

**DS502 - HW3**  
**Vandana Anand**  
**Kratika Agrawal**

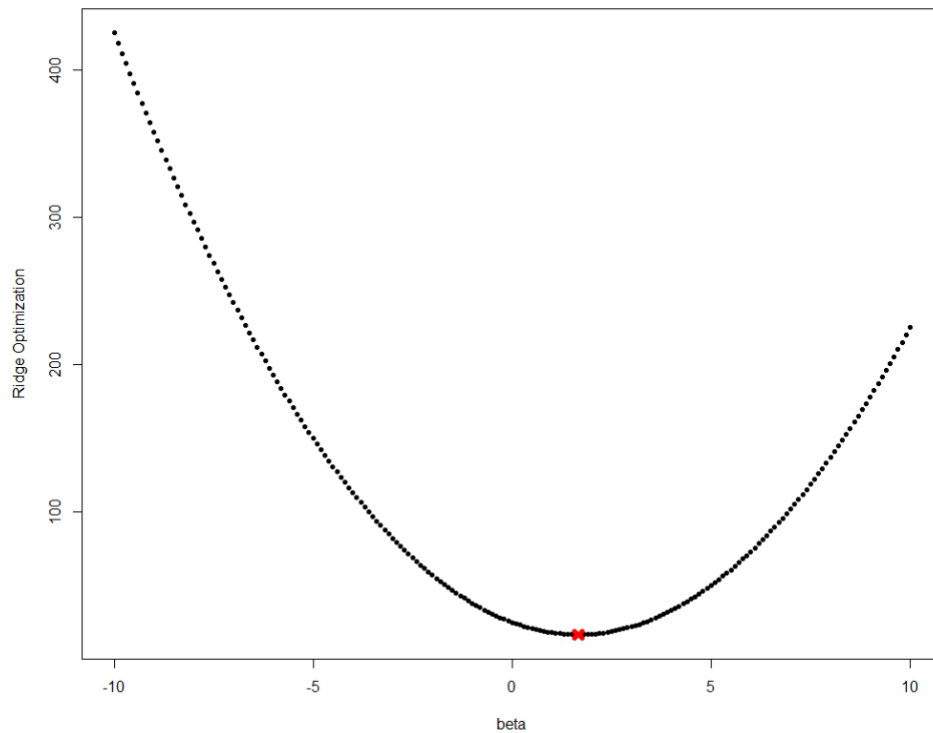
**Question 1:**

- a) Best subset will have the smallest training RSS because it takes into account all the  $K$  predictors.
- b) Cannot tell exactly. Best subset can have the smallest test RSS because it accounts for more models than forward or backward selection, but forward or backward selection can also pick the best one by chance.
- c) i) True, the  $(k+1)$  model is achieved by adding an additional predictor to the model with  $k$  predictors  
ii) True, the model with  $k$  predictors is achieved by removing a predictor from the model with  $(k+1)$  predictors  
iii) False, there is no explicit relation between the forward and backward models  
iv) False, same as above in which there is no explicit relation between forward and backward models  
v) False, the model with  $(k+1)$  predictors is achieved by selecting from a list of possible  $(k+1)$  models which does not mean that all predictors will be used from the model with  $k$  predictors

**Question 2:**

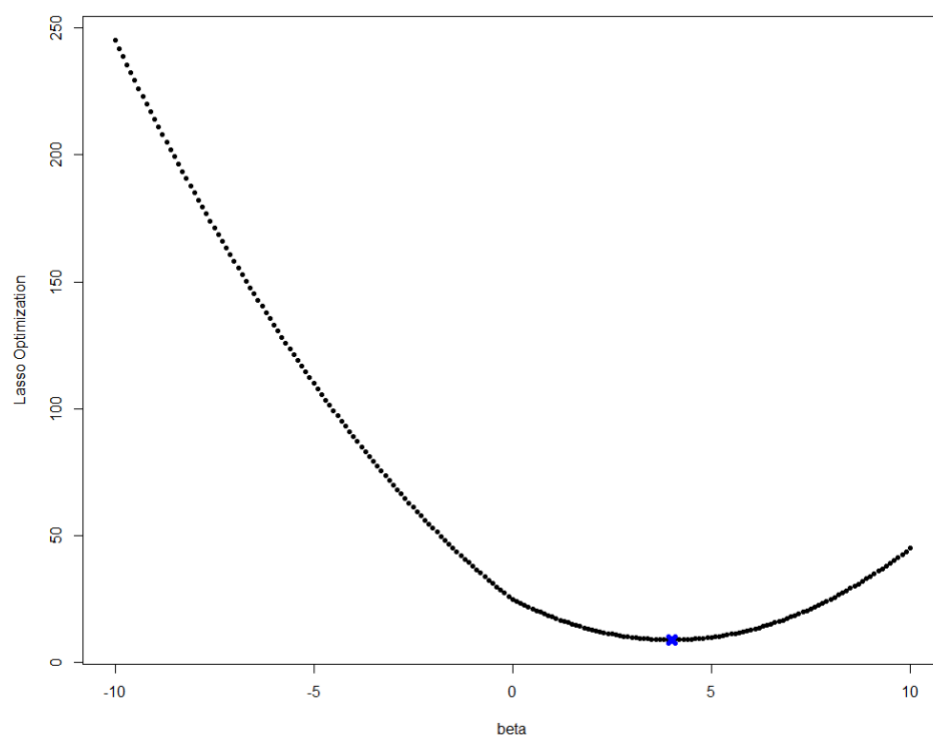
- a) Ridge-Regression optimization plot for  $p=1$ . Minimum point is marked with red-cross

```
> #answer a
> y=5
> lambda=2
> beta=seq(from=-10,to=10,by=0.1)
> ridge = (y - beta)^2 + lambda * beta
> plot(beta, ridge, pch = 20, xlab = "beta", ylab = "Ridge Optimization")
> beta.est = y / (1 + lambda)
> points(beta.est, (y - beta.est)^2 + lambda * beta.est^2, col = "red", pch = 4, lwd = 5)
```



