DS 502 –Statistical Methods for Data Science

Assignment 3

Submitted By:

Siddhant Bhavsar and Jay Trivedi

Assignment 3

1. **We perform best subset, forward stepwise, and backward stepwise selection on a single data set. For each approach, we obtain p +1 models, containing 0, 1, 2..., p predictors. Explain your answers:**

   (a) Which of the three models with k predictors has the smallest training RSS?
   Best subset selection will surely have the least training RSS as it considers all the models that can be made with each number of predictor and then selects the best one among them. But for forward/backward they add/remove variables one by one, hence the least error depends on the order of addition/removal of variables which may or may not be best.

   (b) Which of the three models with k predictors has the smallest test RSS?
   Any of the three models can have the least test RSS, though more often than not it best subset will have least RSS as it searches all possible models, but it might also overfit the data if p >>> n. Also, it might very well happen that for a particular test data forward/backward selection perfectly fits the data. Hence it is hard to say which method will have smallest test RSS.

   (c) True or False:

   i. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1)-variable model identified by forward stepwise selection.
   True: As forward stepwise adds one predictor after every step, so k+1 variable model will definitely have all the variables from k variable model plus a new variable which adds most information.

   ii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the (k +1) variable model identified by backward stepwise selection.
   True: As backward stepwise removes one predictor after every step, so a variable which gives least information is removed from k+1 variable model to get k variable model, hence the k variable model is subset of k+1 variable model

   iii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the (k +1) variable model identified by forward stepwise selection.
   False: Backward stepwise removes one predictor per step from the model while forward stepwise selection adds one predictor per step. Each next step here depends on the previous step i.e. which variable is selected to be added/removed maybe completely different for both the process, hence it cannot be said that it is always true, though there is non-zero probability that it will be true.

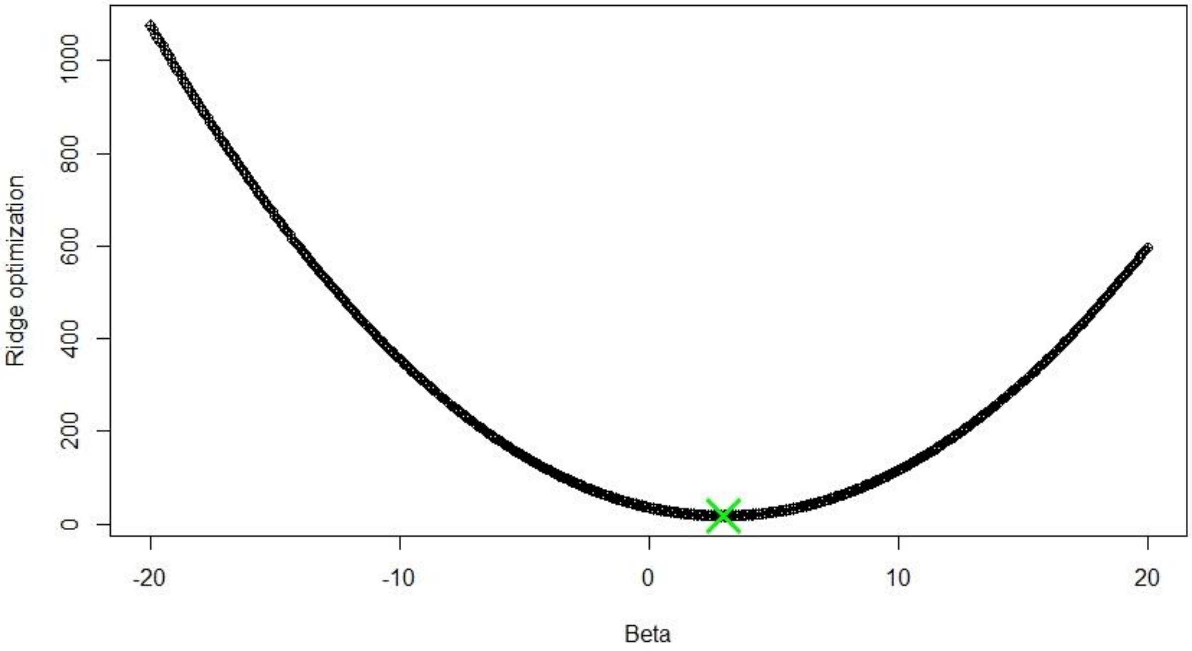
   iv. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1) variable model identified by backward stepwise selection.
   False: As explained above addition/removal of variable in forward/backward stepwise method depends on which variable is added/removed in the step before it, so although probability of statement being true is non-zero it is very small.

v. The predictors in the k-variable model identified by best subset are a subset of the predictors in the (k +1) variable model identified by best subset selection.
False: The best subset method at kth step selects the best model of k predictors and save it. In k+1th step it again considers all variables and select best k+1 variable irrespective of whether they occur in k-variable model.

2. **We will now explore (6.12) and (6.13) further.**
   (a) **Consider (6.12) with p = 1. For some choice of y1 and λ> 0, plot (6.12) as a function of β1. Your plot should confirm that (6.12) is solved by (6.14).**
   Code:

```
y1 = 6
l = 1
beta = seq(-20, 20, 0.1)
f = (y1 - beta) ^2 + l * beta^2
plot (beta, f, pch = 10, xlab = "Beta", ylab = "Ridge optimization")
beta_est = y1/ (1 + l)
func_est = (y1 - beta_est) ^2 + l * beta_est^2
points (beta_est, func_est, col = "green", pch = 4, lwd = 3, cex = beta_est)
```

Plot:



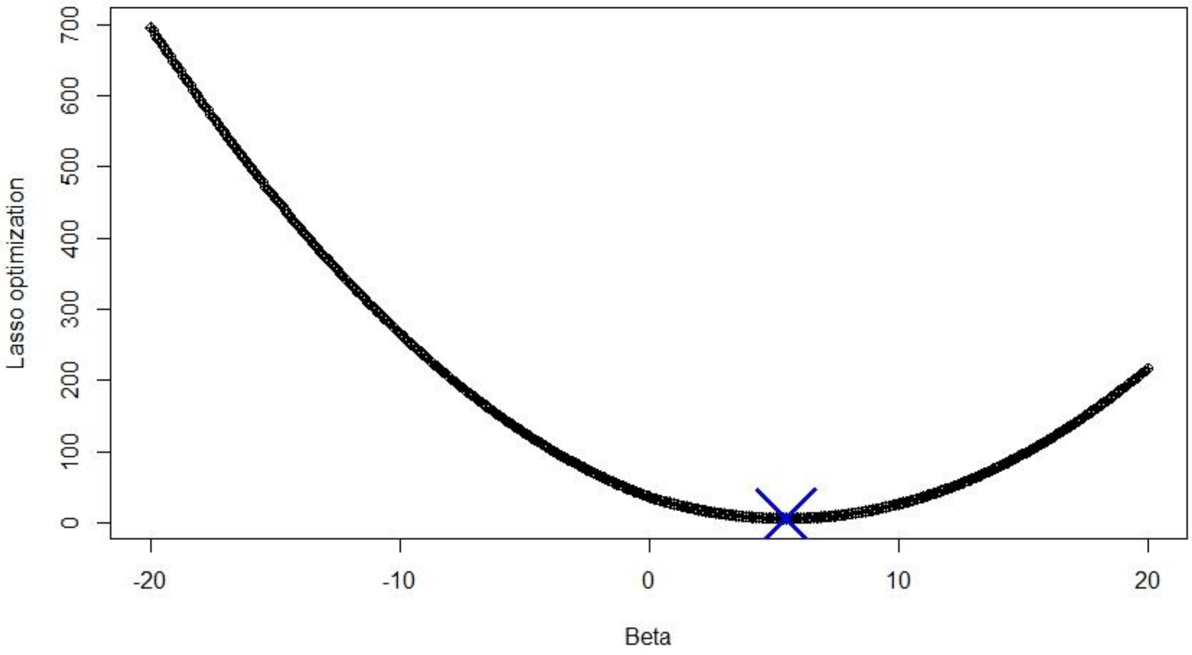
The green cross mark in the plot shows the estimated minimum value of ridge regression which coincides with the plot and as we can observe is the min value of the function. Here estimated minimum value is given by y1/(1+l). Hence 6.12 is solved by 6.14

(b) Consider (6.13) with p = 1. For some choice of y1 and λ> 0, plot (6.13) as a function of β1. Your plot should confirm that (6.13) is solved by (6.15).

