HW5_Ghoshal

Gourav Ghoshal

November 9, 2018

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).

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.4.2
```

```
head(College)
```

```
##                              Private Apps Accept Enroll Top10perc
## Abilene Christian University     Yes 1660   1232    721        23
## Adelphi University               Yes 2186   1924    512        16
## Adrian College                   Yes 1428   1097    336        22
## Agnes Scott College              Yes  417    349    137        60
## Alaska Pacific University        Yes  193    146     55        16
## Albertson College                Yes  587    479    158        38
##                              Top25perc F.Undergrad P.Undergrad Outstate
## Abilene Christian University        52        2885         537     7440
## Adelphi University                  29        2683        1227    12280
## Adrian College                      50        1036          99    11250
## Agnes Scott College                 89         510          63    12960
## Alaska Pacific University           44         249         869     7560
## Albertson College                   62         678          41    13500
##                              Room.Board Books Personal PhD Terminal
## Abilene Christian University       3300   450     2200  70       78
## Adelphi University                 6450   750     1500  29       30
## Adrian College                     3750   400     1165  53       66
## Agnes Scott College                5450   450      875  92       97
## Alaska Pacific University          4120   800     1500  76       72
## Albertson College                  3335   500      675  67       73
##                              S.F.Ratio perc.alumni Expend Grad.Rate
## Abilene Christian University      18.1          12   7041        60
## Adelphi University                12.2          16  10527        56
## Adrian College                    12.9          30   8735        54
## Agnes Scott College                7.7          37  19016        59
## Alaska Pacific University         11.9           2  10922        15
## Albertson College                  9.4          11   9727        55
```

```
names(College)
```

```
##  [1] "Private"     "Apps"        "Accept"      "Enroll"      "Top10perc"  
##  [6] "Top25perc"   "F.Undergrad" "P.Undergrad" "Outstate"    "Room.Board" 
## [11] "Books"       "Personal"    "PhD"         "Terminal"    "S.F.Ratio"  
## [16] "perc.alumni" "Expend"      "Grad.Rate"
```

```r
dim(College)
```

```
## [1] 777  18
```

```r
sum(is.na(College))
```

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

(a) Perform best subset selection to the data. 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.

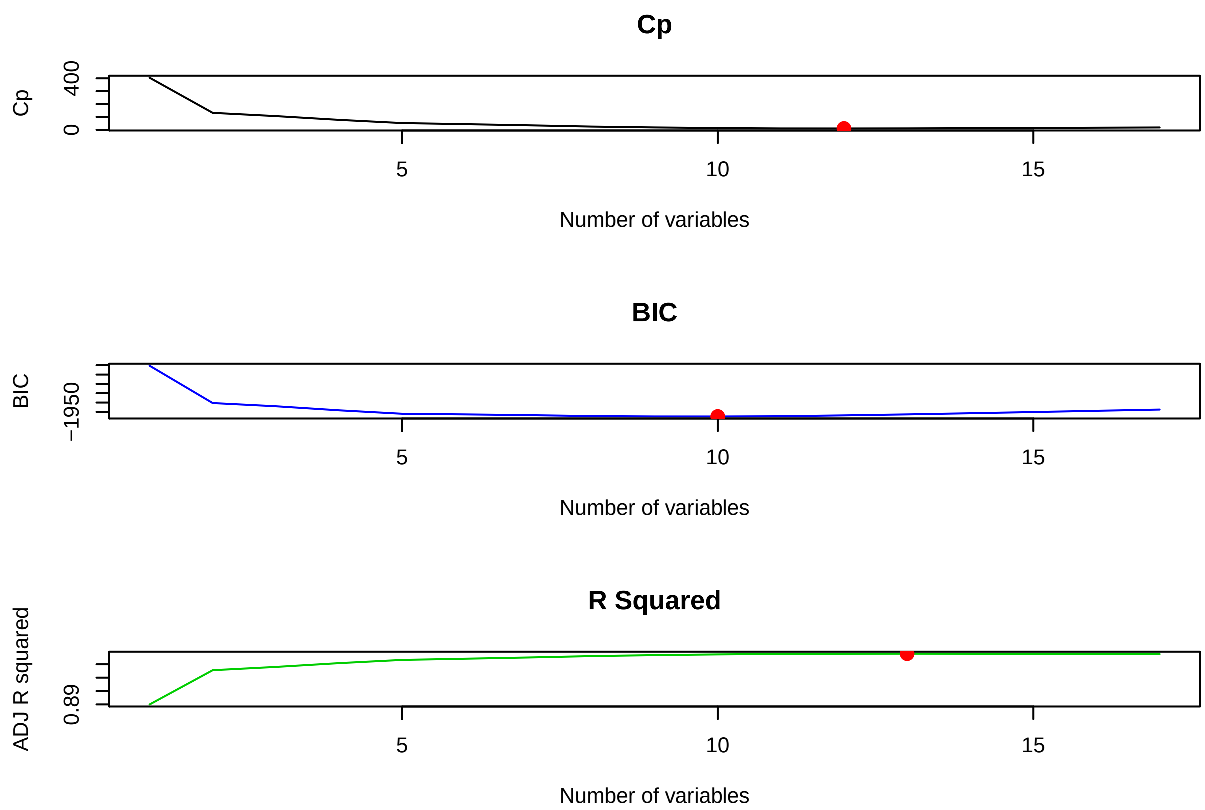
```r
# a) Best subset
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.4.3
```