b) Lasso optimization plot for  $p=1$ . Minimum point is marked with blue-cross

```
> #answer b
> y1=5
> lambda1=2
> beta1=seq(from=-10,to=10,by=0.1)
> lasso = (y1 - beta1)^2 + lambda1 * abs(beta1)
> plot(beta1, lasso, pch = 20, xlab = "beta", ylab = "Lasso Optimization")
> beta.est1 = y1 - lambda1 / 2
> points(beta.est1, (y1 - beta.est1)^2 + lambda1 * abs(beta.est1), col = "blue", pch = 4, lwd = 5)
```



### Question 3:

a)

```
> X = rnorm(100)
> print(X)
[1] 1.572811625 0.780971848 -1.420947777 1.313922852 1.500466967 1.572601419 0.320873510
[8] 0.266054665 -0.358781423 -1.455010644 0.008345388 0.915967690 -0.352883504 0.057068022
[15] -0.371916790 -0.086373366 1.552282794 1.092933176 -0.336224420 0.746190244 0.412800629
[22] -0.388417586 -0.202330250 2.643844846 0.486020881 0.546572608 0.296103739 0.211759239
[29] 1.564742022 -0.648982423 0.018343735 0.331200438 0.469972889 -0.238855454 1.654410637
[36] 0.273247011 -1.071088939 -0.848405608 -0.127589553 -0.617698635 -1.388631344 0.009807972
[43] -0.179299439 2.310902142 0.964434329 0.911025079 0.539405024 0.396191029 1.194989355
[50] -0.191333364 0.294093475 0.343087452 0.936413697 -1.111776497 0.917297671 -0.472013168
[57] 1.139136757 0.447037643 -0.132862854 0.920285224 0.467461809 1.461597714 -0.765009390
[64] 0.754972616 2.321174325 -0.802375570 0.668474997 0.246218868 -0.189907791 -0.389410318
[71] -0.691435054 -0.869735489 0.578819965 0.693310427 1.202624968 0.276861500 0.216298711
[78] 0.182300444 -0.038493102 -0.693806352 0.426144124 -0.047031392 0.481128044 -0.534493420
[85] 0.432347342 -0.704107413 1.037099071 -0.735772798 1.013443704 0.073244111 1.086505438
[92] -0.436003376 -0.634971143 0.521737335 0.678135522 -0.334026303 -0.663015093 1.081133166
[99] 0.578612965 -0.828046011

> E = rnorm(100)
> print(E)
[1] -0.291191968 -2.638856192 -0.641337403 1.270158583 0.731393983 -1.526897905 -1.047261087
[8] 0.269386742 0.870344638 1.652005435 -0.239205373 -1.659932524 -0.229642418 -1.681208170
[15] 1.805610377 -1.499714250 -0.750717640 2.230034278 -0.308857060 1.685940568 1.057690658
[22] -0.859116952 -1.496276314 0.717054978 0.015507831 0.514452782 -0.817393772 0.850433080
[29] 0.151039483 0.110196472 0.887531898 -3.139001113 -0.588382566 -1.197520957 -0.027254108
[36] -0.112372991 -0.118370905 -0.068722638 -0.009291501 0.828773758 0.897427919 -0.100192455
[43] 0.821919958 -1.639731500 0.095978863 -1.043410715 -0.947350283 0.041113225 -0.525870912
[50] 3.158187532 -0.682065084 -1.164191983 -0.036394897 -1.221653419 1.713555298 -0.577989522
[57] -1.332304516 0.373853174 0.280026949 -2.016144126 0.609746725 1.021660687 0.696618578
[64] 0.839436304 0.095655024 0.629825305 -0.636449170 0.324710724 1.109422328 0.472699645
[71] 0.440972833 -1.134264828 0.593960109 -0.742446841 0.368282707 0.566031099 0.595243755
[78] -1.961764002 -0.501006504 -1.823720958 0.963744292 -0.585410127 -0.828554735 -0.227558098
[85] 2.440124798 0.559880696 -0.526942037 0.082318847 -0.511886264 0.662005227 0.064772966
[92] -0.180939483 0.194524969 -0.095734883 0.066424444 -1.354347279 0.951039771 0.089897617
[99] 1.338292854 0.796837652
```

b)

```
> Y = 3 + 5*X + 4*X^2 + 1*X^3 + E
> print(Y)
[1] 24.6721501 9.9372536 1.1401586 18.6822370 22.8844033 24.6934981 5.0333949 4.5165812
[9] 1.6802054 0.9336600 3.1949358 11.6902884 1.7885788 3.4017281 1.6452386 2.6881952
[17] 24.1076222 14.3993634 1.5933002 9.3690480 5.7175668 1.6449421 2.2459965 62.6036447
[25] 6.4334542 7.0391403 4.9170855 4.2445650 24.4347156 1.0440942 3.1511236 4.9955788
[33] 6.2505458 2.0861037 26.7295770 4.7257564 1.0095364 1.0085213 2.4472225 1.0353361
[41] 1.1528509 3.0780389 2.2018424 48.0143746 12.3352944 11.6088536 6.9491828 5.6619911
[49] 16.5337417 1.9482560 4.6834387 5.2881887 12.0009792 1.0696424 11.8430433 1.4122298
[57] 15.2808862 6.1554456 2.3536081 11.4338485 6.2342398 21.8950365 0.7687583 9.4028611
[65] 48.6121087 1.0036843 8.4951871 4.4426273 2.2761267 1.6890510 1.1287273 1.0382720
[73] 7.3744009 8.8853387 16.4509398 4.8366128 4.1861227 4.0902014 2.7443529 1.1455625
[81] 5.7093421 2.7199172 6.3427651 1.3697124 6.0954730 1.1766829 13.7648439 1.0948876
[89] 13.1135924 3.3990015 14.3148613 1.4760973 1.3360798 6.9702795 8.4383459 1.7155002
[97] 1.0659152 14.4006120 7.4255756 1.1393767
```