Code:

```
y1 = 6
l = 1
beta = seq(-20, 20, 0.1)
f = (y1 - beta) ^2 + l * abs(beta)
plot (beta, f, pch = 10, xlab = "Beta", ylab = "Lasso optimization")
beta_est = y1 - l/2
func_est = (y1 - beta_est) ^2 + l * abs(beta_est)
points (beta_est, func_est, col = "blue", pch = 4, lwd = 3, cex = beta_est)
```

Plot:



The blue cross mark in the plot shows the estimated minimum value of lasso regression which coincides with the plot and as we can observe is the min value of the function. Here the estimated minimum value is given (y1 – lambda/2) as y1 > lambda/2. Hence 6.13 is solved by 6.15

3. In this exercise, we will generate simulated data, and will then use this data to perform best subset selection.

a. Use the rnorm() function to generate a predictor X of length n=100, as well as a noise vector E of length n=100.

Code:

```
set.seed(1)
X <- rnorm(100)
noise <- rnorm(100)
```

b. Generate a response vector Y of length n = 100 according to the model

$$Y=\beta_0+\beta_1X+\beta_2X_2+\beta_3X_3+\epsilon$$

Where β0,β1,β2,β3,β4 are constants of your choice.

Code:

```
b0 <- 2

b1 <- 1

b2 <- 3

b3 <- (-2)

y <- b0 + b1*X + b2*X^2 + b3*X^3 + noise
```

c. Use the regsubsets() function to perform best subset selection in order to choose the best model containing the predictors X,X2,⋯,X10. What is the best model obtained according to Cp, BIC, and adjusted R2? Show some plots to provide evidence for your answer and report the coefficients of the best model obtained. Note you will need to use the data.frame() function to create a single data set containing both X and Y.

Code:

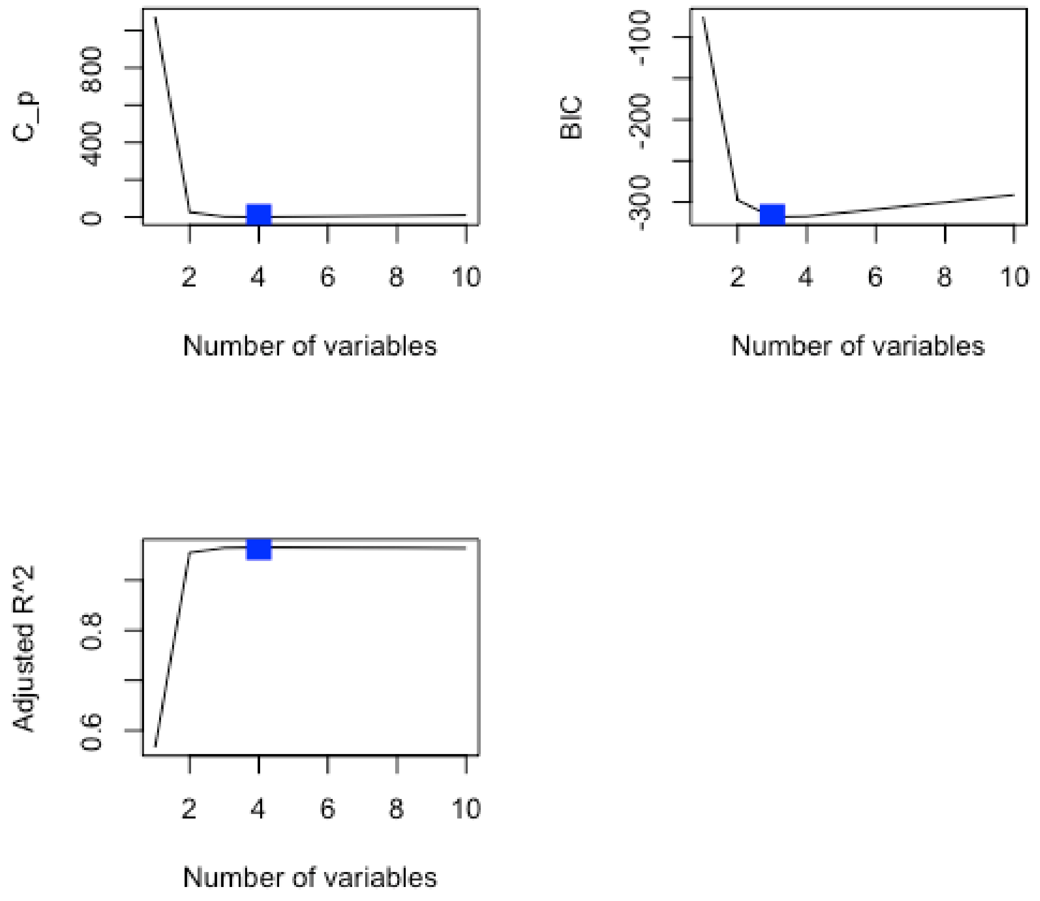
```
library(leaps)

dat=data.frame(Y=Y,X=X)
regfull <- regsubsets(Y ~ X + I(X^2) + I(X^3) + I(X^4) + I(X^5) + I(X^6) + I(X^7) +
I(X^8) + I(X^9) + I(X^10), data = dat, nvmax = 10)
regsummary=summary(regfull)
par(mfrow = c(2, 2))
plot(1:10,regsummary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
cp.min = min(regsummary$cp)
points(c(1:10)[regsummary$cp==cp.min],cp.min, col = "blue", cex = 2, pch = 15)
plot(1:10,regsummary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
bic.min = min(regsummary$bic)
```

```
points(c(1:10)[regsummary$bic==bic.min],bic.min, col = "blue", cex = 2, pch = 15)
plot(1:10,regsummary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2",
type = "l")
adjr2.max = max(regsummary$adjr2)
points(c(1:10)[regsummary$adjr2==adjr2.max],adjr2.max, col = "blue", cex = 2,
pch = 15)

coef(regfull,which.max(regsummary$adjr2))
coef(regfull,which.min(regsummary$bic))
```

Plots:



Output:

```
> coef(regfull,which.max(regsummary$adjr2))
(Intercept)          X            I(X^2)           I(X^3)          I(X^5)
 2.07200775  1.38745596  2.84575641 -2.44202574  0.08072292
```

```
> coef(regfull,which.min(regsummary$bic))
(Intercept)          X                I(X^2)        I(X^3)
  2.0615072   0.9752803   2.8762090  -1.9823614
```

The best model selected by Cp has four predictors: X, X2, X3 and X5. The best model selected by BIC has three predictors: X, X2 and X3. The best model selected by adjusted R2 is the same as the one selected by Cp, i.e. a model with predictors X, X2, X3 and X5.

d. Repeat(c), using forward stepwise selection and also using backwards stepwise selection. How does your answer compare to the results in (c)?

