FE590. Assignment #3.

Yifu He

2019-04-12

Instructions

In this assignment, you should use R markdown to answer the questions below. Simply type your R code into embedded chunks as shown above.

When you have completed the assignment, knit the document into a PDF file, and upload *both* the .pdf and .Rmd files to Canvas.

Note that you must have LaTeX installed in order to knit the equations below. If you do not have it installed, simply delete the questions below.

Question 1 (based on JWHT Chapter 5, Problem 8)

In this problem, you will perform cross-validation on a simulated data set.

You will use this personalized simulated data set for this problem:

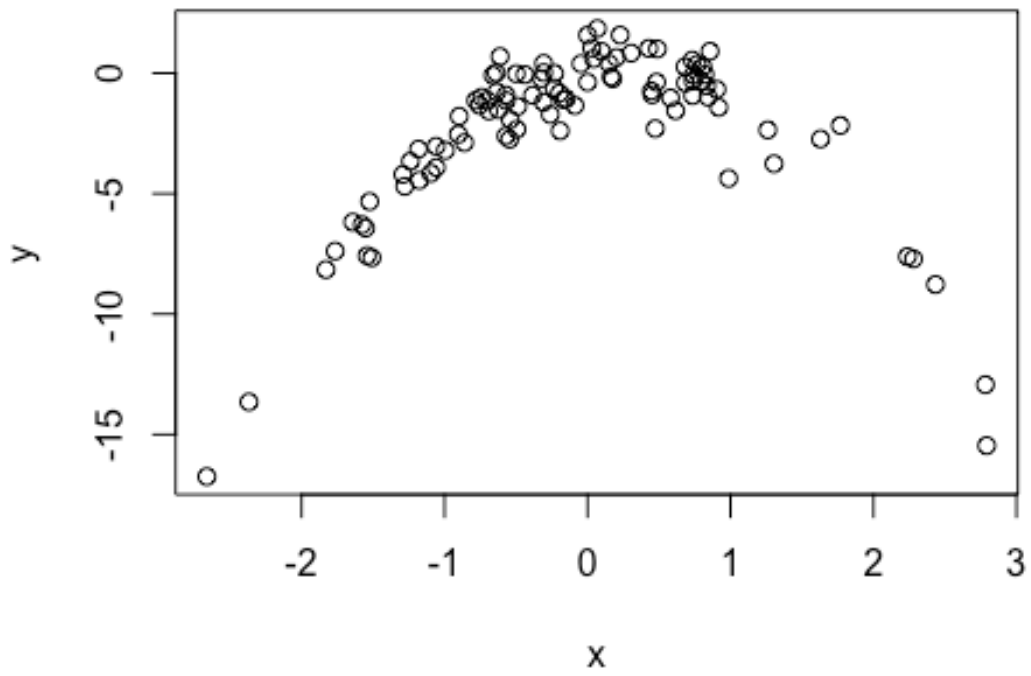
```
CWID = 10442277 #Place here your Campus wide ID number, this will personalize
#your results, but still maintain the reproduceable nature of using seeds.
#If you ever need to reset the seed in this assignment, use this as your seed
#Papers that use -1 as this CWID variable will earn 0's so make sure you
change
#this value before you submit your work.
personal = CWID %% 10000
set.seed(personal)
y <- rnorm(100)
x <- rnorm(100)
y <- x - 2*x^2 + rnorm(100)
```

(a) In this data set, what is n and what is p?

In this case, $n = 100$ and $p = 2$

(b) Create a scatterplot of x against y. Comment on what you find.

```
plot(x,y)
```



The scatterplot resembles the shape of a parabola.

```
library(boot)
Data = data.frame(x, y)
```

(c) Compute the LOOCV errors that result from fitting the following four models using least squares:

1. $Y = \beta_0 + \beta_1 X + \epsilon$
2. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon$
3. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$
4. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \epsilon$

```
cv1=glm(y~x)
cv.glm(Data,cv1)$delta

## [1] 13.58737 13.57970

cv2=glm(y~poly(x,2))
cv.glm(Data,cv2)$delta

## [1] 1.054212 1.053795

cv3=glm(y~poly(x,3))
cv.glm(Data,cv3)$delta
```

```
## [1] 1.085230 1.084616

cv4=glm(y~poly(x,4))
cv.glm(Data,cv4)$delta

## [1] 1.110495 1.109608
```

(d) Which of the models in (c) had the smallest LOOCV error? Is this what you expected? Explain your answer.

Model number 2 has the lowest LOOCV error. This is expected because the model is built upon the quadratic equation of x so the 2nd model should have the lowest error.

