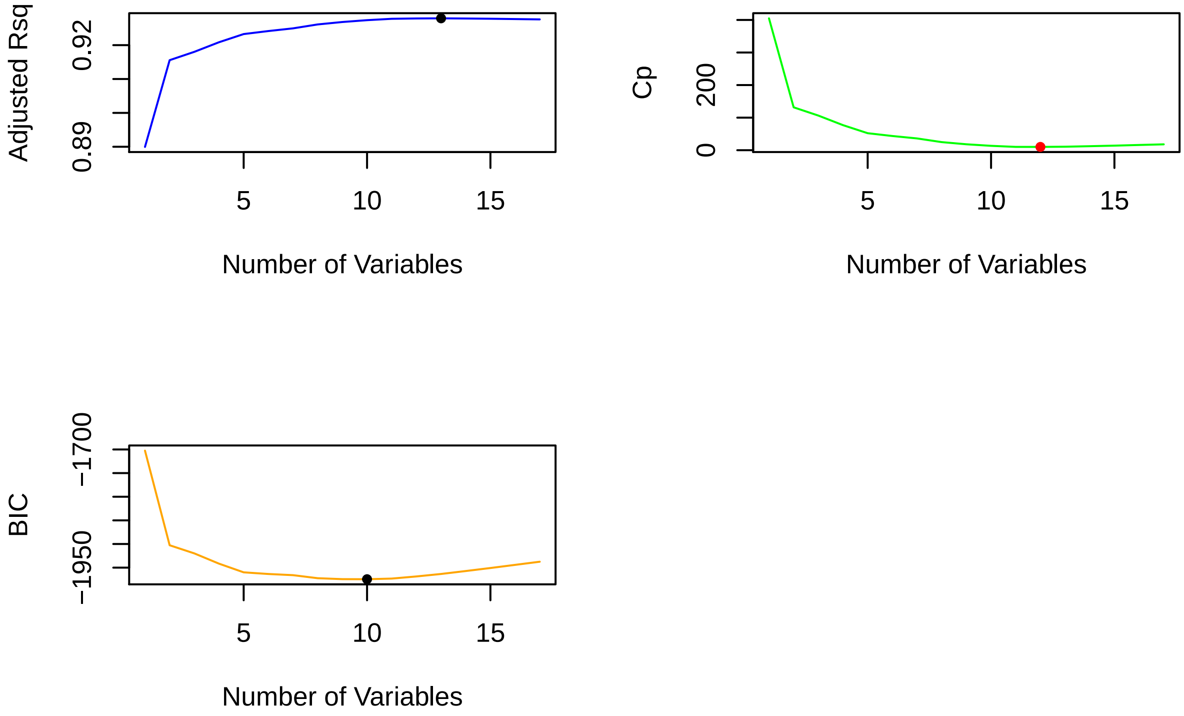
```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
## -440.74148270 -484.77261885    1.58542302   -0.87824288   50.41461998 
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board 
##  -14.63667155    0.05762769    0.04642270   -0.08823311    0.14696204 
##           PhD     S.F.Ratio        Expend     Grad.Rate 
##  -10.91804823   15.15475056    0.07786425    8.58578735
```

```r
coef(regit.full,c)    #coeffecients of Cp
```

```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc 
## -157.28685883 -511.78760196    1.58691470   -0.88265385   50.41131660 
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board 
##  -14.74735373    0.05945481    0.04593068   -0.09017643    0.14776586 
##           PhD        Expend     Grad.Rate 
##  -10.70502848    0.07246655    8.63961002
```

```r
coef(regit.full,d)    #coeffecients of BIC
```

```
##   (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
```