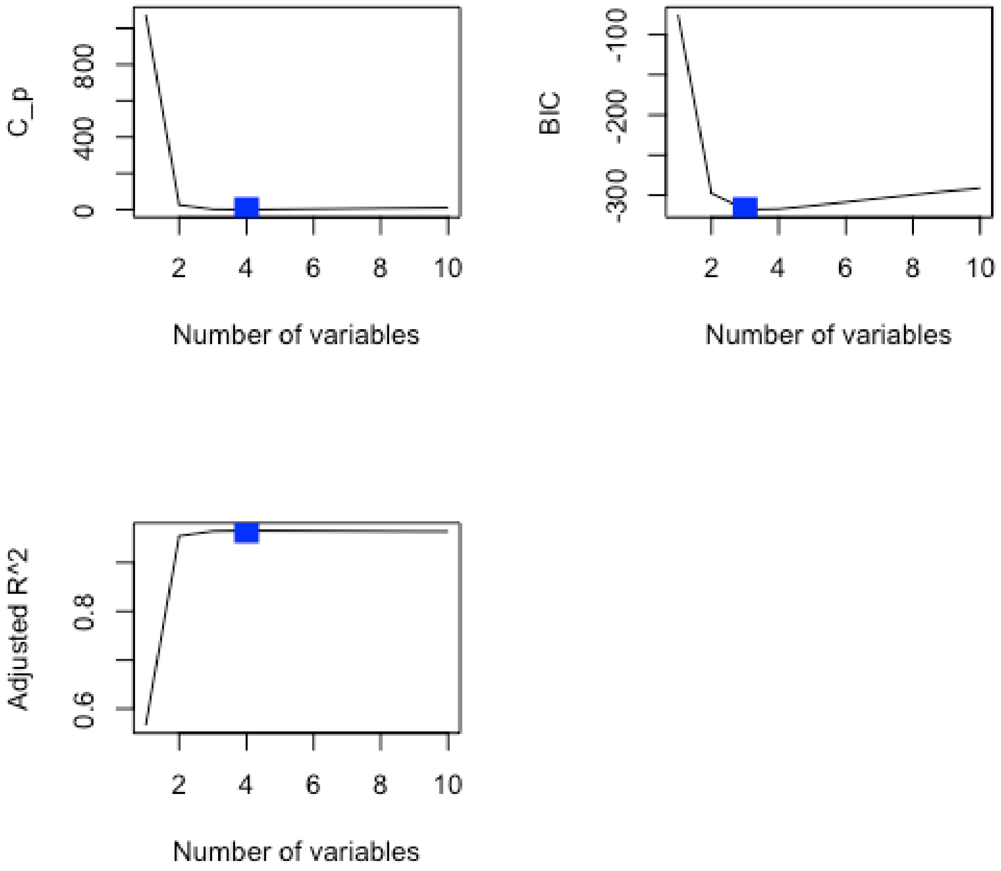
Forward Stepwise Selection

Code:

```
regfullfor <- regsubsets(Y ~ X + I(X^2) + I(X^3) + I(X^4) + I(X^5) + I(X^6) + I(X^7) +
I(X^8) + I(X^9) + I(X^10), data = dat, nvmax = 10, method = "forward")
regsummaryfor=summary(regfullfor)
par(mfrow = c(2, 2))
plot(1:10,regsummaryfor$cp, xlab = "Number of variables", ylab = "C_p", type =
"l")
cp.minfor = min(regsummaryfor$cp)
points(c(1:10)[regsummaryfor$cp==cp.minfor],cp.minfor, col = "blue", cex = 2, pch
= 15)
plot(1:10,regsummaryfor$bic, xlab = "Number of variables", ylab = "BIC", type =
"l")
bic.minfor = min(regsummaryfor$bic)
points(c(1:10)[regsummaryfor$bic==bic.minfor],bic.minfor, col = "blue", cex = 2,
pch = 15)
plot(1:10,regsummaryfor$adjr2, xlab = "Number of variables", ylab = "Adjusted
R^2", type = "l")
adjr2.maxfor = max(regsummaryfor$adjr2)
points(c(1:10)[regsummaryfor$adjr2==adjr2.maxfor],adjr2.maxfor, col = "blue",
cex = 2, pch = 15)

coef(regfullfor,which.max(regsummaryfor$adjr2))
coef(regfullfor,which.min(regsummaryfor$bic))
```

Plot:



Output:

```
> coef(regfullfor,which.max(regsummaryfor$adjr2))
(Intercept)          X              I(X^2)            I(X^3)       I(X^5)
 2.07200775  1.38745596  2.84575641 -2.44202574  0.08072292
> coef(regfullfor,which.min(regsummaryfor$bic))
(Intercept)          X                I(X^2)       I(X^3)
  2.0615072   0.9752803   2.8762090  -1.9823614
```

Forward selection method has a plot similar to (c). Even the number of predictor variables for Adjusted R^2 and C_p are 4 (X, X2, X3, X5)and number of predictor variables for BIC are 3 (X, X2, X3).

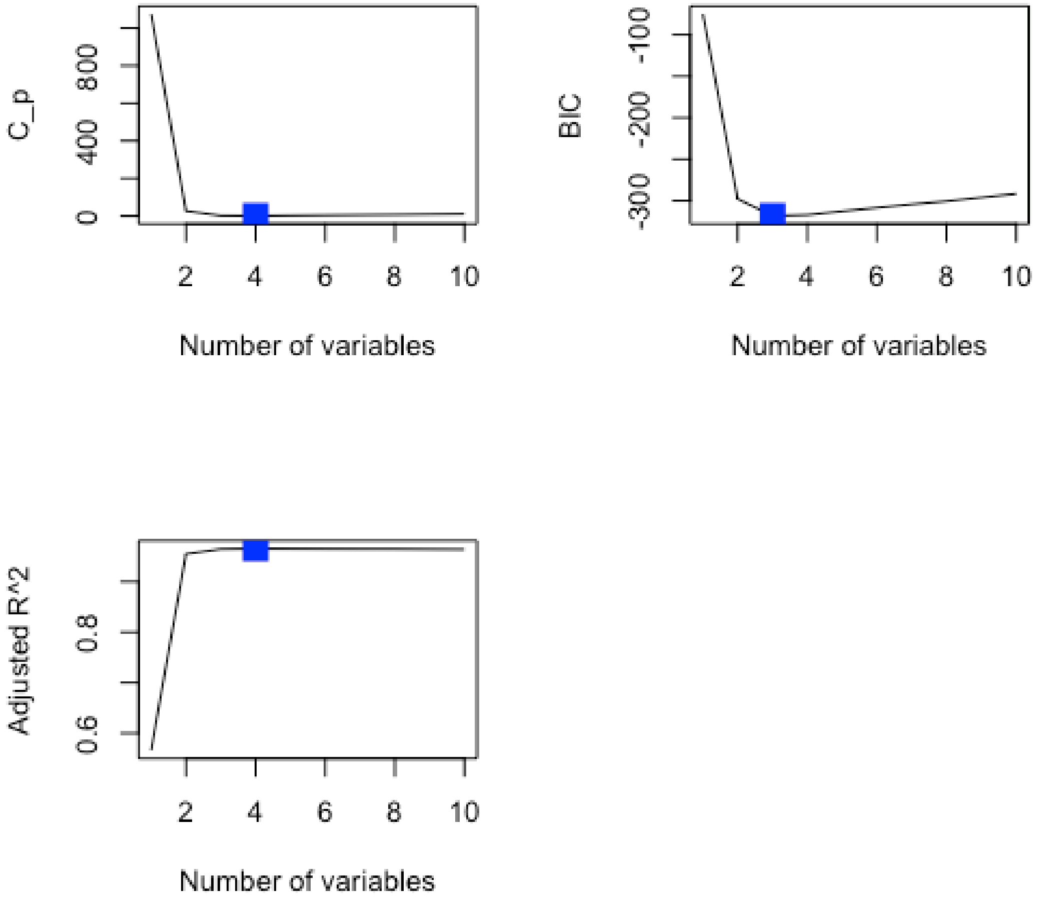
Backward Stepwise Selection:

Code:

```
regfullbac <- regsubsets(Y ~ X + I(X^2) + I(X^3) + I(X^4) + I(X^5) + I(X^6) + I(X^7) + I(X^8) + I(X^9) + I(X^10), data = dat, nvmax = 10, method = "backward")
regsummarybac=summary(regfullbac)
par(mfrow = c(2, 2))
plot(1:10,regsummarybac$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
cp.minbac = min(regsummarybac$cp)
points(c(1:10)[regsummarybac$cp==cp.minbac],cp.minbac, col = "blue", cex = 2, pch = 15)
plot(1:10,regsummarybac$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
bic.minbac = min(regsummarybac$bic)
points(c(1:10)[regsummarybac$bic==bic.minbac],bic.minbac, col = "blue", cex = 2, pch = 15)
plot(1:10,regsummarybac$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", type = "l")
adjr2.maxbac = max(regsummarybac$adjr2)
points(c(1:10)[regsummarybac$adjr2==adjr2.maxbac],adjr2.maxbac, col = "blue", cex = 2, pch = 15)

coef(regfullbac,which.max(regsummarybac$adjr2))
coef(regfullbac,which.min(regsummarybac$bic))
```

Plot:



Output:

```
> coef(regfullbac,which.max(regsummarybac$adjr2))
 (Intercept)           X              I(X^2)            I(X^3)           I(X^9)
 2.079236362  1.231905828  2.833494180 -2.180444193  0.001290827
> coef(regfullbac,which.min(regsummarybac$bic))
(Intercept)          X                I(X^2)        I(X^3)
  2.0615072   0.9752803   2.8762090  -1.9823614
```