(e) Comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawnbased on the cross-validation results?

```
lsq=glm(y~poly(x,4))
summary(lsq)

##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5124  -0.6218   0.0971   0.6358   2.1329
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.2610     0.1005 -22.503  < 2e-16 ***
## poly(x, 4)1   4.5247     1.0047   4.503  1.9e-05 ***
## poly(x, 4)2 -33.3632     1.0047 -33.206  < 2e-16 ***
## poly(x, 4)3  -0.4344     1.0047  -0.432    0.666
## poly(x, 4)4  -1.2281     1.0047  -1.222    0.225
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.009511)
##
##     Null deviance: 1231.175  on 99  degrees of freedom
## Residual deviance:   95.904  on 95  degrees of freedom
## AIC: 291.6
##
## Number of Fisher Scoring iterations: 2
```

The result from the least square models agrees with the result from the LOOCV in c) as the p value for 3rd and 4th is very large, meaning they are very likely to be zero.

Question 2 (based on JWTH Chapter 7, Problem 10)

The question refers to the 'College' data set

```
library(ISLR)
library(leaps)
library(gam)
```

```
## Loading required package: splines
```

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

```
## Loaded gam 1.16
```

```
attach(College)
```

(a) Split the data into a training set and a test set. Using out-of-state tuition as the response and the other variables as the predictors, perform subset selection (your choice on how) in order to identify a satisfactory model that uses just a subset of the predictors (if your approach suggests using all of the predictors, then follow your results and use them all).

```
n=length(Outstate)
sampling=sample(n,n/2)
train=College[sampling,]
test=College[-sampling,]
reg.fit=regsubsets(Outstate~.,data=train,nvmax=17)
result=data.frame(rank(summary(reg.fit)$cp),
                  rank(summary(reg.fit)$bic),
                  rank(-summary(reg.fit)$adjr))
colnames(result)=c("Mallow's Cp", "BIC","Adj R^2")
result
```

```
##    Mallow's Cp BIC Adj R^2
## 1           17  17      17
## 2           16  16      16
## 3           15  15      15
## 4           14  13      14
## 5           13   9      13
## 6           12   7      12
## 7           11   6      11
## 8           10   4      10
## 9            8   3       9
## 10           3   1       7
## 11           1   2       3
## 12           2   5       1
## 13           4   8       2
## 14           5  10       4
## 15           6  11       5
## 16           7  12       6
## 17           9  14       8
```

Based on the overall ranking of the 17 parameters over 3 criteria, the model with 11, 12, and 13 parameters share the best position. For that, I would choose the 12 parameters model over the other two for compromisation. The 12 chosen parameters are:

```
coefmodel=coef(reg.fit,id=12)
names(coefmodel)
```

```
##  [1] "(Intercept)" "PrivateYes"  "Apps"        "Accept"      "Top10perc"
##  [6] "F.Undergrad" "Room.Board"  "Books"       "Terminal"    "S.F.Ratio"
## [11] "perc.alumni" "Expend"      "Grad.Rate"
```

(b) Fit a GAM on the training data, using out-of-state tuition as the response and the features selected in the previous step as the predictors, using splines of each feature with 5 df.

```
gam.fit=gam(Outstate~s(Apps,5)
                    +s(Accept,5)
                    +s(Top10perc,5)
                    +s(F.Undergrad,5)
                    +s(Room.Board,5)
                    +s(Personal,5)
                    +s(Terminal,5)
                    +s(S.F.Ratio,5)
                    +s(perc.alumni,5)
                    +s(Expend,5)
                    +s(Grad.Rate,5)
                    +Private,data=train)
```

(c) Evaluate the model obtained on the test set, and explain the results obtained

```
mean((train$Outstate-predict(gam.fit,train))^2)
```

```
## [1] 2516834
```

```
mean((test$Outstate-predict(gam.fit,test))^2)
```

```
## [1] 3547894
```

The result obtained show that the model does not perform well with a new data set, meaning the model used maybe only suitable for the training set only.