```r
attach(College)
best_subset_model = regsubsets(Apps ~., College, nvmax = dim(College)[2]-1)
res_summ = summary(best_subset_model)
names(res_summ)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```r
par(mfrow=c(3,1))
plot(res_summ$cp,type='l',
     main = 'Cp',xlab = 'Number of variables', ylab = 'Cp')
points(which.min(res_summ$cp), res_summ$cp[which.min(res_summ$cp)],col=2,
       cex=2, pch=20)
plot(res_summ$bic,type='l',col = 4,
     main = 'BIC', xlab = 'Number of variables', ylab = 'BIC')
points(which.min(res_summ$bic), res_summ$bic[which.min(res_summ$bic)],col=2,
       cex=2, pch=20)
plot(res_summ$adjr2,type='l',col = 3,
     main = 'R Squared', xlab = 'Number of variables', ylab = 'ADJ R squared')
points(which.max(res_summ$adjr2), res_summ$adjr2[which.max(res_summ$adjr2)],col=2,
       cex=2, pch=20)
```




```
# Cp ==> 12, BIC==> 10, R-squared ==> 17 (since not adj R-square)
which.min(res_summ$cp)
```

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

```
which.min(res_summ$bic)
```

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

```
which.max(res_summ$adjr2)
```

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

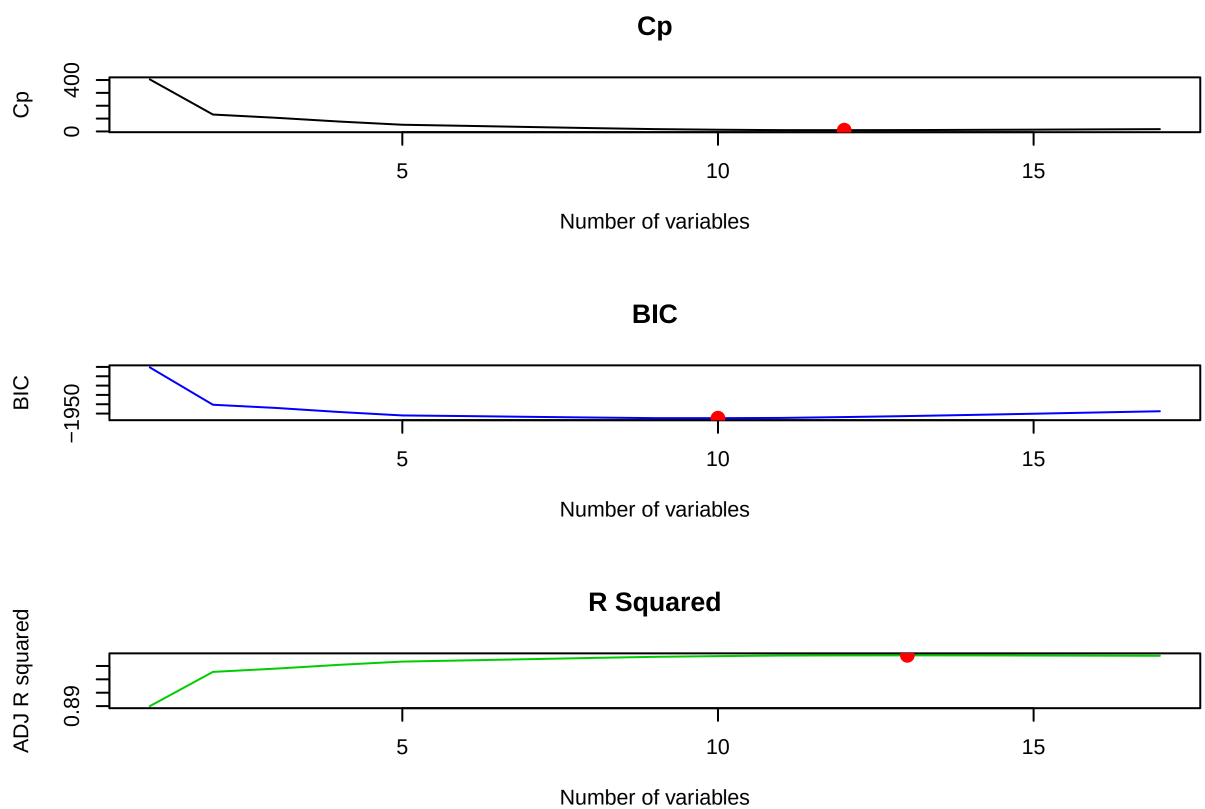
(b) Repeat (a) using forward stepwise selection and backwards stepwise selection. How does your answer compare to the results in (a)?

```
# b) Forward and Backward selection
## Forward
f_subset_model = regsubsets(Apps ~., College, nvmax = dim(College)[2]-1,
                               method = 'forward')
f_summ = summary(f_subset_model)
names(f_summ)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```
par(mfrow=c(3,1))
plot(f_summ$cp,type='l',
     main = 'Cp',xlab = 'Number of variables', ylab = 'Cp')
```

```
points(which.min(f_summ$cp), f_summ$cp[which.min(f_summ$cp)],col=2,
       cex=2, pch=20)
plot(f_summ$bic,type='l',col = 4,
     main = 'BIC', xlab = 'Number of variables', ylab = 'BIC')
points(which.min(f_summ$bic), f_summ$bic[which.min(f_summ$bic)],col=2,
       cex=2, pch=20)
plot(f_summ$adjr2,type='l',col = 3,
     main = 'R Squared', xlab = 'Number of variables', ylab = 'ADJ R squared')
points(which.max(f_summ$adjr2), f_summ$adjr2[which.max(f_summ$adjr2)],col=2,
       cex=2, pch=20)
```




```
# Cp ==> 12, BIC==> 10, R-squared ==> 17 (since not adj R-square)
which.min(f_summ$cp)
```

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

```
which.min(f_summ$bic)
```

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