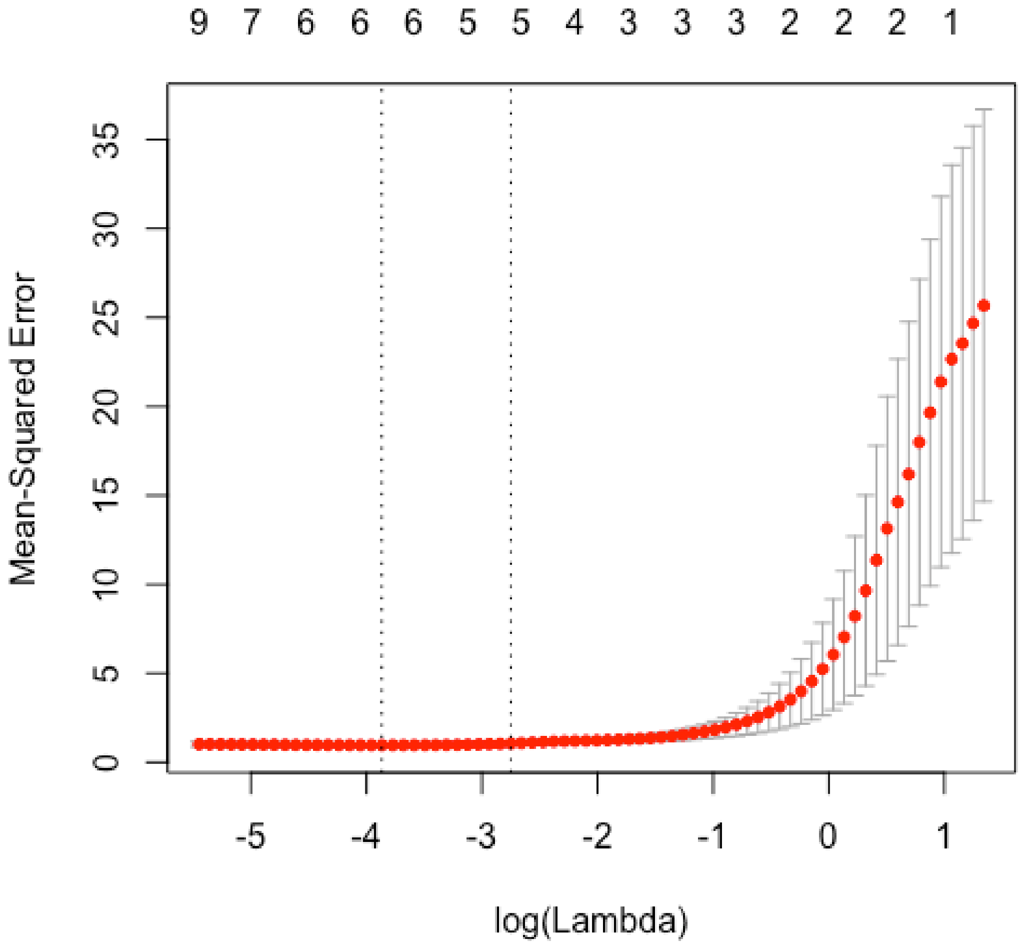
In Backward Stepwise selection, same number of predictor variables are chosen but the predictor variables chosen to have changed. Adjusted R square in backward stepwise selection choose predictor variables as X, X2, X3 and X9.

e. Now fit a lasso model to the simulated data, again using X, X2, . . . , X10 as predictors. 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, and discuss the results obtained.

Code:

```
library(glmnet)
set.seed(100)
y <- Y
x <- cbind(X,X^2,X^3,X^4,X^5,X^6,X^7,X^7,X^8,X^9,X^10)
lasso = cv.glmnet(x,y, alpha=1)
summary(lasso)
lasso$lambda.min
lasso$lambda.1se
plot(lasso)
lasso.mod=glmnet(x,y,alpha=1, lambda=lasso$lambda.min)
coef(lasso.mod)[,1]
summary(lasso)
```

Plot:



Output:

```
> library(glmnet)
> set.seed(100)
> y <- Y
> x <- cbind(X,X^2,X^3,X^4,X^5,X^6,X^7,X^7,X^8,X^9,X^10)
> lasso = cv.glmnet(x,y, alpha=1)
> summary(lasso)
       Length Class  Mode
lambda    74   -none- numeric
cvm       74   -none- numeric
cvsd      74   -none- numeric
cvup      74   -none- numeric
cvlo      74   -none- numeric
nzero     74   -none- numeric
```

```
           name        1     -none- character
glmnet.fit 12     elnet  list
lambda.min  1     -none- numeric
lambda.1se  1     -none- numeric
> lasso$lambda.min
[1] 0.02092228
> lasso$lambda.1se
[1] 0.06389363
> plot(lasso)
> lasso.mod=glmnet(x,y,alpha=1, lambda=lasso$lambda.min)
> coef(lasso.mod)[,1]
  (Intercept)            X
 2.149878e+00  8.962685e-01  2.702807e+00 -1.963746e+00  8.786128e-03
0.000000e+00  2.460194e-03  0.000000e+00

 0.000000e+00  5.705123e-04  0.000000e+00  3.466511e-05
> summary(lasso)
          Length Class  Mode
lambda     74     -none- numeric
cvm        74     -none- numeric
cvsd       74     -none- numeric
cvup       74     -none- numeric
cvlo       74     -none- numeric
nzero      74     -none- numeric
name        1     -none- character
glmnet.fit 12     elnet  list
lambda.min  1     -none- numeric
lambda.1se  1     -none- numeric
```

So, this plot has a cross validation curve which is a red dotted line, and the standard deviation curve with respect to values of lambda. Lambda giving the minimum cv error and the lambda within one standard deviation of the minimum cv error are selected by indicating two vertical lines in the plot. Now the Lasso shrinks majority of the predictors to zero leaving only X5 and X6 as they give the minimum CV error.

f. **Now generate a response vector Y according to the model $Y = \beta_0 + \beta_7X^7 + \varepsilon$, and perform best subset selection and the lasso. Discuss the results obtained.**

Code:

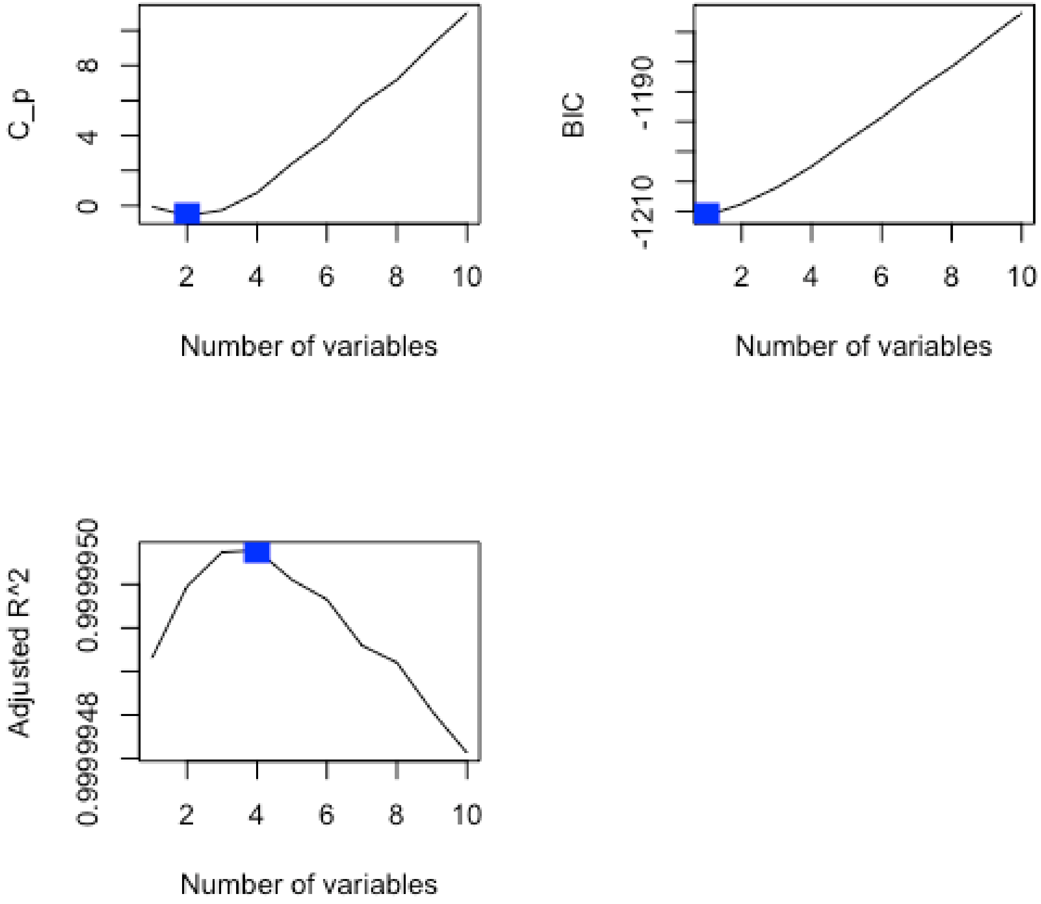
```
B7 <- 9
Y <- B0 +  B7*X^7 + noise
dat=data.frame(Y=Y,X=X)
library(leaps)
```