c) Best Subset model

```
bestsubset = 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 = dataX, nvmax =
10)

bestsubsetSummary = summary(bestsubset)
```

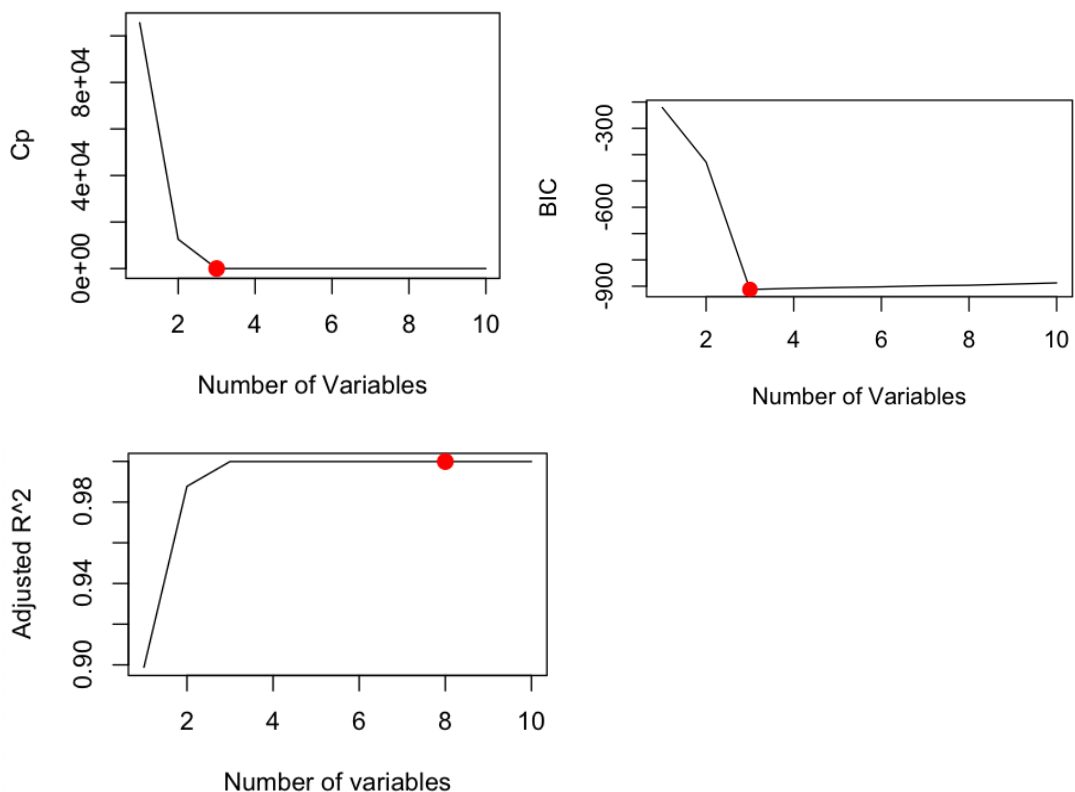
	X	I(X^2)	I(X^3)	I(X^4)	I(X^5)	I(X^6)	I(X^7)	I(X^8)	I(X^9)	I(X^10)
1	( 1 )	11 11 11 11	11 11 11	11 11	11 11	11 11	11 11	11 11	11 11	11 11
2	( 1 )	11 11 11 11	11 11	11 11	11 11	11 11	11 11	11 11	11 11	11 11
3	( 1 )	11 11 11 11	11 11 11	11 11	11 11	11 11	11 11	11 11	11 11	11 11
4	( 1 )	11 11 11 11	11 11 11	11 11 11	11 11	11 11	11 11	11 11	11 11	11 11
5	( 1 )	11 11 11 11	11 11 11	11 11 11	11 11	11 11	11 11	11 11 11	11 11	11 11
6	( 1 )	11 11 11 11	11 11 11	11 11 11	11 11	11 11 11	11 11 11	11 11	11 11	11 11
7	( 1 )	11 11 11 11	11 11 11	11 11 11	11 11	11 11 11	11 11	11 11	11 11 11	11 11
8	( 1 )	11 11 11 11	11 11 11	11 11	11 11 11	11 11 11	11 11 11	11 11 11	11 11	11 11
9	( 1 )	11 11 11 11	11 11 11	11 11 11	11 11 11	11 11 11	11 11	11 11 11	11 11 11	11 11
10	( 1 )	11 11 11 11	11 11 11	11 11 11	11 11 11	11 11 11	11 11 11	11 11 11	11 11 11	11 11