```
which.max(f_summ$adjr2)
```

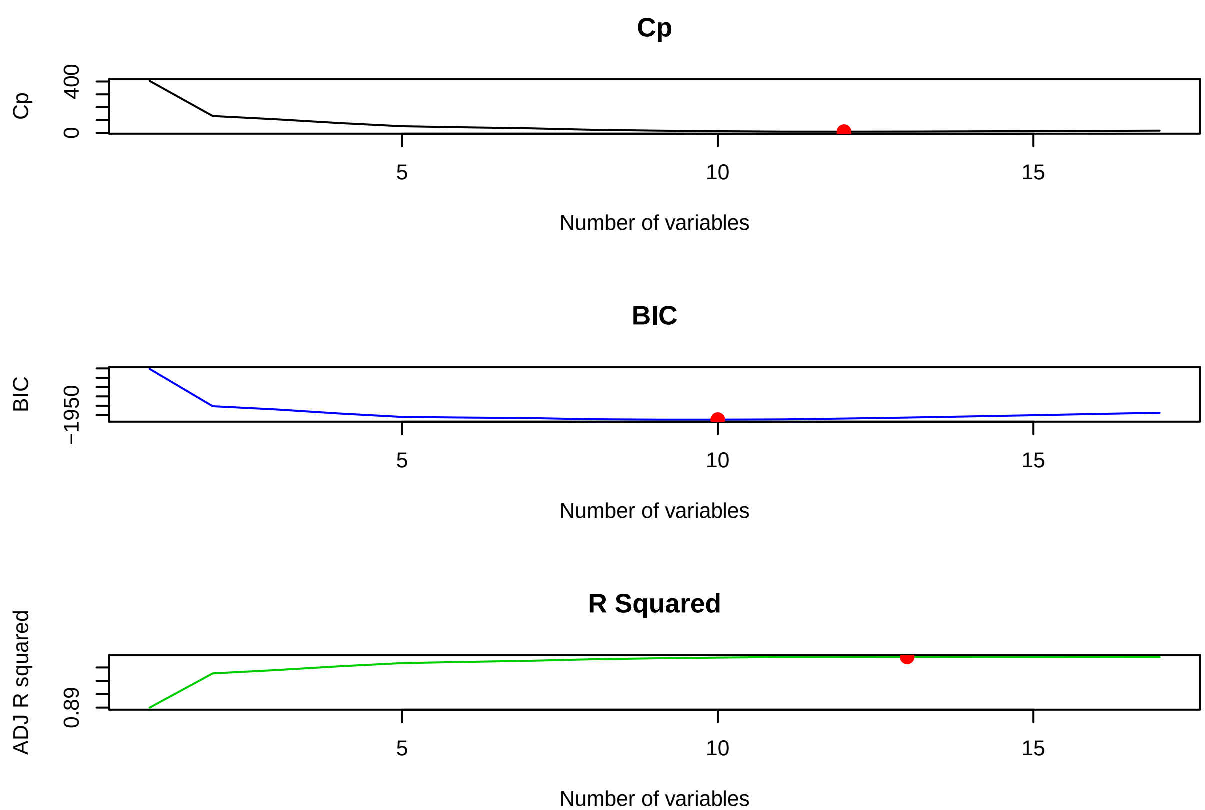
```
## [1] 13
```

```
#backward
b_subset_model = regsubsets(Apps ~., College, nvmax = dim(College)[2]-1,
                            method = 'backward')
b_summ = summary(b_subset_model)
```

```r
names(b_summ)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

```r
par(mfrow=c(3,1))
plot(b_summ$cp,type='l',
     main = 'Cp',xlab = 'Number of variables', ylab = 'Cp')
points(which.min(b_summ$cp), b_summ$cp[which.min(b_summ$cp)],col=2,
       cex=2, pch=20)
plot(b_summ$bic,type='l',col = 4,
     main = 'BIC', xlab = 'Number of variables', ylab = 'BIC')
points(which.min(b_summ$bic), b_summ$bic[which.min(b_summ$bic)],col=2,
       cex=2, pch=20)
plot(b_summ$adjr2,type='l',col = 3,
     main = 'R Squared', xlab = 'Number of variables', ylab = 'ADJ R squared')
points(which.max(b_summ$adjr2), b_summ$adjr2[which.max(b_summ$adjr2)],col=2,
       cex=2, pch=20)
```




```r
# Cp ==> 12, BIC==> 10, R-squared ==> 17 (since not adj R-square)
which.min(b_summ$cp)
```

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

```r
which.min(b_summ$bic)
```

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

```r
which.max(b_summ$adjr2)
```

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

Part (B) gives the same choices as part (A)

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

```r
#c) Lasso
x = model.matrix(Apps ~., College)[,-2]
y = Apps
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.3
```