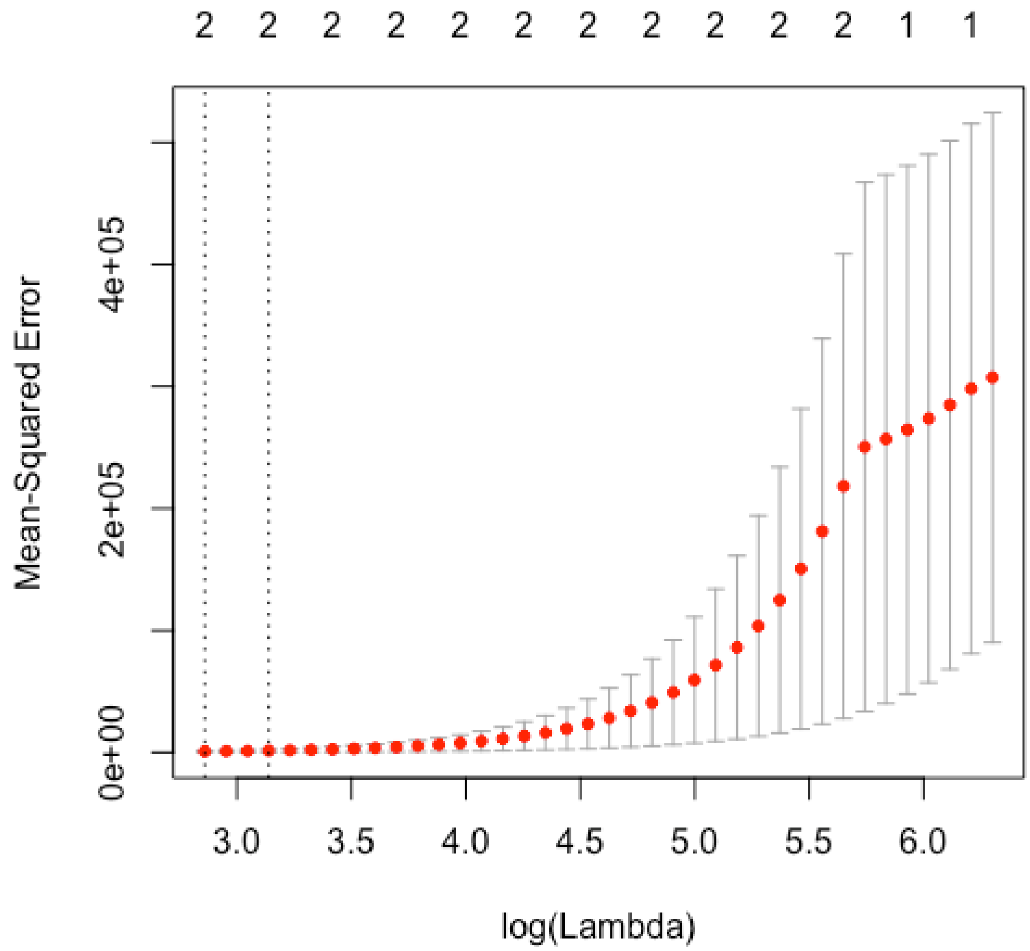
```
regfull <- regsubsets(Y ~ X + I(X^2) + I(X^3) + I(X^4) + I(X^5) + I(X^6) + I(X^7) +
I(X^8) + I(X^9) + I(X^10), data = dat, nvmax = 10)
regsummary=summary(regfull)
par(mfrow = c(2, 2))
plot(1:10,regsummary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
cp.min = min(regsummary$cp)
points(c(1:10)[regsummary$cp==cp.min],cp.min, col = "blue", cex = 2, pch = 15)
plot(1:10,regsummary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
bic.min = min(regsummary$bic)
points(c(1:10)[regsummary$bic==bic.min],bic.min, col = "blue", cex = 2, pch = 15)
plot(1:10,regsummary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2",
type = "l")
adjr2.max = max(regsummary$adjr2)
points(c(1:10)[regsummary$adjr2==adjr2.max],adjr2.max, col = "blue", cex = 2,
pch = 15)

coef(regfull,which.max(regsummary$adjr2))
coef(regfull,which.min(regsummary$bic))
coef(regfull,which.min(regsummary$cp))


library(glmnet)
set.seed(1)
y <- Y
x <- cbind(X,X^2,X^3,X^4,X^5,X^6,X^7,X^7,X^8,X^9,X^10)
lasso = cv.glmnet(x,y, alpha=1)
summary(lasso)
lasso$lambda.min
lasso$lambda.1se
plot(lasso)
lasso.mod=glmnet(x,y,alpha=1, lambda=lasso$lambda.min)
coef(lasso.mod)[,1]
summary(lasso)
```

Plot:


C_p
0
4
8
2
4
6
8
10
Number of variables
BIC
-1210
-1190
2
4
6
8
10
Number of variables
Adjusted R^2
0.9999948
0.9999950
2
4
6
8
10
Number of variables



Output:

```
> coef(regfull,which.max(regsummary$adjr2))
(Intercept)          X              I(X^2)          I(X^3)          I(X^7)
  2.0762524   0.2914016  -0.1617671  -0.2526527   9.0091338
> coef(regfull,which.min(regsummary$bic))
(Intercept)      I(X^7)
    1.95894     9.00077
> coef(regfull,which.min(regsummary$cp))
(Intercept)        I(X^2)          I(X^7)
  2.0704904  -0.1417084   9.0015552

> library(glmnet)
> set.seed(1)
> y <- Y
> x <- cbind(X,X^2,X^3,X^4,X^5,X^6,X^7,X^7,X^8,X^9,X^10)
> lasso = cv.glmnet(x,y, alpha=1)
> summary(lasso)
         Length Class  Mode
lambda     38     -none- numeric
cvm        38     -none- numeric
cvsd       38     -none- numeric
cvup       38     -none- numeric
cvlo       38     -none- numeric
nzero      38     -none- numeric
name        1     -none- character
glmnet.fit 12     elnet  list
lambda.min  1     -none- numeric
lambda.1se  1     -none- numeric
> lasso$lambda.min
[1] 17.45287
> lasso$lambda.1se
[1] 23.07166
> plot(lasso)
> lasso.mod=glmnet(x,y,alpha=1, lambda=lasso$lambda.min)
> coef(lasso.mod)[,1]
 (Intercept)            X
3.180442e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
0.000000e+00 0.000000e+00 8.695863e+00 1.124199e-15

0.000000e+00 2.748771e-03 0.000000e+00
```

We see cp chooses 2 variable model, BIC chooses 1 variable model while adj r sqaure chooses 4 variable model. Here the lasso chooses the best model with 3 variables.

4. In this exercise, we will predict the number of applications received using the other variables in the College data set.

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

Code:

```
library(ISLR)
data(College)
set.seed(11)
sum(is.na(College))
trainsize = dim(College)[1] / 2
train = sample(1:dim(College)[1], trainsize )
test <- -train
Collegetrain <- College[train, ]
```

```
Collegetest <- College[test, ]
```

b. Fit a linear model using least squares on the training set and report the test error obtained.