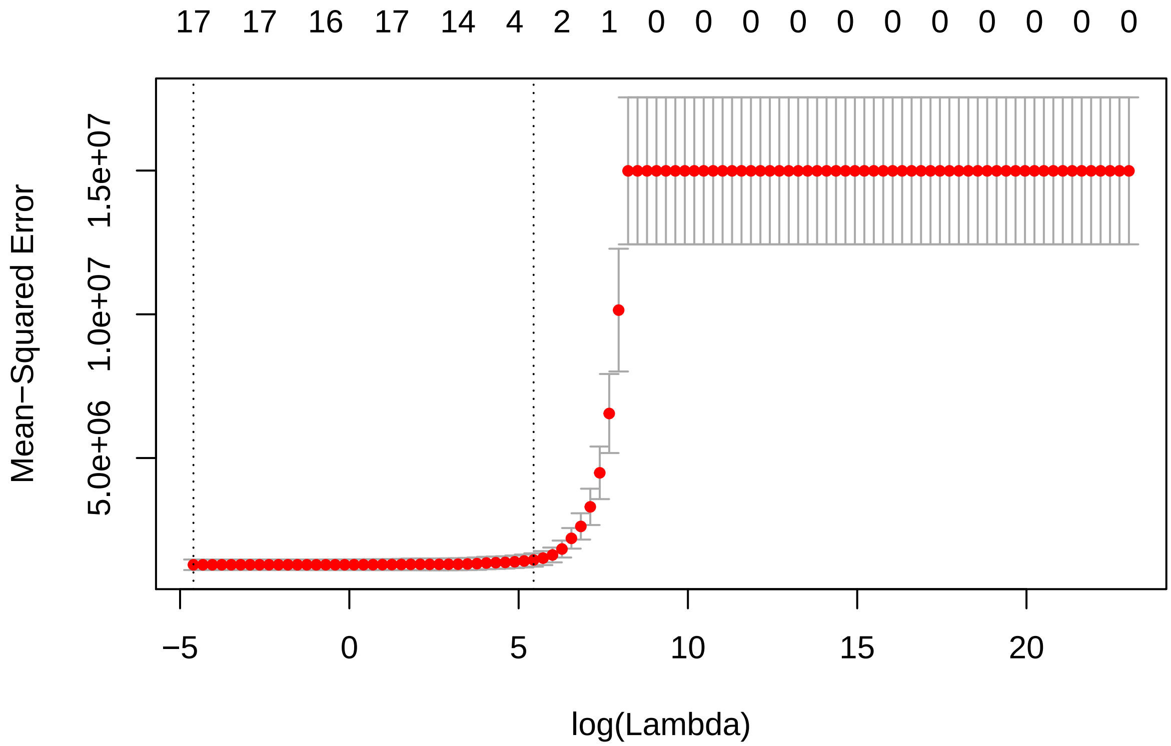
The answers for forward and backward stepwise selection and for a) part that is best subset selection are exactly same.

### c)

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.4

## Loading required package: Matrix

## Loading required package: foreach

## Warning: package 'foreach' was built under R version 3.4.4

## Loaded glmnet 2.0-16
```

```
x=model.matrix(Apps~.,College)[,-1]
y=College$Apps
grid=10^seq(10,-2,length=100)

cv.out=cv.glmnet(x,y,alpha=1,lambda = grid)
plot(cv.out)
```



```r
bestlam=cv.out$lambda.min
bestlam
```

```
## [1] 0.01
```

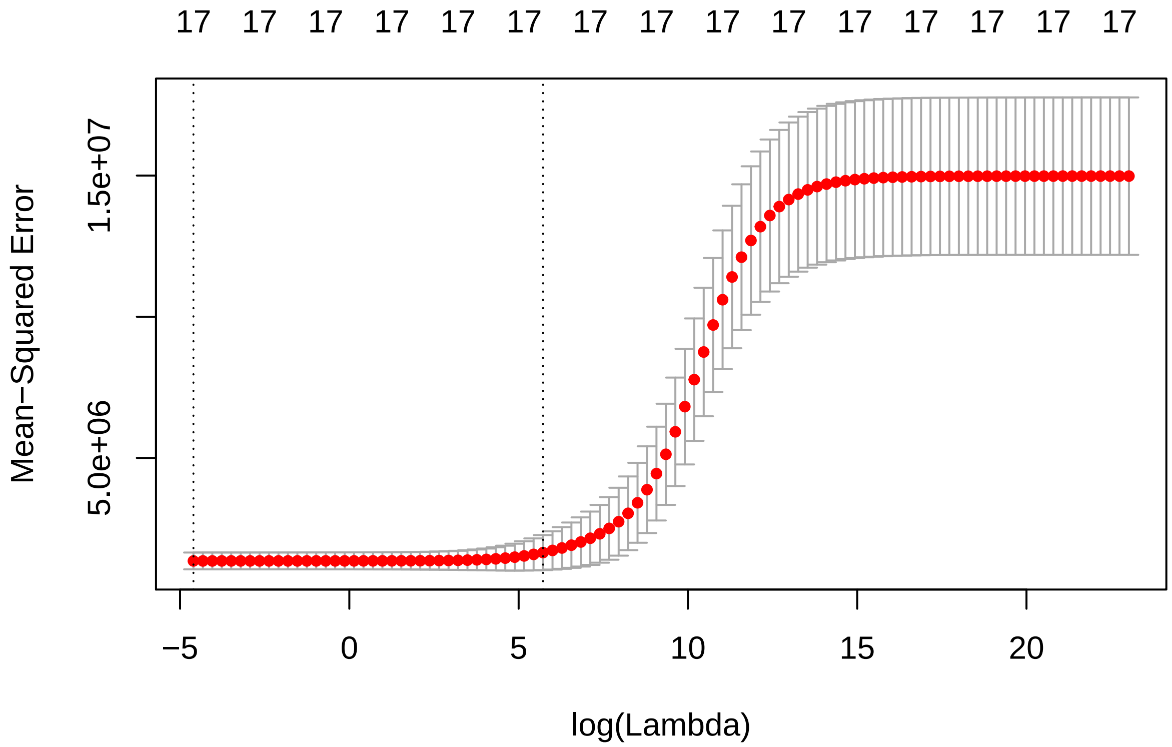
```r
lasso.mod=glmnet(x,College$Apps,alpha=1)
predict(lasso.mod, s=bestlam,type = "coefficients")[1:18,]
```