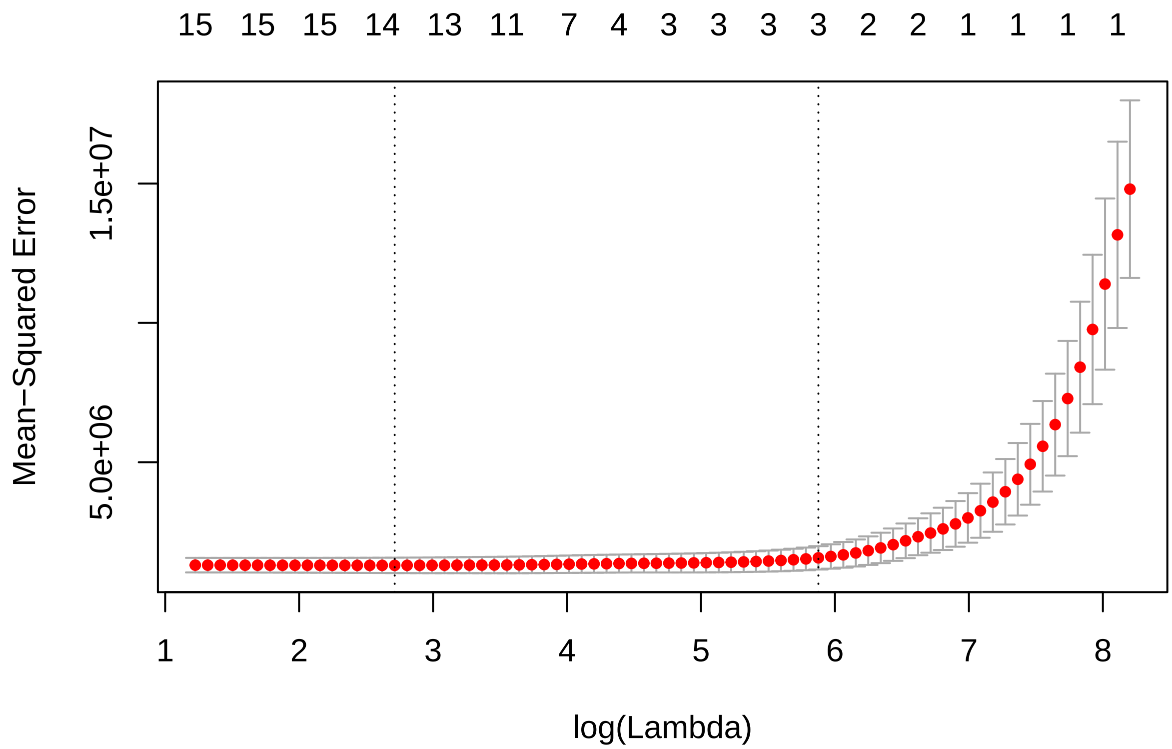
```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 3.4.2
```

```
## Loaded glmnet 2.0-13
```

```r
grid = 10^seq(10,-2,length = 100)
model_lasso = glmnet(x,y,alpha = 1,lambda = grid)
set.seed(42)
cv_lasso = cv.glmnet(x,y,alpha = 1)
plot(cv_lasso)
```

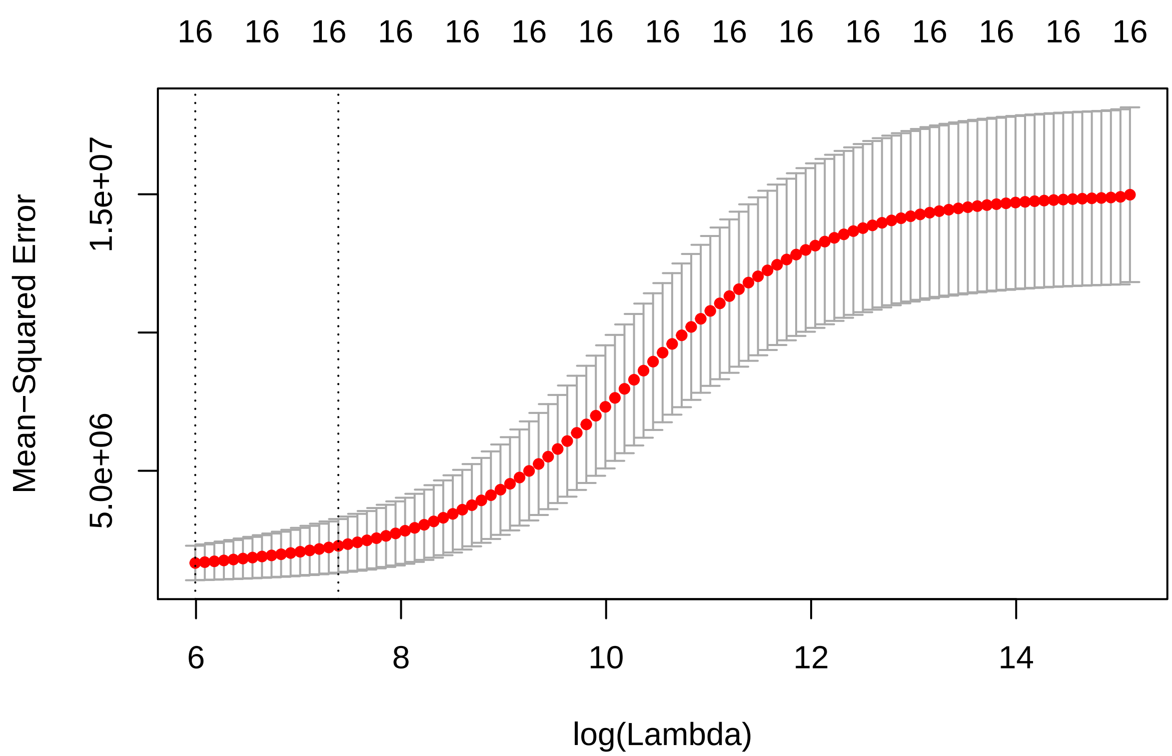



```r
best_lmda_lasso = cv_lasso$lambda.min

predict(model_lasso, type='coefficients', s=best_lmda_lasso)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -998.38928506
## (Intercept)      .
## Accept         1.49980274
## Enroll        -0.26942335
## Top10perc     38.11353044
## Top25perc     -5.67831165
## F.Undergrad    .
## P.Undergrad    0.04077278
## Outstate      -0.08946605
## Room.Board     0.11894501
## Books          .
## Personal       0.01100391
## PhD           -4.73850286
## Terminal      -1.31391498
## S.F.Ratio     15.41836974
## perc.alumni   -2.04515340
## Expend         0.07584338
## Grad.Rate      5.50890148
```