Code:

```
linear_model <- lm(Apps ~ ., data = Collegetrain)
summary(linear_model)
linear_prediction <- predict(linear_model, Collegetest)
linear_MSE <- mean((linear_prediction - Collegetest$Apps)^2); linear_MSE
```

Output:

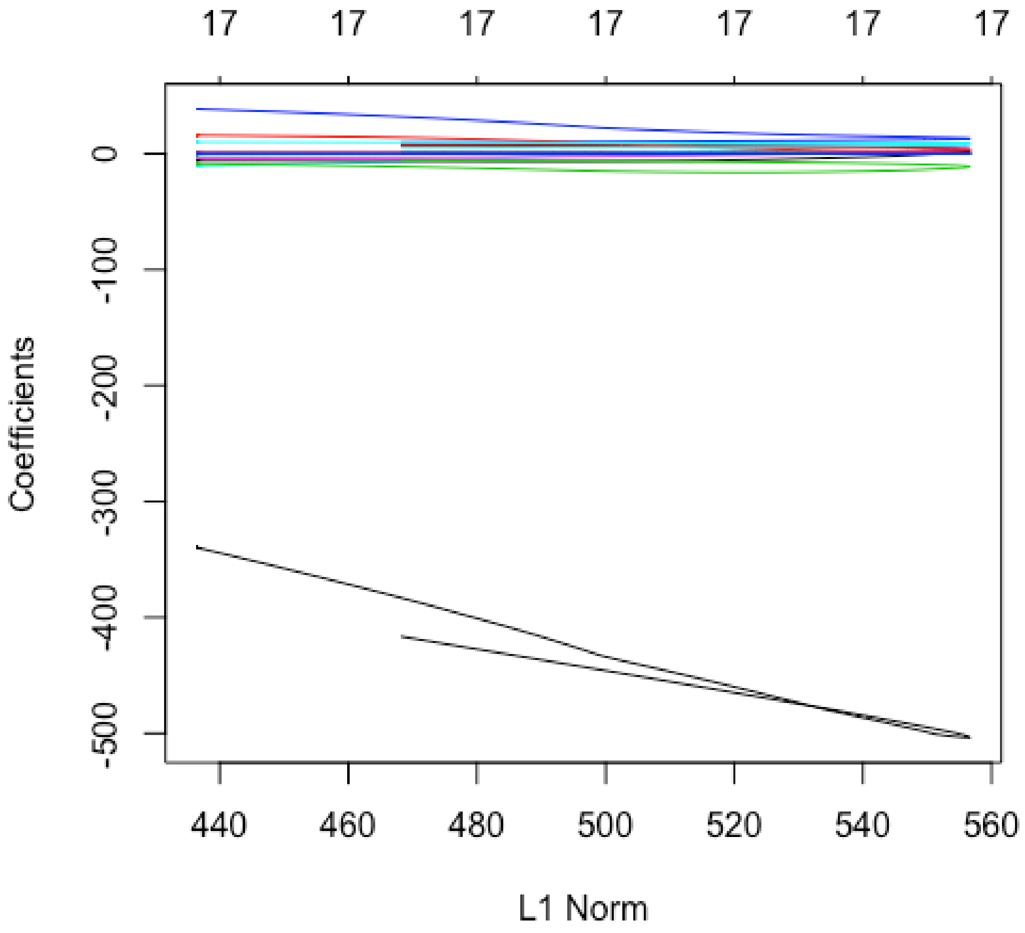
```
> linear_MSE <- mean((linear_prediction - Collegetest$Apps)^2); linear_MSE
[1] 1538442
```

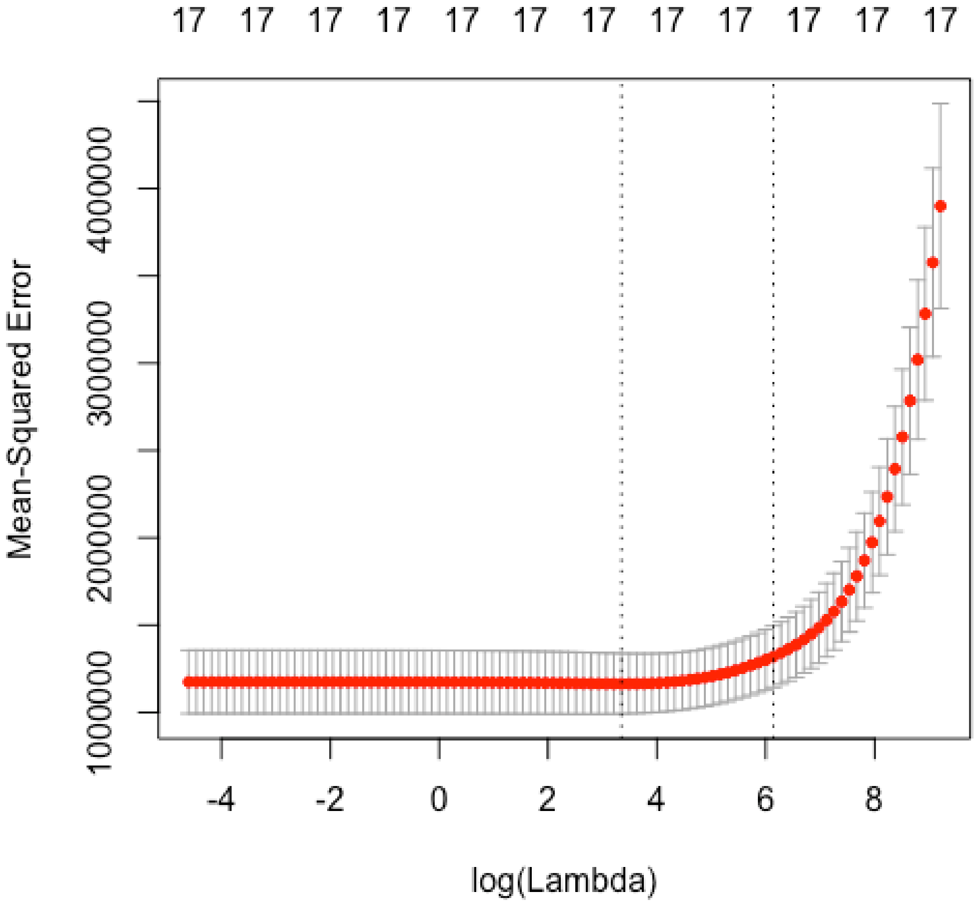
c. Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

Code:

```
trainmatrix <- model.matrix(Apps ~ ., data = Collegetrain)
testmatrix <- model.matrix(Apps ~ ., data = Collegetest)
grid = 10^seq(4, -2, length=100)
ridge <- glmnet(trainmatrix, Collegetrain$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
plot(ridge)
glmridge  <- cv.glmnet(trainmatrix, Collegetrain$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
plot(glmridge)
bestlambda <- glmridge$lambda.min
bestlambda
pred.ridge <- predict(ridge, s = bestlambda, newx = testmatrix)
mean((pred.ridge - Collegetest$Apps)^2)
```

Plot:


17
17
17
17
17
17
17
0
-100
-200
-300
-400
-500
Coefficients
440
460
480
500
520
540
560
L1 Norm



Output:

```
> bestlambda
[1] 18.73817
> mean((pred.ridge - Collegetest$Apps)^2)
[1] 1608859
```

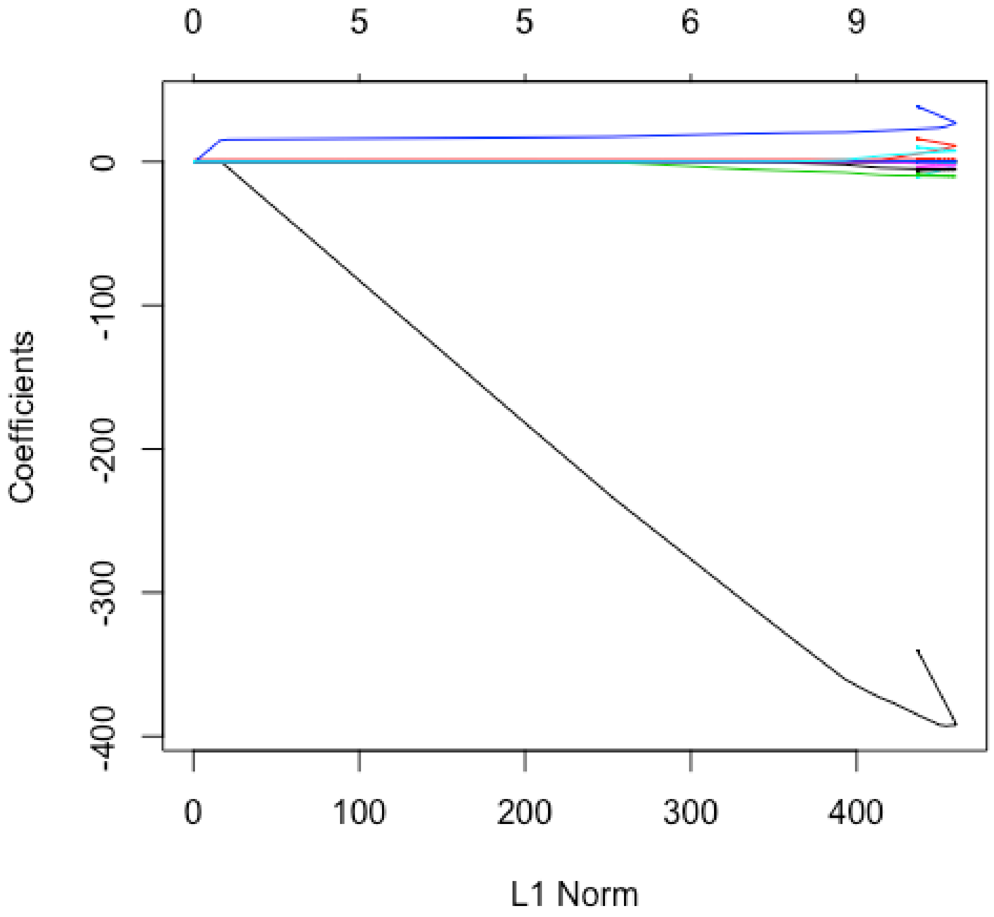
d. Fit a lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates.