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

```
plot(bestsubsetSummary$cp, xlab = "Number of Variables", ylab = "Cp",
     type = "l")
points(which.min(bestsubsetSummary$cp),
       bestsubsetSummary$cp[which.min(bestsubsetSummary$cp)], col = "red",
       cex = 2, pch = 20)
```

```
plot(bestsubsetSummary$bic, xlab = "Number of Variables", ylab =
     "BIC", type = "l")
points(which.min(bestsubsetSummary$bic),
       bestsubsetSummary$bic[which.min(bestsubsetSummary$bic)], col = "red",
       cex = 2, pch = 20)
```

```
plot(bestsubsetSummary$adjr2, xlab = "Number of Variables", ylab =
     "Adjusted R^2", type = "l")
points(which.max(bestsubsetSummary$adjr2),
       bestsubsetSummary$adjr2[which.max(bestsubsetSummary$adjr2)], col =
       "red", cex = 2, pch = 20)
```



Cp - pick the 3-variables model, Bic - pick the 3 variables model, and Adj R<sup>2</sup> - pick the 8 variables model

```
> coef(bestsubset, which.max(bestsubsetSummary$adjr2))
```

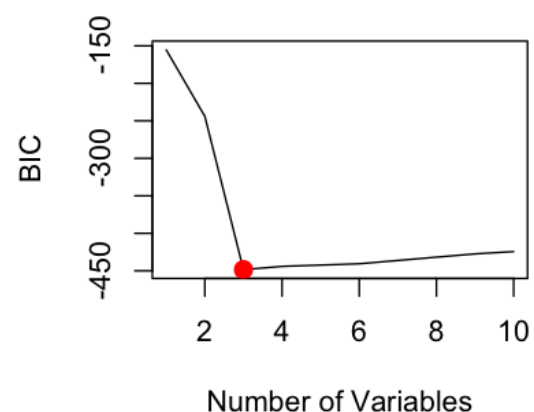
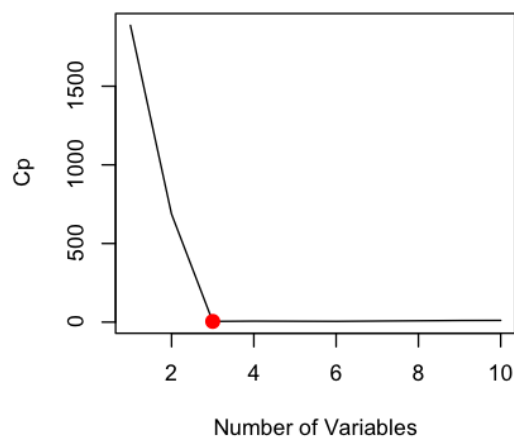
(Intercept)	X	I(X^2)	I(X^3)	I(X^4)	I(X^8)	I(X^10)
2.616881878	4.827080683	5.429391881	1.058400609	-0.723212109	0.037286705	-0.003569669

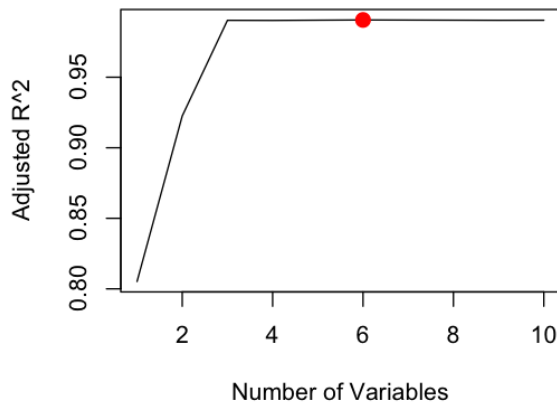
d) Forward model

```
plot(forwardSummary$cp, xlab = "Number of Variables", ylab = "Cp",
     type = "l")
points(which.min(forwardSummary$cp),
       forwardSummary$cp[which.min(forwardSummary$cp)], col = "red", cex =
       2, pch = 20)

plot(forwardSummary$bic, xlab = "Number of Variables", ylab = "BIC",
     type = "l")
points(which.min(forwardSummary$bic),
       forwardSummary$bic[which.min(forwardSummary$bic)], col = "red", cex =
       2, pch = 20)

plot(forwardSummary$adjr2, xlab = "Number of Variables", ylab =
     "Adjusted R^2", type = "l")
points(which.max(forwardSummary$adjr2),
       forwardSummary$adjr2[which.max(forwardSummary$adjr2)], col = "red",
       cex = 2, pch = 20)
```





Cp - pick the 3 variables model, Bic - pick the 3 variables model, and Adj R<sup>2</sup> - pick the 6 variables model

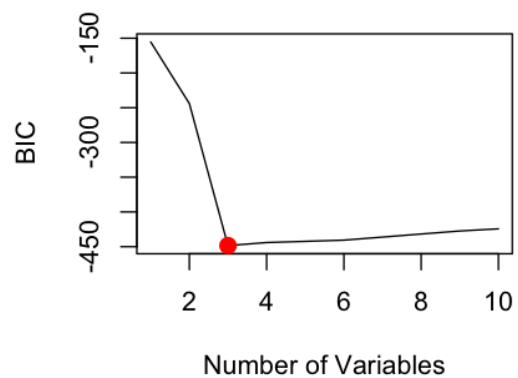
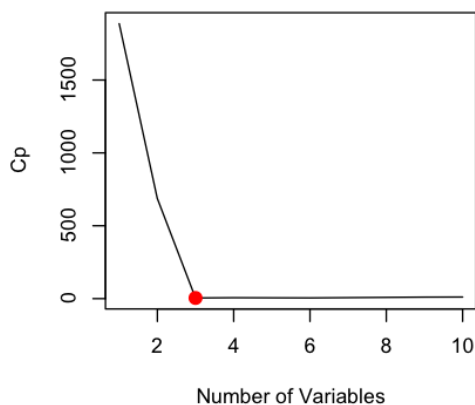
```
> coef(forward, which.max(forwardSummary$adjr2))
(Intercept)      X      I(X^2)      I(X^3)      I(X^4)      I(X^6)      I(X^10)
2.592789735  4.833287398  5.663984303  1.053787063 -1.094339766  0.191123540 -0.001228671
```