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

```
#d) Ridge
x = model.matrix(Apps ~., College)[,-2]
y = Apps
library(glmnet)
grid = 10^seq(10,-2,length = 100)
model_ridge = glmnet(x,y,alpha = 0,lambda = grid)
set.seed(42)
cv_ridge = cv.glmnet(x,y,alpha = 0)
plot(cv_ridge)
```




```
best_lmda_ridge = cv_ridge$lambda.min

predict(model_ridge, type='coefficients', s=best_lmda_ridge)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept) -1.984049e+03
## (Intercept)  .
## Accept        9.848056e-01
## Enroll        4.912930e-01
## Top10perc     2.480453e+01
## Top25perc     9.576670e-01
```

```
## F.Undergrad  8.599654e-02
## P.Undergrad  3.326895e-02
## Outstate    -4.095601e-02
## Room.Board   1.797027e-01
## Books        1.072766e-01
## Personal    -3.852404e-03
## PhD         -1.948046e+00
## Terminal    -2.800965e+00
## S.F.Ratio    2.074290e+01
## perc.alumni -1.045259e+01
## Expend       7.721708e-02
## Grad.Rate    1.047647e+01
```

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

```r
# e) train- test splitting and model fitting
train_indx = sample(1:nrow(x), nrow(x)/2)
test_indx = (-train_indx)
```

i. Fit the best models obtained in the best subset selection (according to Cp, BIC or adjusted R2) to the training set, and report the test error obtained.

```r
best_subset_model = regsubsets(Apps ~., College[train_indx,],
                               nvmax = dim(College)[2]-1)
res_summ = summary(best_subset_model)
which.min(res_summ$cp)
```

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

```r
which.min(res_summ$bic)
```

```
## [1] 5
```

```r
which.max(res_summ$adjr2)
```

```
## [1] 14
```

```r
X = model.matrix(Apps~., College[test_indx,])
coef_cp = coef(best_subset_model, id = which.min(res_summ$cp) )
coef_bic = coef(best_subset_model, id = which.min(res_summ$bic) )
coef_adj_rsq = coef(best_subset_model, id = which.min(res_summ$adjr2) )

pred_cp = X[,names(coef_cp)]%*%coef_cp
pred_bic = X[,names(coef_bic)]%*%coef_bic
pred_adj_rsq = X[,names(coef_adj_rsq)]%*%coef_adj_rsq

error_cp = mean((Apps[test_indx]-pred_cp)^2)
error_bic = mean((Apps[test_indx]-pred_bic)^2)
error_adj_rsq = mean((Apps[test_indx]-pred_adj_rsq)^2)

print(error_cp)
```

```
## [1] 1562163
```

```r
print(error_bic)
```

```
## [1] 1599488
```

```r
print(error_adj_rsq)
```

```
## [1] 2071866
```

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

```r
best_lmda_lasso = cv_lasso$lambda.min   # 15.07769
pred_lasso = predict(model_lasso, s=best_lmda_lasso, x)
mean((pred_lasso-y)^2) # 1102253
```

```
## [1] 1102253
```

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

```r
best_lmda_ridge = cv_ridge$lambda.min   # 400.4766
pred_ridge = predict(model_ridge, s=best_lmda_ridge, x)
mean((pred_ridge-y)^2) # 1405339
```

```
## [1] 1405339
```

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

Best model is = Best Subset Selection with Cp

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).

```r
library(ISLR)
head(Carseats)
```

```
##   Sales CompPrice Income Advertising Population Price ShelveLoc Age
## 1  9.50       138     73          11        276   120       Bad  42
## 2 11.22       111     48          16        260    83      Good  65
## 3 10.06       113     35          10        269    80    Medium  59
## 4  7.40       117    100           4        466    97    Medium  55
## 5  4.15       141     64           3        340   128       Bad  38
## 6 10.81       124    113          13        501    72       Bad  78
##   Education Urban  US
## 1        17   Yes Yes
## 2        10   Yes Yes
## 3        12   Yes Yes
## 4        14   Yes Yes
## 5        13   Yes  No
## 6        16    No Yes
```

```r
attach(Carseats)
```

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

```r
train_indx = sample(1:nrow(Carseats), nrow(Carseats)/2)
test_indx = (-train_indx)

train = Carseats[c(train_indx),]
test = Carseats[test_indx,]
```

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

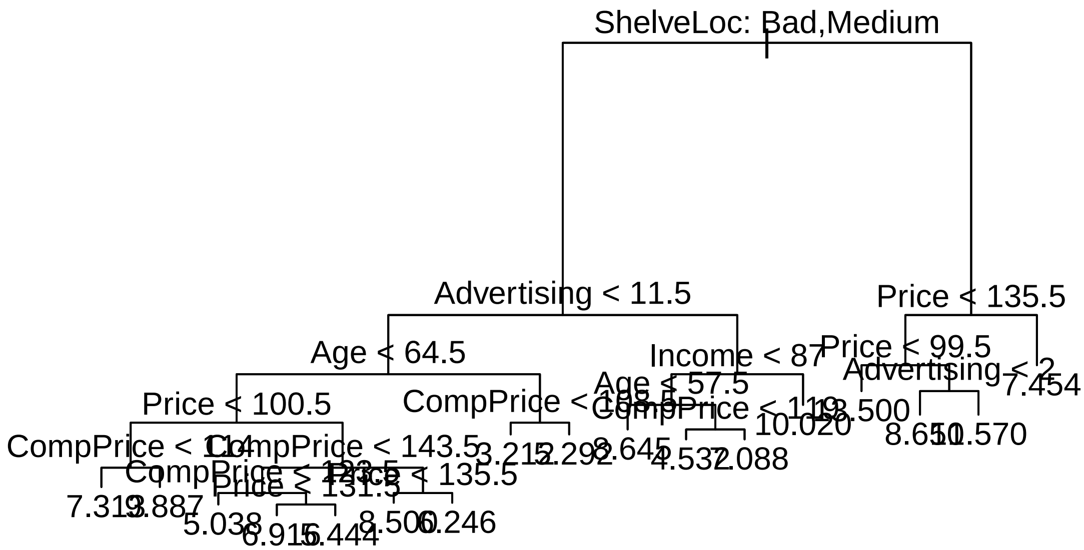
```r
library(tree)
```

```
## Warning: package 'tree' was built under R version 3.4.3
```

```r
reg_tree = tree(Sales~., Carseats[train_indx,])
summary(reg_tree)
```

```
##
## Regression tree:
## tree(formula = Sales ~ ., data = Carseats[train_indx, ])
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Advertising" "Age"         "Price"       "CompPrice"
## [6] "Income"
## Number of terminal nodes:  17
## Residual mean deviance:  2.434 = 445.4 / 183
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -5.08200 -0.96750  0.01808  0.00000  1.04100  4.16600
```

```r
par(mfrow=c(1,1))
plot(reg_tree)
text(reg_tree,pretty = 0)
```