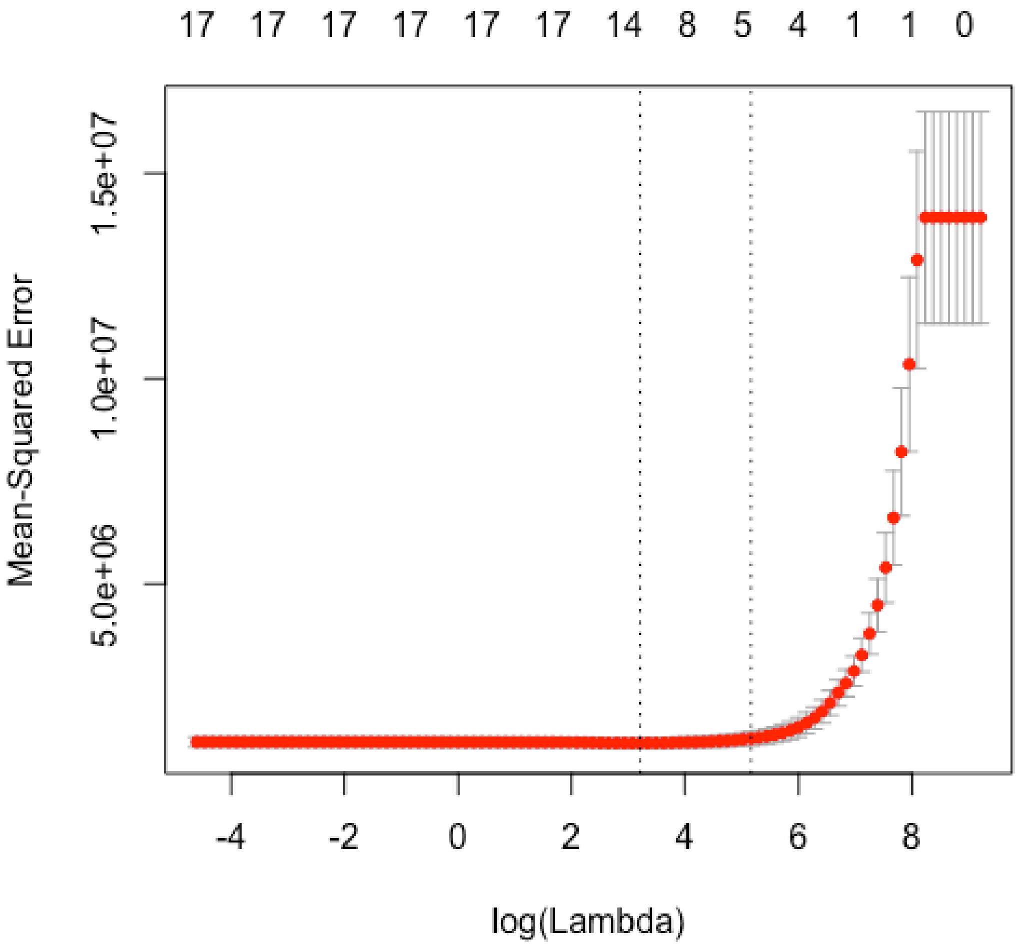
Code:

```
lasso <- glmnet(trainmatrix, Collegetrain$Apps, alpha = 1, lambda = grid, thresh =
1e-12)
plot(lasso)
cvlasso <- cv.glmnet(trainmatrix, Collegetrain$Apps, alpha = 1, lambda = grid,
thresh = 1e-12)
plot(cvlasso)
```

```
bestlambdalasso <- cvlasso$lambda.min
bestlambdalasso
pred.lasso <- predict(lasso, s = bestlambdalasso, newx = testmatrix)
mean((pred.lasso - Collegetest$Apps)^2)
predict(lasso, s = bestlambdalasso, type = "coefficients")
```

Plot:





Output:

```
> bestlambdalasso
[1] 24.77076
> mean((pred.lasso - Collegetest$Apps)^2)
[1] 1639664
> predict(lasso, s = bestlambdalasso, type = "coefficients")
19 x 1 sparse Matrix of class "dgCMatrix"
                      1
(Intercept) -8.237282e+02
(Intercept)  .
PrivateYes  -3.816547e+02
Accept       1.176501e+00
Enroll       .
Top10perc    2.225429e+01
Top25perc    .
F.Undergrad  7.248741e-02
```

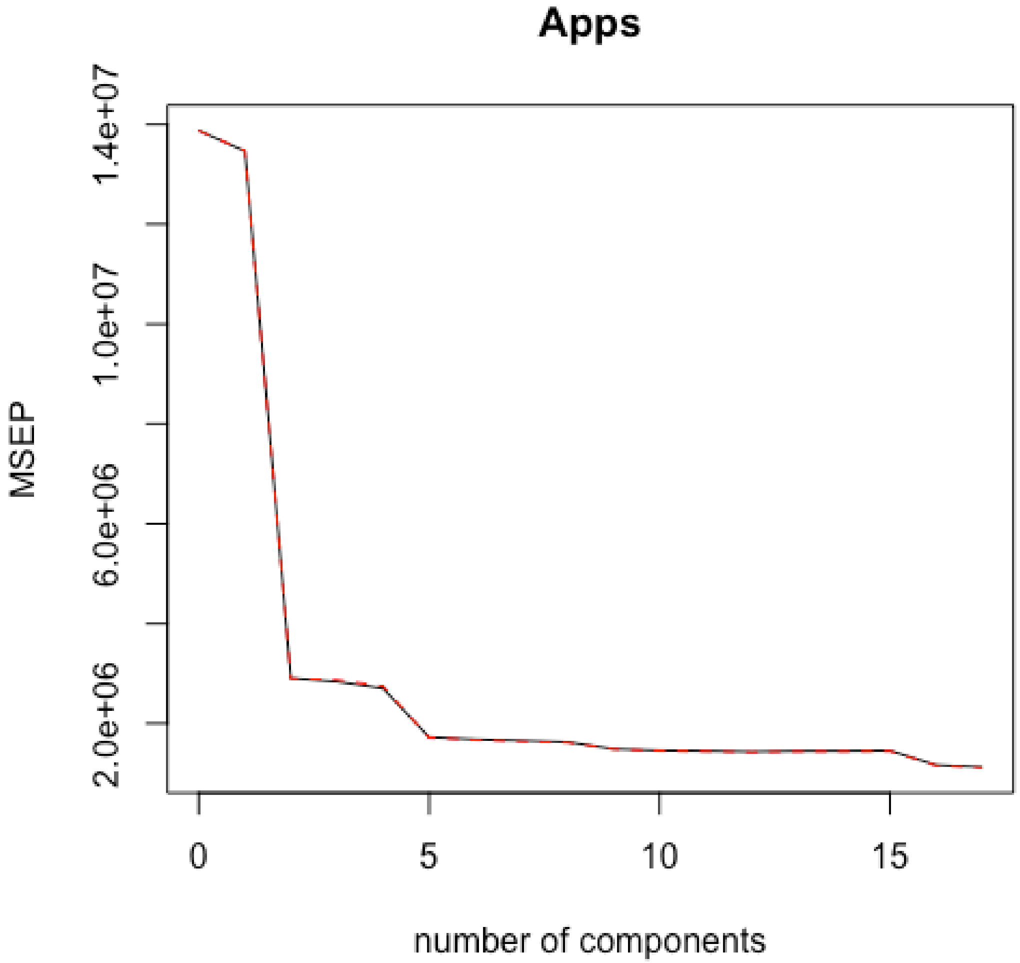
```
P.Undergrad  9.876833e-03
Outstate    -2.967710e-02
Room.Board   5.628549e-03
Books       .
Personal    .
PhD        -1.044834e+00
Terminal    -4.946523e+00
S.F.Ratio    4.216147e+00
perc.alumni -9.808938e+00
Expend       1.455640e-01
Grad.Rate    5.307550e+00
```

e. Fit a PCR model on the training set, with M chosen by cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.