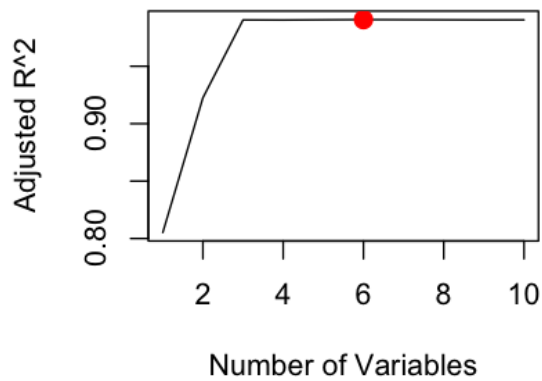
Backward model

```
plot(backwardSummary$cp, xlab = "Number of Variables", ylab = "Cp",
     type = "l")
points(which.min(backwardSummary$cp),
       backwardSummary$cp[which.min(backwardSummary$cp)], col = "red", cex =
       2, pch = 20)

plot(backwardSummary$bic, xlab = "Number of Variables", ylab = "BIC",
     type = "l")
points(which.min(backwardSummary$bic),
       backwardSummary$bic[which.min(backwardSummary$bic)], col = "red", cex =
       2, pch = 20)

plot(backwardSummary$adjr2, xlab = "Number of Variables", ylab =
     "Adjusted R^2", type = "l")
points(which.max(backwardSummary$adjr2),
       backwardSummary$adjr2[which.max(backwardSummary$adjr2)], col = "red",
       cex = 2, pch = 20)
```





Cp - pick the 3 variables model, Bic - pick the 3 variables model, and Adj R<sup>2</sup> - pick the 6 variables model

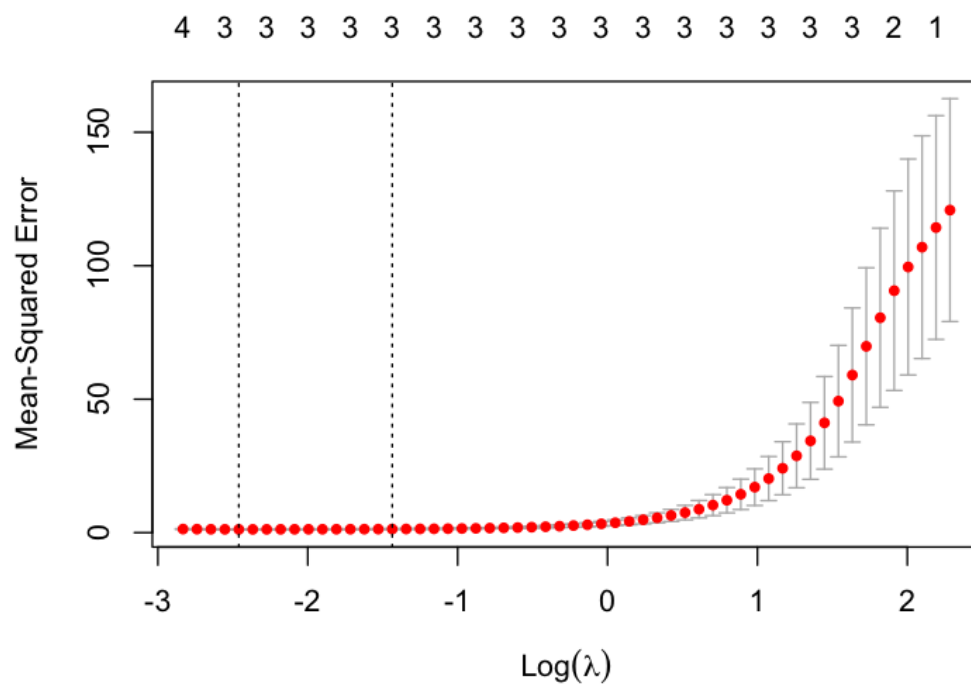
```
> coef(backward, which.max(backwardSummary$adjr2))
(Intercept)      X      I(X^2)      I(X^5)      I(X^6)      I(X^7)      I(X^8)
 2.77591030  5.31679015  4.51613281  0.51143911 -0.12213077 -0.06768933  0.02116181
```

e) Lasso method

```
library(glmnet)

lasso = model.matrix(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 = dataX)[-1]

cvLasso = cv.glmnet(lasso, Y, alpha=1)
plot(cvLasso)
```



The best lambda is:

```
> bestLambda = crossvalLasso$lambda.min
> print(bestLambda)
[1] 0.07092195
```

So we run the model using this value:

```
> lassoBestLambda = glmnet(lasso, Y, alpha = 1)
> predict(lassoBestLambda, s = bestLambda, type = "coefficients")[1:11,]
(Intercept)      X      I(X^2)      I(X^3)      I(X^4)      I(X^5)      I(X^6)      I(X^7)
  2.979571  4.753819  3.994324  1.049389  0.000000  0.000000  0.000000  0.000000
      I(X^8)      I(X^9)      I(X^10)
  0.000000  0.000000  0.000000
```

The lasso methods picks X, X<sup>2</sup>, and X<sup>3</sup> as variables for the model.