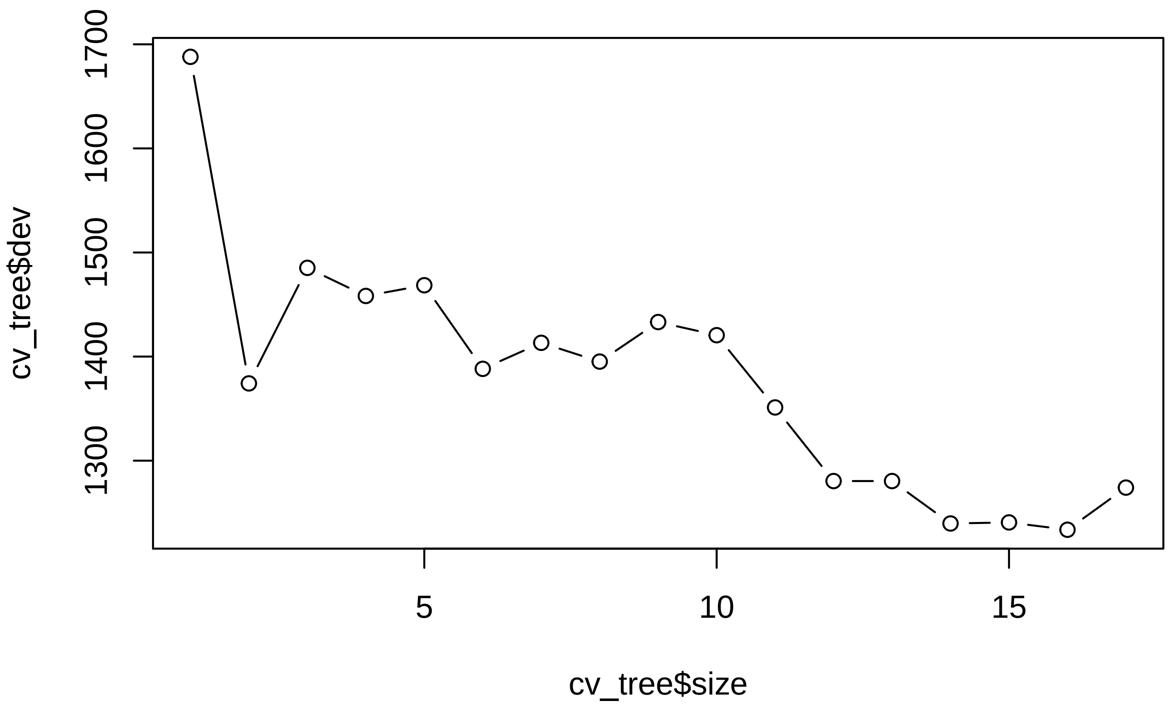



```r
pred_tree = predict(reg_tree, Carseats[test_indx,])
mean((pred_tree-Carseats[test_indx,]$Sales)^2)
```

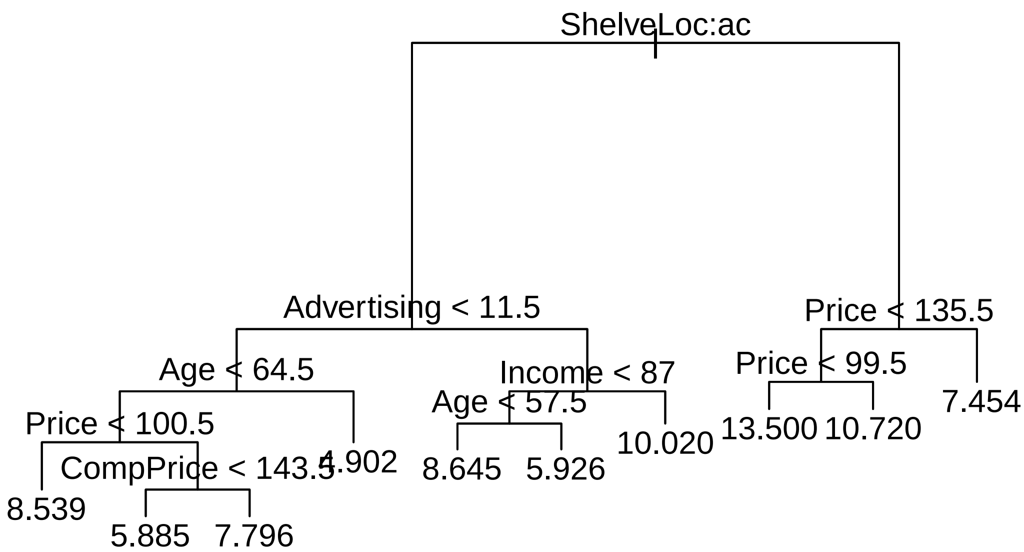
```
## [1] 5.371699
```

(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?

```r
cv_tree = cv.tree(reg_tree)
plot(cv_tree$size, cv_tree$dev, type='b')
```




```r
# There is not much gain after 10
prune_tree = prune.tree(reg_tree, best = 10)
plot(prune_tree)
text(prune_tree)
```




```r
prune_pred_tree = predict(prune_tree, Carseats[test_indx,])
mean((prune_pred_tree-Carseats[test_indx,]$Sales)^2)
```

```
## [1] 5.205643
```

```r
# Yes, pruned tree produces better test results
```