```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc
## -471.39372069 -491.04485135    1.57033288   -0.75961467   48.14698891
##     Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board
##  -12.84690694    0.04149116    0.04438973   -0.08328388    0.14943472
##         Books      Personal           PhD      Terminal     S.F.Ratio
##    0.01532293    0.02909954   -8.39597537   -3.26800340   14.59298267
##   perc.alumni        Expend     Grad.Rate
##   -0.04404771    0.07712632    8.28950241
```

Lambda=0.3764936

## d)

```r
cv.out=cv.glmnet(x,y,alpha=0,lambda = grid)
plot(cv.out)
```



```
bestlam=cv.out$lambda.min
bestlam
```

```
## [1] 0.01
```

```
##refit using best lamdbda
lasso.mod=glmnet(x,College$Apps,alpha=0)
predict(lasso.mod, s=bestlam,type = "coefficients")[1:18,]
```

```
##    (Intercept)    PrivateYes        Accept        Enroll     Top10perc
## -1.468326e+03 -5.278781e+02  1.004588e+00  4.313442e-01  2.580619e+01
##      Top25perc   F.Undergrad   P.Undergrad      Outstate    Room.Board
##  5.501092e-01  7.258520e-02  2.420595e-02 -2.407454e-02  1.987732e-01
##          Books      Personal           PhD      Terminal     S.F.Ratio
##  1.285477e-01 -8.146131e-03 -4.028284e+00 -4.811071e+00  1.302180e+01
##    perc.alumni        Expend     Grad.Rate
## -8.544783e+00  7.589013e-02  1.126699e+01
```

lambda=0.01

### e)

#### i)

```
set.seed(1)
attach(College)
```

```
x=model.matrix(Apps~.,College)[,-1]
y=College$Apps
train=sample(1:nrow(x),nrow(x)/2)
test=(-train)
y.test=College[test,]
x.train=College[train,]

regit.full=regsubsets(Apps~., data=x.train,nvmax=19)
reg.summary=summary(regit.full)
coef(regit.full,which.max(reg.summary$adjr2))
```

```
##   (Intercept)    PrivateYes        Accept        Enroll     Top10perc
##   84.95670099 -691.04103152    1.67873705   -0.86164941   66.92631417
##     Top25perc      Outstate    Room.Board           PhD        Expend
##  -22.35416377   -0.09482472    0.24520032  -10.14399113    0.03783190
##     Grad.Rate
##    6.45828153
```

```
a=which.max(reg.summary$adjr2)
a
```

```
## [1] 10
```

```
glm.fit=glm(Apps~Private+Accept+Enroll+Top10perc+Top25perc+Outstate+Room.Board+PhD+Expend+Grad.Rate,data
pred.glm=predict(glm.fit,y.test)
mean((pred.glm-y.test$Apps)^2)
```

```
## [1] 1078371
```

The best model contains 10 predictors. Test Error=1078371

### ii)Fit a lasso