(d) For which variables, if any, is there evidence of a non-linear relationship with the response? Which are probably linear? Justify your answers.

```
summary(gam.fit)
```

```
##
## Call: gam(formula = Outstate ~ s(Apps, 5) + s(Accept, 5) + s(Top10perc,
##     5) + s(F.Undergrad, 5) + s(Room.Board, 5) + s(Personal, 5) +
##     s(Terminal, 5) + s(S.F.Ratio, 5) + s(perc.alumni, 5) + s(Expend,
##     5) + s(Grad.Rate, 5) + Private, data = train)
## Deviance Residuals:
##       Min        1Q    Median        3Q       Max
## -6043.036  -993.877    -5.458   984.647  7081.285
```

```
##
## (Dispersion Parameter for gaussian family taken to be 2950300)
##
##     Null Deviance: 6314402547 on 387 degrees of freedom
## Residual Deviance: 976547754 on 330.9995 degrees of freedom
## AIC: 6935.646
##
## Number of Local Scoring Iterations: 3
##
## Anova for Parametric Effects
##                     Df     Sum Sq    Mean Sq  F value    Pr(>F)
## s(Apps, 5)           1   36492332   36492332  12.3690 0.0004974 ***
## s(Accept, 5)         1  534789904  534789904 181.2663 < 2.2e-16 ***
## s(Top10perc, 5)      1 1506012323 1506012323 510.4608 < 2.2e-16 ***
## s(F.Undergrad, 5)    1  809050930  809050930 274.2267 < 2.2e-16 ***
## s(Room.Board, 5)     1  622826971  622826971 211.1063 < 2.2e-16 ***
## s(Personal, 5)       1     883213     883213   0.2994 0.5846503
## s(Terminal, 5)       1   17419064   17419064   5.9042 0.0156364 *
## s(S.F.Ratio, 5)      1  110971687  110971687  37.6137 2.457e-09 ***
## s(perc.alumni, 5)    1  117713919  117713919  39.8990 8.617e-10 ***
## s(Expend, 5)         1  417865617  417865617 141.6350 < 2.2e-16 ***
## s(Grad.Rate, 5)      1   42791525   42791525  14.5041 0.0001667 ***
## Private              1   86858238   86858238  29.4405 1.116e-07 ***
## Residuals          331  976547754    2950300
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                    Npar Df  Npar F     Pr(F)
## (Intercept)
## s(Apps, 5)               4  1.5406 0.1900663
## s(Accept, 5)             4  9.0554 5.947e-07 ***
## s(Top10perc, 5)          4  1.6457 0.1624293
## s(F.Undergrad, 5)        4  8.3547 1.974e-06 ***
## s(Room.Board, 5)         4  1.4386 0.2208779
## s(Personal, 5)           4  4.8187 0.0008642 ***
## s(Terminal, 5)           4  1.2397 0.2938418
## s(S.F.Ratio, 5)          4  3.3573 0.0103241 *
## s(perc.alumni, 5)        4  1.7674 0.1350354
## s(Expend, 5)             4 16.1281 4.447e-12 ***
## s(Grad.Rate, 5)          4  1.4091 0.2305886
## Private
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the ANOVA non-parametric test, we can see clearly that there should exist a non-linear relationship between the group of Accept, F.Undergrad, and Expend and the response. In addition, Apps, Top10perc, S.F.Ratio, and Grad.Rate also seem to have good possibility of having a non-linear relationship. It is very likely that the rest parameters should have a linear relationship with the response.

Question 3 (based on JWHT Chapter 7, Problem 6)

In this exercise, you will further analyze the Wage data set.

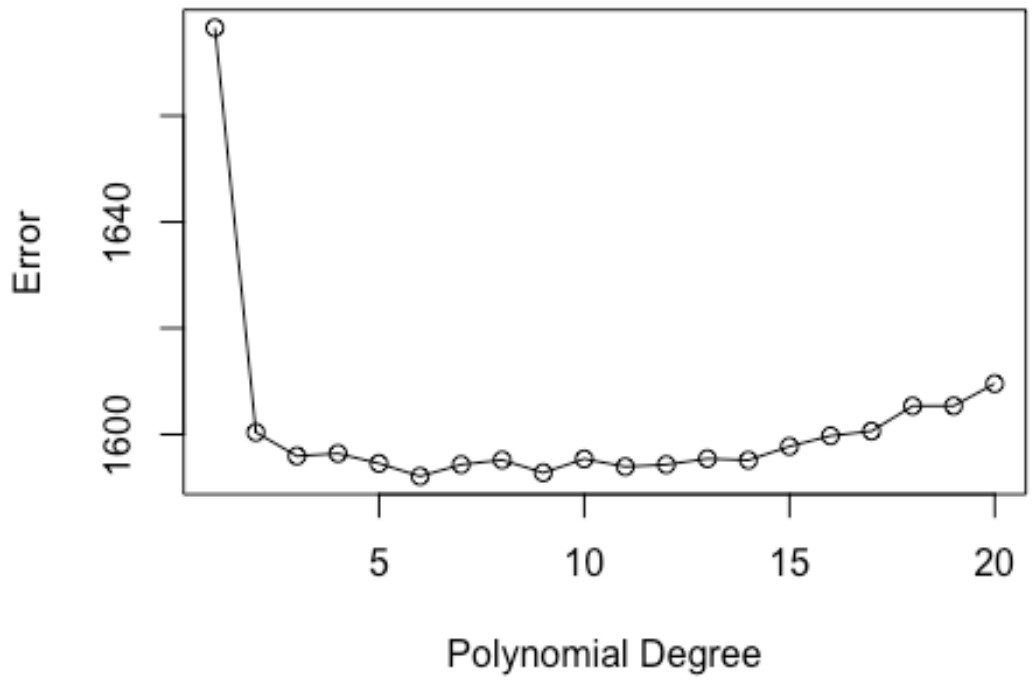
```
attach(Wage)
```

(a) Perform polynomial regression to predict wage using age. Use cross-validation to select the optimal degree d for the polynomial. What degree was chosen? Make a plot of the resulting polynomial fit to the data.