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

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.2

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.4.2
```

```r
library(ipred)
```

```
## Warning: package 'ipred' was built under R version 3.4.2
```

```r
bag_tree = bagging(Sales~., Carseats[train_indx,], coob = T)
pred_bag = predict(bag_tree, Carseats[test_indx,])
mean((pred_bag-Carseats[test_indx,]$Sales)^2)
```

```
## [1] 3.255091
```

```r
varImp(bag_tree)
```

```
##               Overall
## Advertising 1.7237883
## Age         1.6732181
## CompPrice   1.7461440
## Education   0.6413526
## Income      1.2862672
## Population  1.0274530
## Price       2.3206199
## ShelveLoc   1.2002635
## Urban       0.4687182
## US          0.4570515
```

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

```r
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.
```

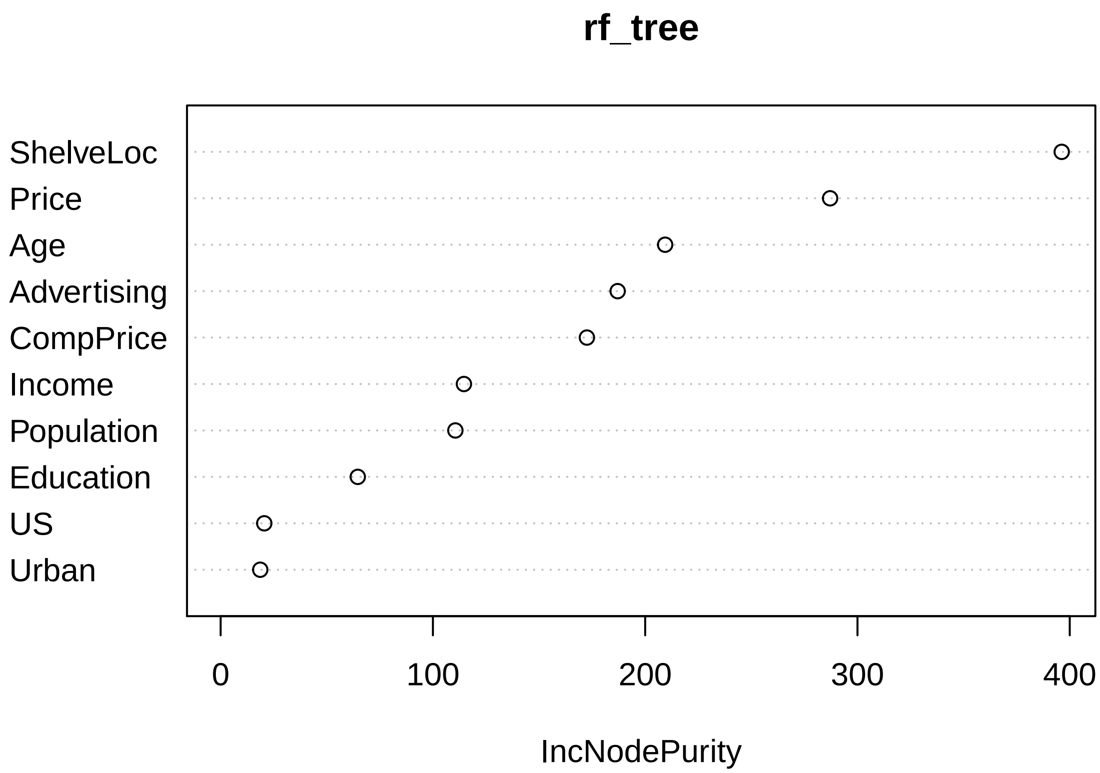
```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
rf_tree = randomForest(Sales~., train)
pred_rf = predict(rf_tree, Carseats[test_indx,])
mean((pred_rf-Carseats[test_indx,]$Sales)^2)
```

```
## [1] 3.080731
```

```r
varImpPlot(rf_tree)
```



ShelveLoc

Problem 3

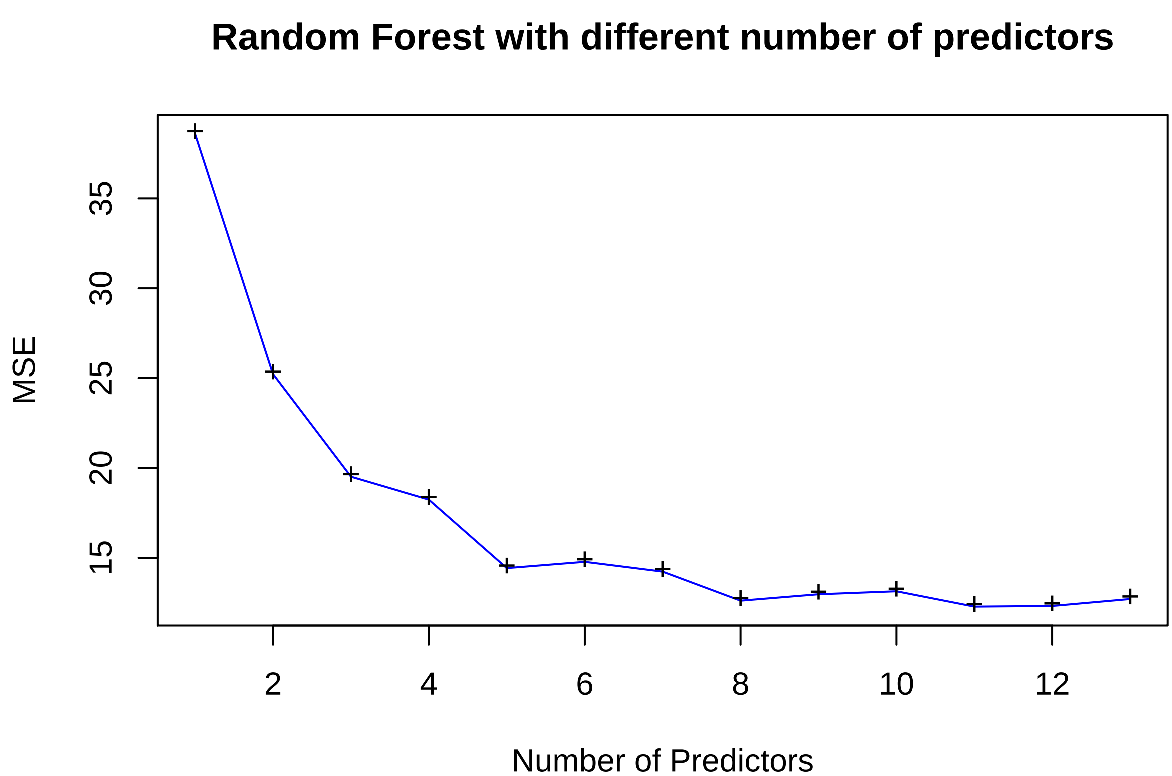
In the lab, we applied random forests to the Boston data using mtry=6 and ntree=100.