```
train=model.matrix(Apps ~., data = x.train)
test = model.matrix(Apps ~., data = y.test)
fit.lasso = glmnet(train, x.train$Apps, alpha = 1, lambda = grid, thresh =
1e-12)
cv.lasso = cv.glmnet(train, x.train$Apps, alpha = 1, lambda = grid, thresh =
1e-12)
bestlam.lasso = cv.lasso$lambda.min
bestlam.lasso
```

```
## [1] 174.7528
```

```
pred.lasso=predict(fit.lasso,s=bestlam.lasso,newx=test)
mean((pred.lasso-y.test$Apps)^2)
```

```
## [1] 1117158
```

Test Error=1038776

### iii) Fit a ridge

```
train = model.matrix(Apps ~., data = x.train)
test = model.matrix(Apps ~., data = y.test)
```

```r
grid=10^seq(10,-2,length=100)
fit.ridge = glmnet(train, x.train$Apps, alpha = 0, lambda = grid, thresh =
1e-12)
cv.ridge = cv.glmnet(train, x.train$Apps, alpha = 0, lambda = grid, thresh =
1e-12)
bestlam.ridge = cv.ridge$lambda.min
bestlam.ridge
```

```
## [1] 0.01321941
```

```r
pred.ridge=predict(fit.ridge,s=bestlam.ridge,newx=test)
mean((pred.ridge-y.test$Apps)^2)
```

```
## [1] 1108509
```

Test Error=1108509 Based on the MSE, Lasso has a better performance amongst the three models

# Problem 2

## a)

```r
set.seed(1)
train=sample(1:nrow(Carseats),nrow(Carseats)/2)
train.data=Carseats[train,]
test.data= Carseats[-train,]
```

## b)

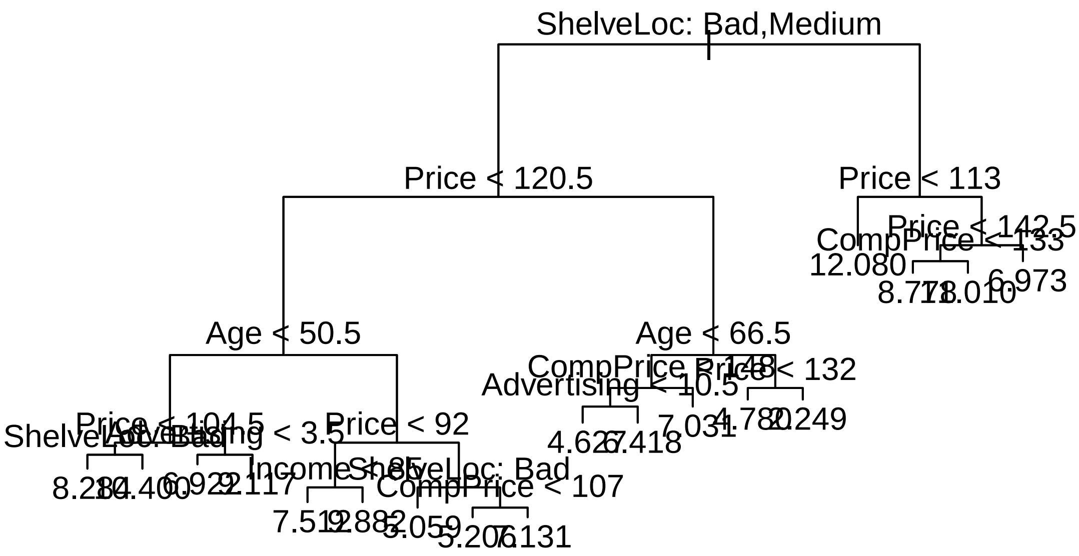
```r
library(tree)
```

```
## Warning: package 'tree' was built under R version 3.4.4
```

```r
tree.carseats=tree(Sales~.,data=train.data)
summary(tree.carseats)
```

```
##
## Regression tree:
## tree(formula = Sales ~ ., data = train.data)
## 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
```