Code:

```
library(pls)
pcr <- pcr(Apps ~ ., data = Collegetrain, scale = TRUE, validation = "CV")
validationplot(pcr, val.type = "MSEP")
predpcr <- predict(pcr, Collegetest, ncomp = 10)
mean((predpcr - Collegetest$Apps)^2)
```

Plot:



Output:

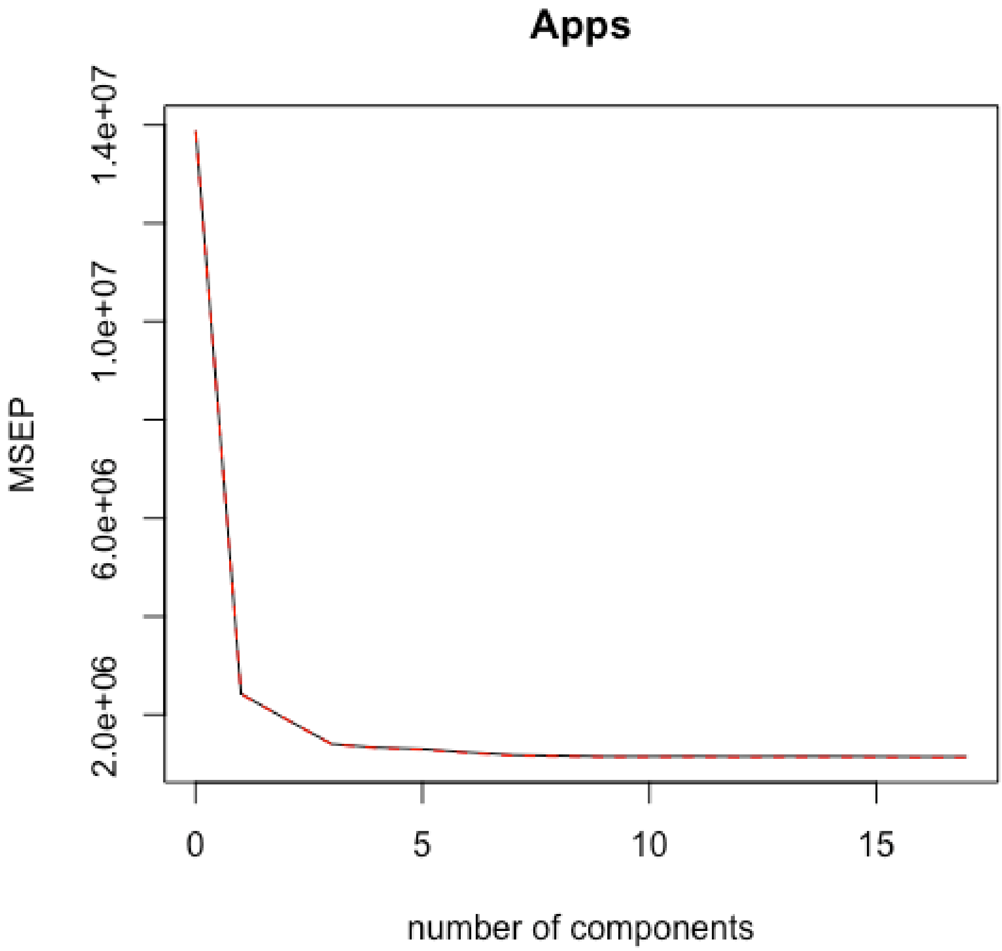
```
> mean((predpcr - Collegetest$Apps)^2)
[1] 3014496
```

f. **Fit a PLS model on the training set, with M chosen by cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.**

Code:

```
pls <- plsr(Apps ~ ., data = Collegetrain, scale = TRUE, validation = "CV")
validationplot(pls, val.type = "MSEP")
predpls <- predict(pls, Collegetest, ncomp = 10)
mean((predpls - Collegetest$Apps)^2)
```

Plot:



Output:

```
> mean((predpls - Collegetest$Apps)^2)
[1] 1508987
```

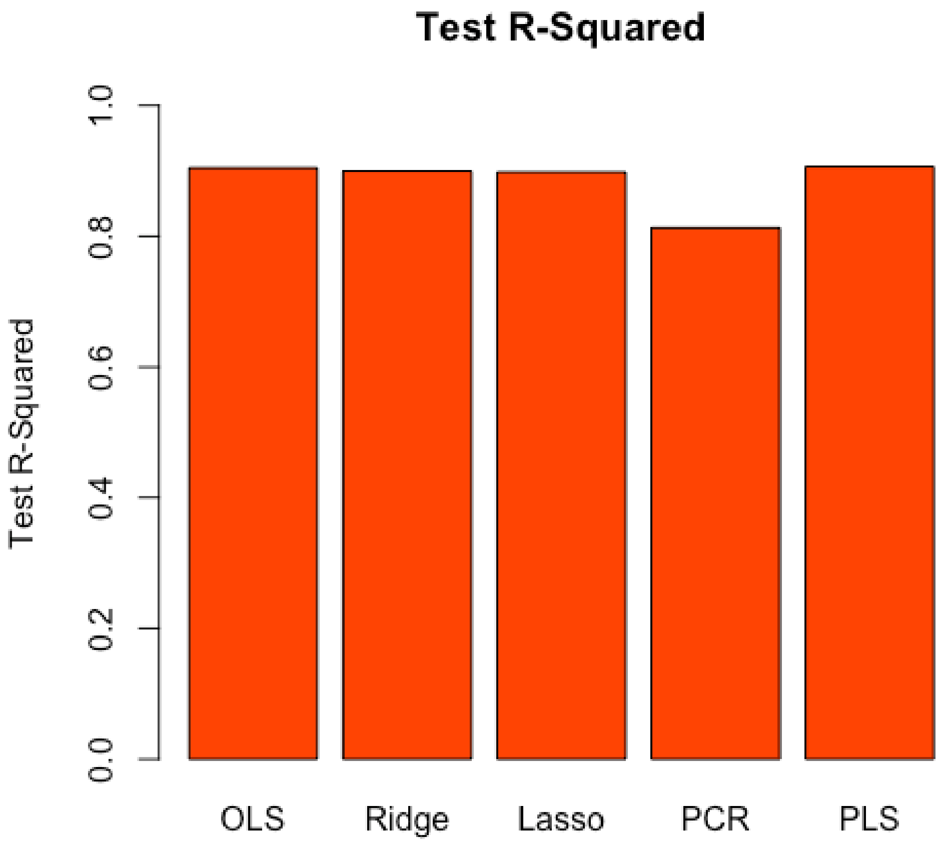
g. Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these five approaches?

Code:

```
testavg <- mean(Collegetest[, "Apps"])
linearr = 1 - mean((Collegetest[, "Apps"] -
linear_prediction)^2)/mean((Collegetest[, "Apps"] - testavg)^2)
ridger = 1 - mean((Collegetest[, "Apps"] -pred.ridge)^2) /mean((Collegetest[,
"Apps"] - testavg)^2)
lassor = 1 - mean((Collegetest[, "Apps"] -pred.lasso)^2) /mean((Collegetest[,
"Apps"] - testavg)^2)
```

```
pcrr = 1 - mean((Collegetest[, "Apps"] -predpcr)^2) /mean((Collegetest[, "Apps"] -
testavg)^2)
plsr = 1 - mean((Collegetest[, "Apps"] -predpls)^2) /mean((Collegetest[, "Apps"] -
testavg)^2)
par(mfrow = c(1,2))
# Let's create a bar plot of the R2 values to visualize any differences
barplot(c(linearr, ridger, lassor, pcrr, plsr), col="orangered",
                names.arg=c("OLS","Ridge", "Lasso", "PCR", "PLS"), main = "Test R-
        Squared",
                ylab = "Test R-Squared", ylim = c(0,1))
testavg
linearr
ridger
lassor
pcrr
plsr
```

Plot:



Output:

```
> testavg
[1] 3077.612
> linearr
[1] 0.9044281
> ridger
[1] 0.9000536
> lassor
[1] 0.8984123
> pcrr
[1] 0.8127319
> plsr
[1] 0.9062579
```

All models except PCR, predict college applications with high accuracy.
The test R^2 for Linear, Ridge, Lasso, PCR, PLS are 0.9044,0.9,0.898,0.812,0.906 respectively.