```
library(MASS)
head(Boston)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio  black
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12
##   lstat medv
## 1  4.98 24.0
## 2  9.14 21.6
## 3  4.03 34.7
## 4  2.94 33.4
## 5  5.33 36.2
## 6  5.21 28.7
```

(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.

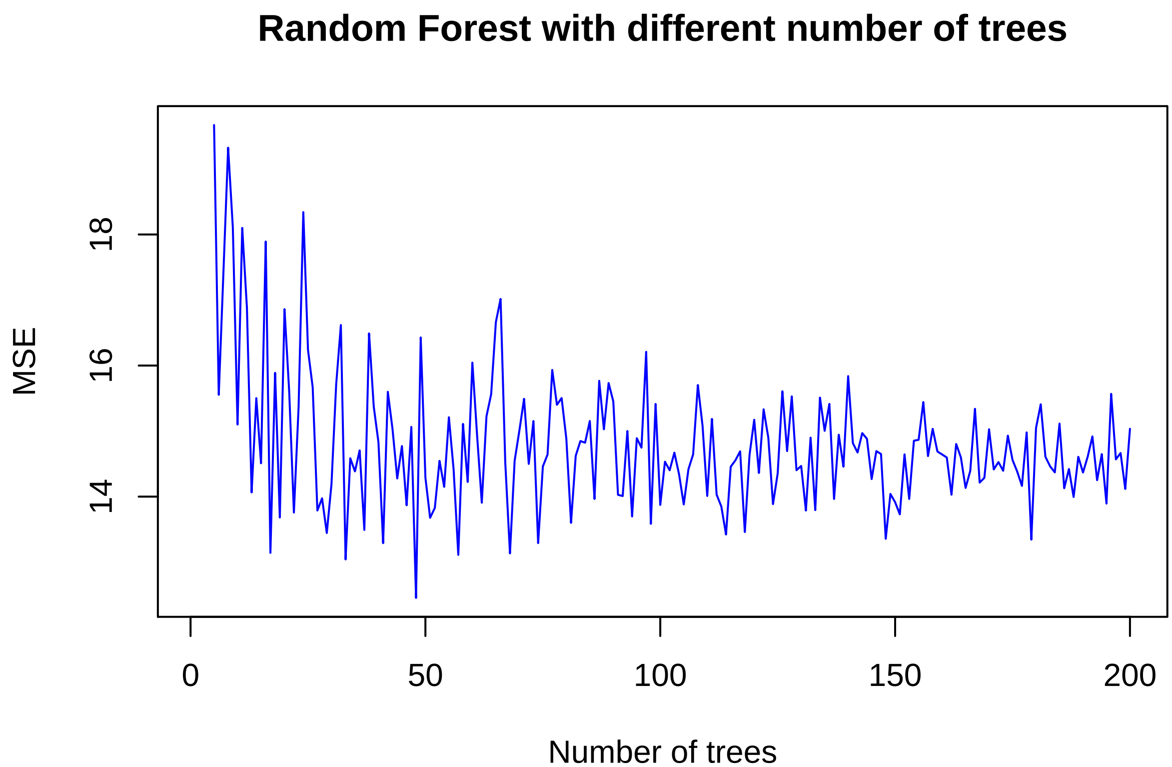
```
train_indx = sample(1:nrow(Boston), nrow(Boston)/2)
test_indx = (-train_indx)

attach(Boston)
train = Boston[c(train_indx),]
test = Boston[test_indx,]

library(randomForest)
mse = rep(NA, 13)
for (i in 1:13){
  rf = randomForest(medv ~., train, mtry = i, ntree = 100)
  pred_rf = predict(rf, test)
  mse[i] = mean((pred_rf-test$medv)^2)
}
plot(c(1:13), mse, pch='+', type='l', col=4,
     main = 'Random Forest with different number of predictors',
     xlab = 'Number of Predictors', ylab = 'MSE')
points(c(1:13), mse, pch='+')
```



As the number of predictors increases, the error decreases, but there is no improvement with more than 6 predictors

(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.

```
mse = rep(NA, length(c(5:200)))
for (i in 5:200){
  rf = randomForest(medv ~., train, mtry = 6, ntree = i)
  pred_rf = predict(rf, test)
  mse[i] = mean((pred_rf-test$medv)^2)
}
plot( mse, pch='+', type='l', col=4,
      main = 'Random Forest with different number of trees',
      xlab = 'Number of trees', ylab = 'MSE')
```



As the number of trees increases, the error decreases, but there is no improvement after tree size of 100