```r
plot(tree.carseats)
text(tree.carseats,pretty = 0)
```



```
tree.pred=predict(tree.carseats,newdata=test.data)
mean((tree.pred-test.data$Sales)^2)          #Test MSE
```

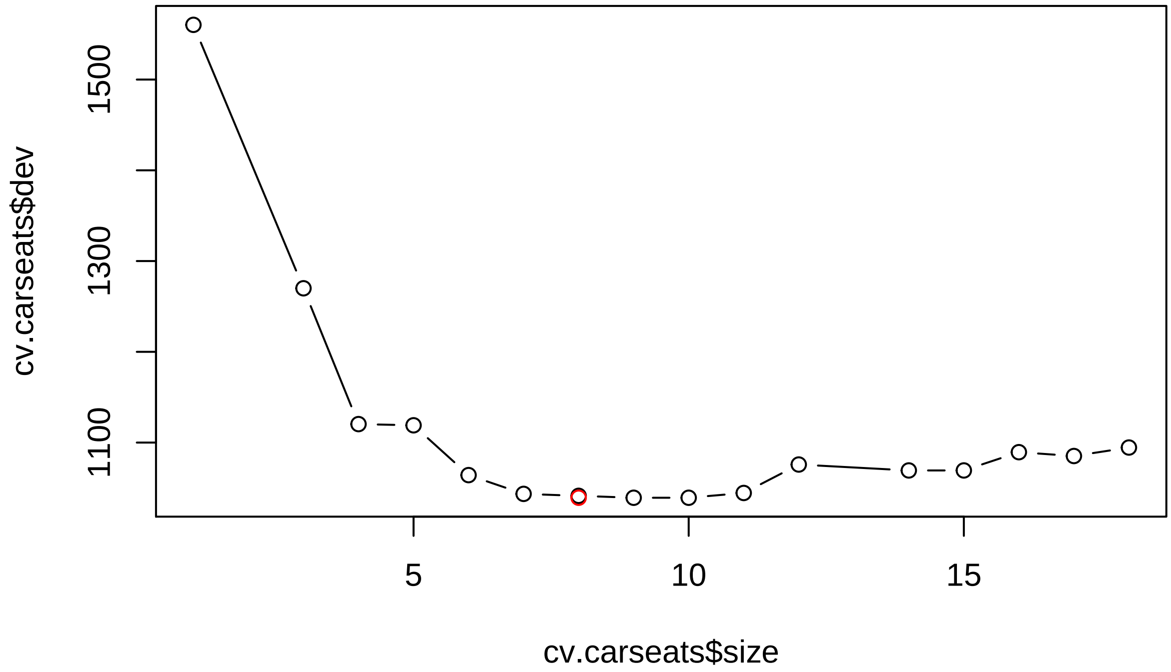
```
## [1] 4.148897
```

### c)

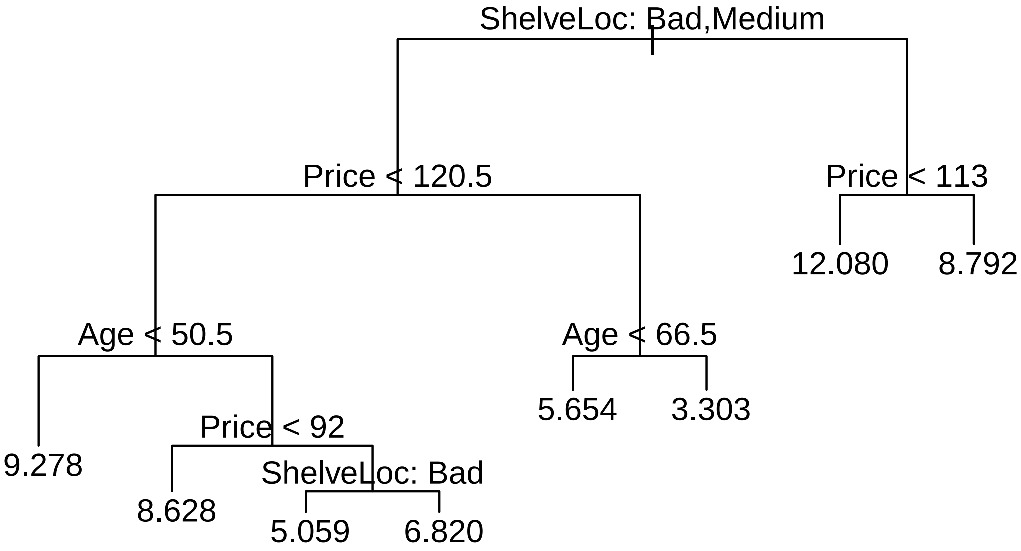
```
cv.carseats=cv.tree(tree.carseats)
plot(cv.carseats$size,cv.carseats$dev,type ="b")
tree.min=which.min(cv.carseats$dev)
tree.min
```