f) Best subset model:

```
Y = 3 + 10*X^7 + E

datanewX = data.frame(X,Y)

bestsubset2 = 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 = datanewX, nvmax = 10)

bestsubsetSummary2 = summary(bestsubset2)

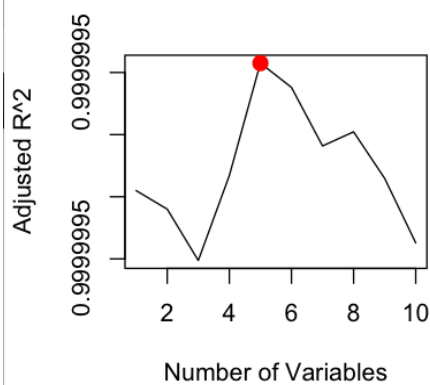
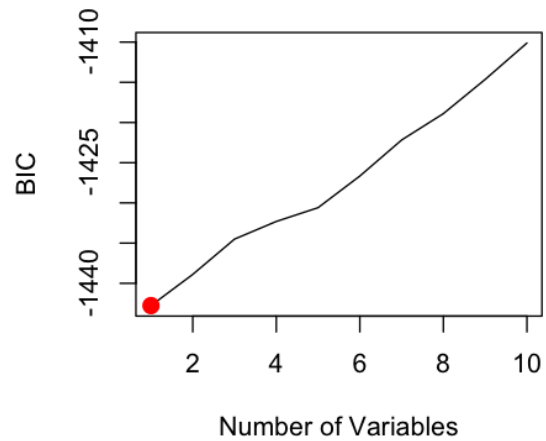
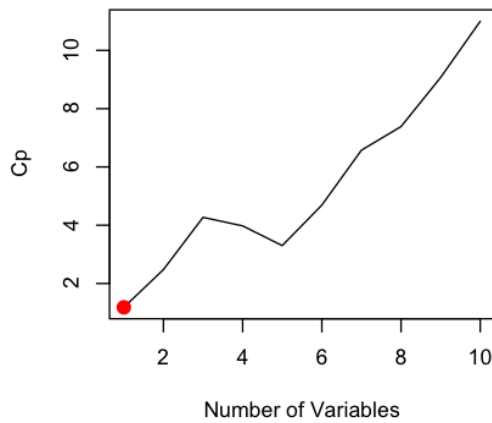
par(mfrow = c(2,2))

plot(bestsubsetSummary2$cp, xlab = "Number of Variables", ylab = "Cp", type =
  "l")
points(which.min(bestsubsetSummary2$cp),
  bestsubsetSummary2$cp[which.min(bestsubsetSummary2$cp)], col = "red", cex =
  2, pch = 20)

plot(bestsubsetSummary2$bic, xlab = "Number of Variables", ylab = "BIC", type
  = "l")
points(which.min(bestsubsetSummary2$bic),
  bestsubsetSummary2$bic[which.min(bestsubsetSummary2$bic)], col = "red", cex
  = 2, pch = 20)

plot(bestsubsetSummary2$adjr2, xlab = "Number of Variables", ylab = "Adjusted
  R^2", type = "l")
points(which.max(bestsubsetSummary2$adjr2),
  bestsubsetSummary2$adjr2[which.max(bestsubsetSummary2$adjr2)], col = "red",
  cex = 2, pch = 20)
```





Cp - pick the 1 variable model, Bic - pick the 1 variable model, and Adj R<sup>2</sup> - pick the 5 variables model

```
> coef(bestsubset2, 1)
(Intercept)      I(X^7)
  2.955573    10.000653
> coef(bestsubset2, 2)
(Intercept)      X      I(X^7)
  2.9646268  -0.1021027  10.0009142
> coef(bestsubset2, 4)
(Intercept)      I(X^2)      I(X^4)      I(X^6)      I(X^7)
  2.73725660   0.78060829  -0.35892187   0.03864099   9.99875833
```

BIC picks the most accurate 1-variable model.

Lasso Method:

```
lasso2 = model.matrix(newY ~ 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 = datanewX)[-1]

crossvallasso2 = cv.glmnet(lasso2, newY, alpha=1)
> bestLambda2 = crossvallasso2$lambda.min
> print(bestLambda2)
[1] 0.07092195
```

```
> lassoBestLambda2 = glmnet(lasso2, newY, alpha = 1)
> predict(lassoBestLambda2, s = bestLambda2, type = "coefficients")[1:11,]
(Intercept)      X      I(X^2)      I(X^3)      I(X^4)      I(X^5)      I(X^6)      I(X^7)
  8.079787    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000    9.709129
      I(X^8)      I(X^9)      I(X^10)
    0.000000    0.000000    0.000000
-
```

Lasso picks an accurate 1 variable model, but the intercept is off by about 5 units.