```
## [1] 8
```

```
points(tree.min,cv.carseats$dev[tree.min],col="red")
```



```
prune.carseats=prune.tree(tree.carseats,best=tree.min)
plot(prune.carseats)
text(prune.carseats, pretty=0)
```



```
tree.pred1=predict(prune.carseats,newdata=test.data)
mean((tree.pred1-test.data$Sales)^2)
```

```
## [1] 5.09085
```

MSE=5.09085 Pruning tree has increased test MSE from 4.148897 to 5.09085.

### d)

```
library(party)
```

```
## Warning: package 'party' was built under R version 3.4.4

## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Warning: package 'strucchange' was built under R version 3.4.4

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.4.4

##
```

```
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
## Warning: package 'sandwich' was built under R version 3.4.4
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.4
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
attach(Carseats)
set.seed(1)
bag.carseats= randomForest(Sales ~ ., train.data, mtry = 10, importance =
TRUE)
yhat.bag= predict(bag.carseats, newdata = test.data)
mean((yhat.bag - test.data$Sales)^2)
```

```
## [1] 2.614642
```

```
importance(bag.carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   16.4714051    126.605047
## Income       4.0561872     78.821925
## Advertising 16.2730251    122.793232
## Population   0.7711188     62.796112
## Price       54.5571815    512.940454
## ShelveLoc   42.4486118    320.749734
## Age         20.5369414    184.804253
## Education    2.7755968     42.427788
## Urban       -2.3962157      8.583232
## US           7.2258536     17.605661
```

Bagging process decreases test MSE to 2.61

## e)

```
set.seed(1)
rf.carseats=randomForest(Sales~.,data=train.data,mtry=3)
xhat=predict(rf.carseats,newdata=test.data)
mean((xhat-test.data$Sales)^2)
```