#### Question 4:

- Split the College data into train and test in 7:3 ratio.
- 

```
call:
lm(formula = Apps ~ ., data = train)

Coefficients:
(Intercept)  PrivateYes      Accept      Enroll      Top10perc      Top25perc      F.Undergrad
 -638.64835   -596.53900     1.32983    -0.27428     52.60987    -17.61480      0.04737
P.Undergrad  Outstate    Room.Board      Books      Personal      PhD      Terminal
  0.04116    -0.07459     0.18067    -0.09611     0.02455    -10.13790     -5.64889
S.F.Ratio   perc.alumni      Expend      Grad.Rate
 23.73129    -6.46646     0.12574     11.01750
```

```
> err.lm = mean((test$Apps - pred.lm)^2)
> err.lm
[1] 1409723
```

Least Square Test Error on Linear Model is 1409723

- Ridge Regression Best Lambda:

```
> ridge.lambda
[1] 28.48036
```

Ridge Regression minimum Test Error:

```
> err.ridge
[1] 1560730
```

- Lasso Best Lambda:

```
> lasso.lambda
[1] 8.111308
```

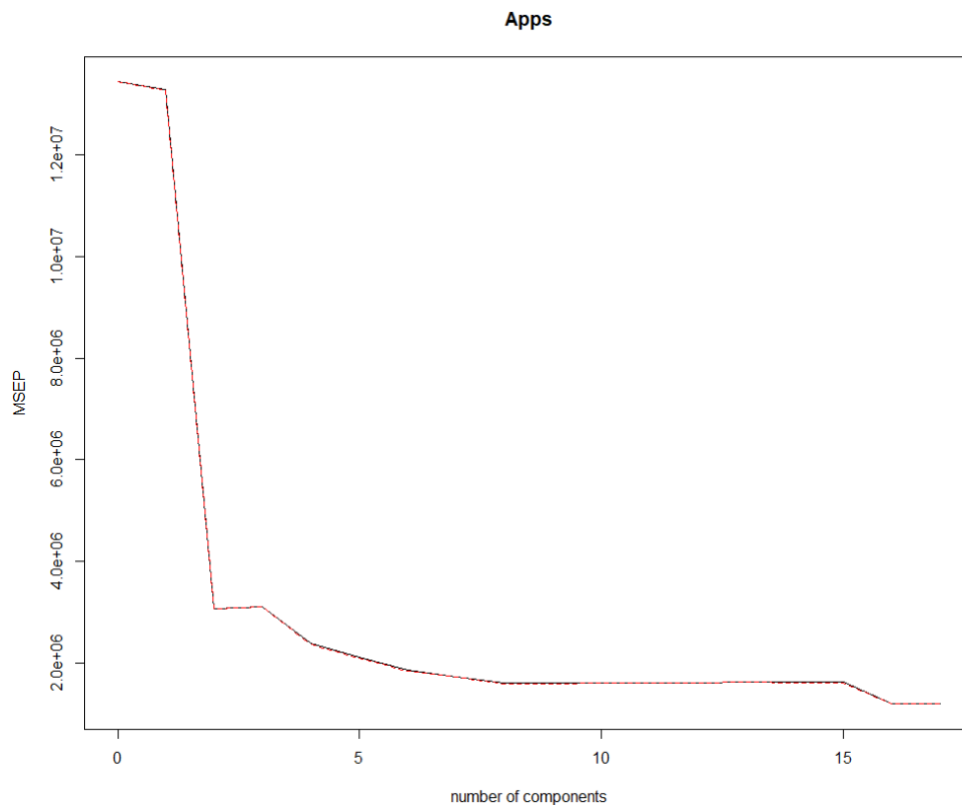
Lasso minimum Test Error:

```
> err.lasso
[1] 1448810
```

Lasso Non-zero coefficient estimates:

```
> coef.lasso[coef.lasso != 0]
(Intercept)  PrivateYes      Accept      Top10perc      Top25perc      F.Undergrad      P.Undergrad
 -755.75876508 -582.61892808     1.28023352    46.91699381    -13.23745442      0.01776872      0.03662615
      Outstate    Room.Board      Books      Personal      PhD      Terminal      S.F.Ratio
 -0.06386837    0.16976881    -0.01678368     0.01062464     -8.59500784    -5.53121221    19.99611424
perc.alumni      Expend      Grad.Rate
 -7.15656228    0.12107184     9.77722658
```

- PCR plot for value of M and MSEP:



Visualizing the validation plot shows that MSEP is minimum for **M=16**.

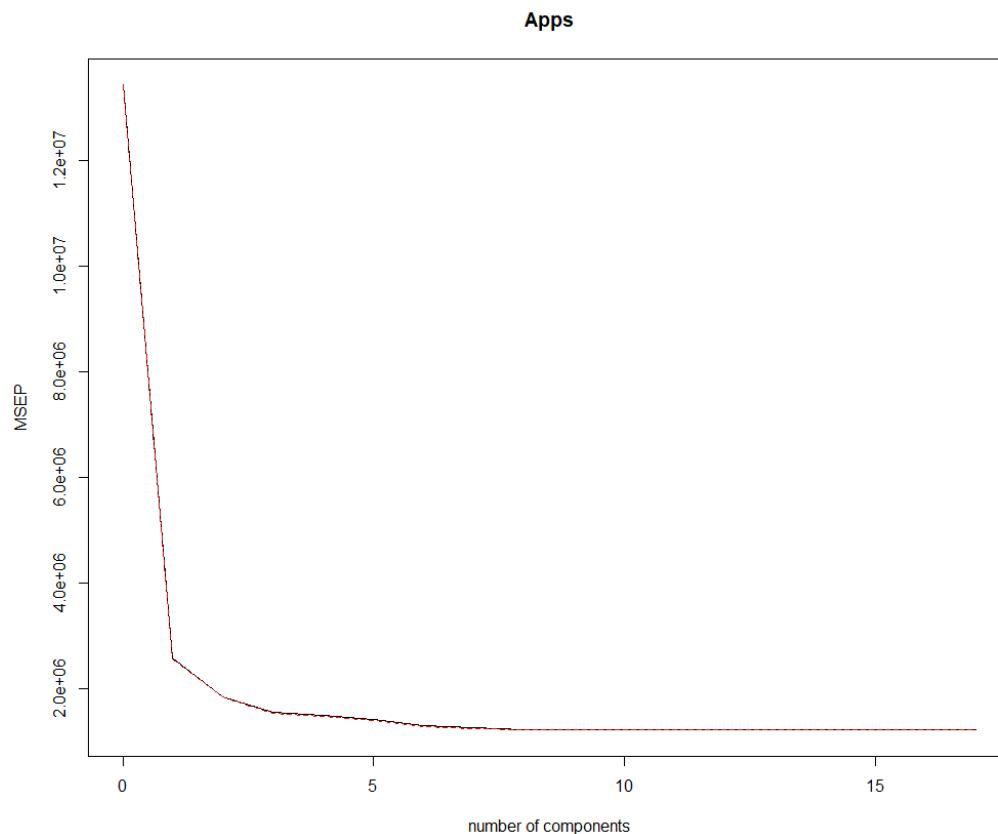
Choosing M=16, yields PCR Test Error = 1542134:

```
> err.pcr
[1] 1542134
> summary(fit.pcr)
Data:  X dimension: 543 17
      Y dimension: 543 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 7 comps 8 comps 9 comps
cv          3665    3643    1754    1764    1548    1454    1364    1313    1268    1268
adjcv       3665    3643    1753    1763    1542    1450    1362    1315    1265    1265
      10 comps 11 comps 12 comps 13 comps 14 comps 15 comps 16 comps 17 comps
cv          1270    1273    1273    1278    1277    1276    1098    1098
adjcv       1268    1271    1270    1274    1273    1273    1095    1094

TRAINING: % variance explained
      1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps 9 comps 10 comps
X       32.539  58.51  65.54  70.98  76.18  80.88  84.60  88.00  91.00  93.36
Apps    1.865  77.55  77.67  83.12  85.15  86.79  87.78  88.66  88.66  88.79
      11 comps 12 comps 13 comps 14 comps 15 comps 16 comps 17 comps
X       95.35  97.03  98.13  98.99  99.48  99.84  100.00
Apps    88.80  89.00  89.04  89.06  89.24  91.98  92.16
```

f) PLS plot for value of M and MSEP:



Visualizing the validation plot shows that MSEP is minimum for **M=10**.

Choosing M=10, yields PCR Test Error = 1420562:

```
> err.plsr
[1] 1420562
> summary(fit.plsr)
Data:   X dimension: 543 17
        Y dimension: 543 1
Fit method: kernelppls
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  7 comps  8 comps  9 comps
cv          3665    1606    1351    1237    1205    1148    1116    1101    1096    1094
adjcv       3665    1604    1354    1234    1200    1144    1112    1097    1092    1090
cv          10 comps 11 comps 12 comps 13 comps 14 comps 15 comps 16 comps 17 comps
adjcv       1089    1088    1088    1089    1090    1090    1090    1090
```

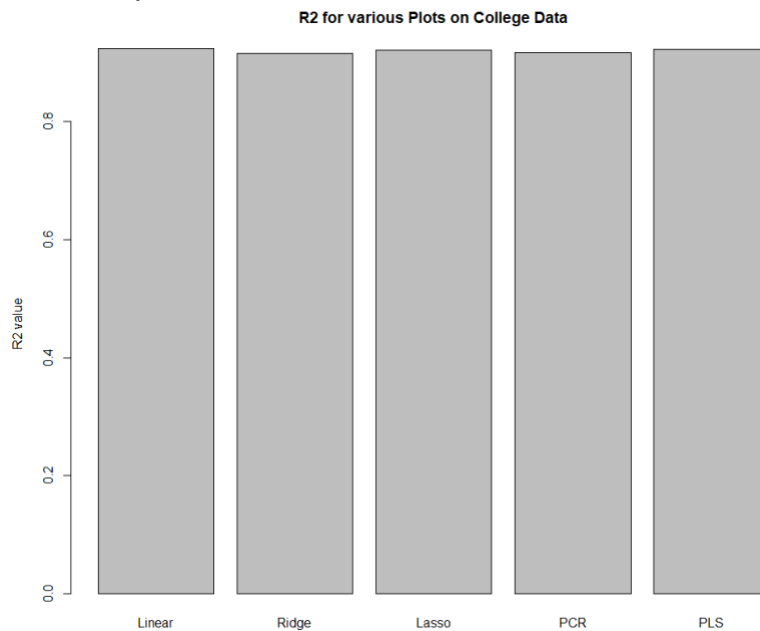
```
TRAINING: % variance explained
      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps  9 comps 10 comps
X        26.10   42.12   63.55   67.17   70.56   74.76   78.01   81.33   84.04   87.16
Apps     81.44   87.06   89.38   90.32   91.37   91.90   92.07   92.11   92.13   92.14
      11 comps 12 comps 13 comps 14 comps 15 comps 16 comps 17 comps
X        89.65   92.05   93.58   94.52   96.94   98.36   100.00
Apps     92.15   92.15   92.15   92.16   92.16   92.16   92.16
```

g)  $R^2$  value for various models is:

Linear Model, Ridge Regression, Lasso, PCR, PLS

```
> r2.all
[1] 0.9240041 0.9158635 0.9218970 0.9168660 0.9234197
```

The boxplot for these values is:



All the values and Boxplot shows that there isn't much difference in the accuracy of various models, and they perform almost the same. All models predict College applications with high accuracy of 91.5-92.4%.

Test Error value for various models:

```
> err.all
Linear Ridge Lasso PCR PLS
1409723 1560730 1448810 1542134 1420562
```

Error is almost the same for all models but with maximum error for Ridge Regression(1560730) and minimum error for Linear Model(1409723).