```
## [1] 3.267852
```

```
importance(rf.carseats)
```

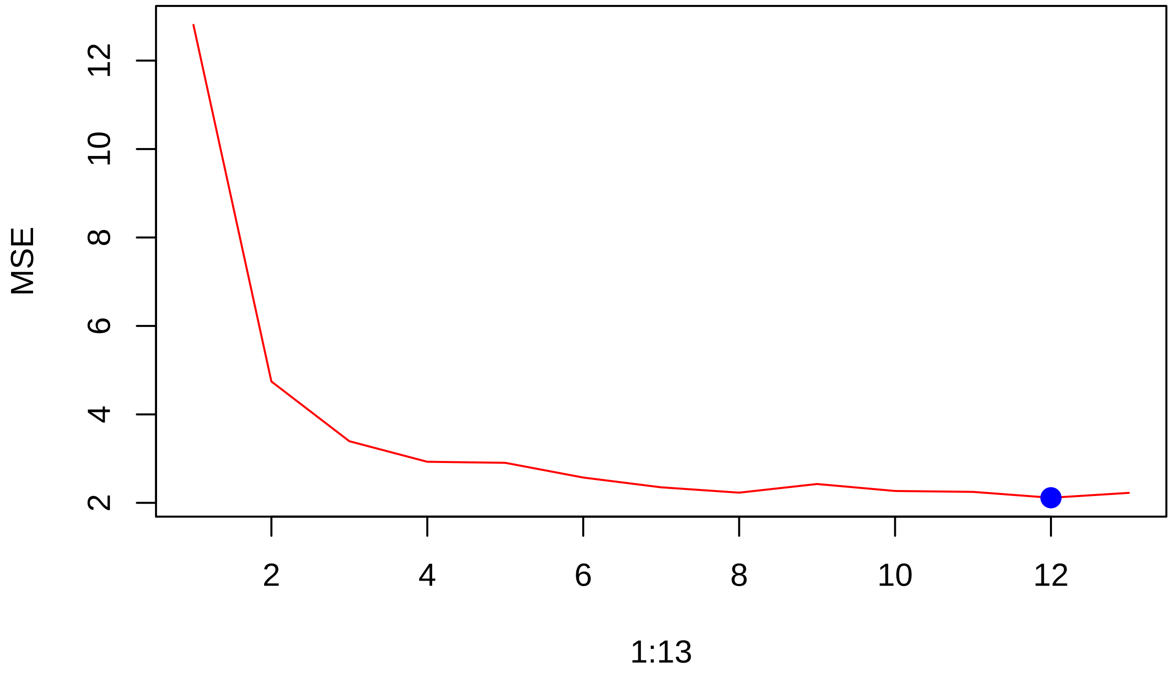
```
##             IncNodePurity
## CompPrice       134.77683
## Income          130.13842
## Advertising     139.26928
## Population       96.91406
## Price           376.12732
## ShelveLoc       240.96742
## Age             198.40681
## Education        64.60793
## Urban            16.08077
## US               32.57764
```

Test MSE=3.27

# Problem 3

## a)

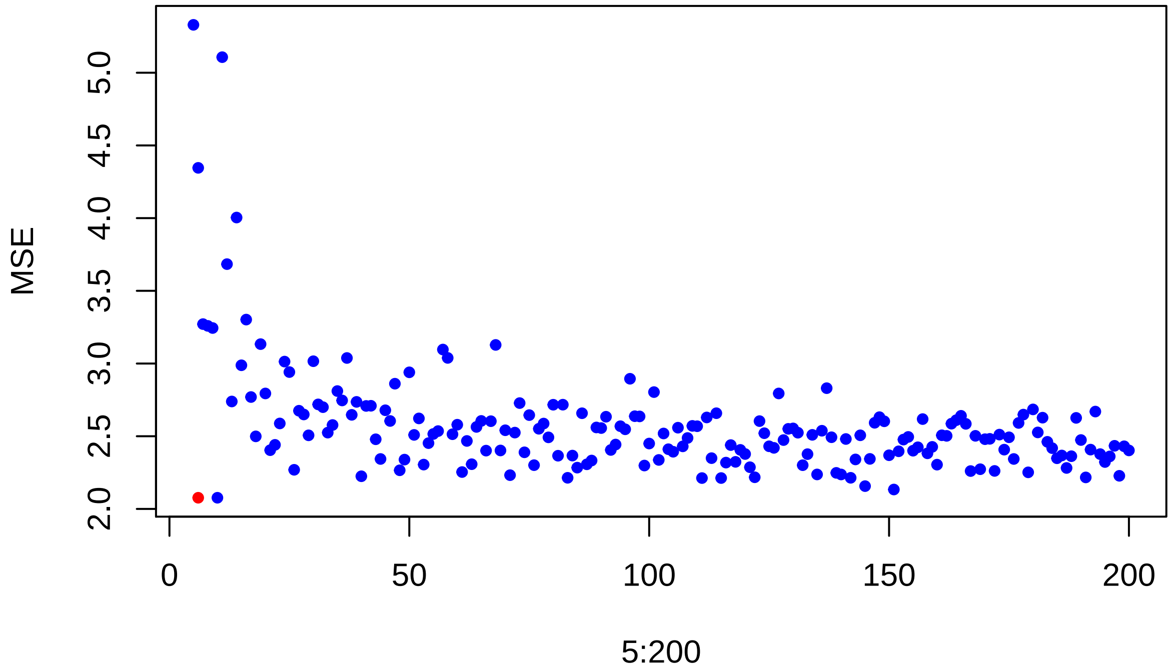
```
library(MASS)
attach(Boston)
trainbos=sample(1:nrow(Boston),nrow(Boston)/2)
testbos=Boston[-trainbos,"medv"]
set.seed(2)
MSE=replicate(0,13)

for (i in 1:13)
{
  rfbos=randomForest(medv~.,data=Boston[-trainbos,],mtry=i,ntree=100,importance=TRUE);
ycaprfbos=predict(rfbos,newdata=Boston[-trainbos,]);
MSE[i]=mean((ycaprfbos-testbos)^2)
  }
plot(1:13,MSE,col="red",cex=2,pch=20,type="l")
minMSE=which.min(MSE)
points(minMSE,MSE[minMSE],col="blue",cex=2,pch=20)
```



From the above graph it is evident that mtry=13 using random forest.

## b)

```
set.seed(5)
MSE=replicate(0,196)
for(i in 5:200)
{rfbos=randomForest(medv~.,data=Boston[-trainbos,],mtry=6,ntree=i,importance=TRUE);
ycaprfbos=predict(rfbos,newdata=Boston[-trainbos,]);
MSE[i-4]=mean((ycaprfbos-testbos)^2)}
plot(5:200,MSE,col="blue",cex=1,pch=20)
minMSE=which.min(MSE)
points(minMSE,MSE[minMSE],col="red",cex=1,pch=20)
```



From above plot is evident that for ntree=70 the MSE is lowest.