```
#Using k=5 Cross-Validation
result=rep(NA,20)
for (i in c(1:20)){
  poly.fit=glm(wage~poly(age,i),data=Wage)
  result[i]=cv.glm(Wage,poly.fit,K=5)$delta[2]
}
which.min(result)
```

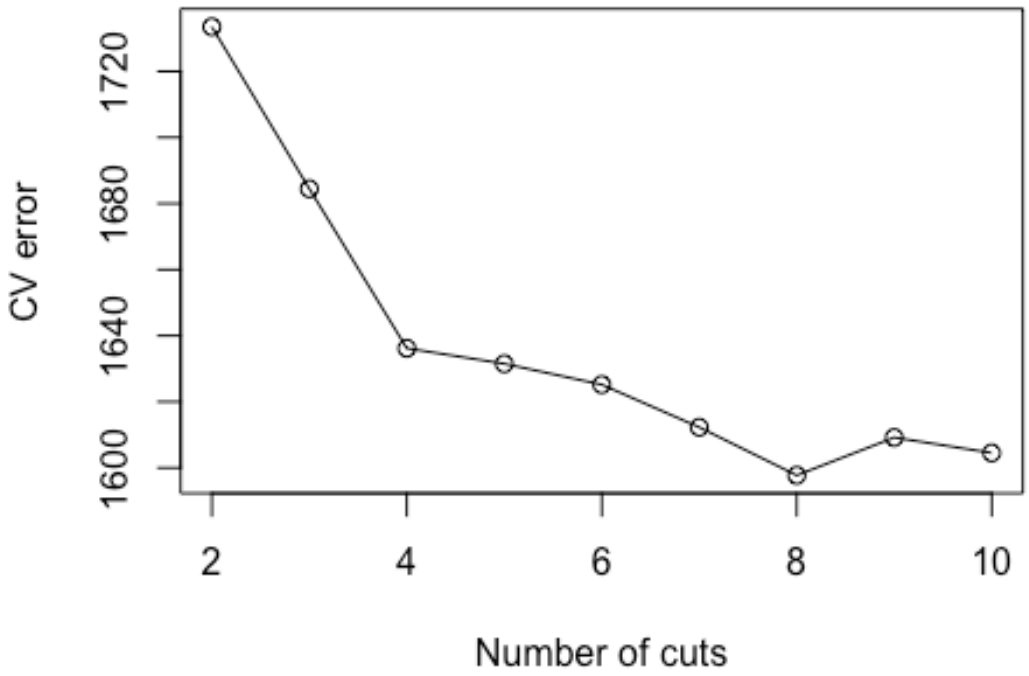
```
## [1] 6
```

```
plot(c(1:20),result,xlab="Polynomial Degree",ylab="Error")
lines(c(1:20),result)
```



(b) Fit a step function to predict wage using age, and perform cross-validation to choose the optimal number of cuts. Make a plot of the fit obtained.

```
cuts = rep(NA, 10)
for (i in 2:10) {
  Wage$age.cut = cut(Wage$age, i)
  lm.fit = glm(wage~age.cut, data=Wage)
  cuts[i] = cv.glm(Wage, lm.fit, K=10)$delta[2]
}
cuts=cuts[-1]
plot(2:10, cuts, xlab="Number of cuts", ylab="CV error")
lines(2:10, cuts)
```




```
which.min(cuts)+1 #Add 1 to adjust for the start from 2, not from 1
```

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

Question 4 (based on JWHT Chapter 8, Problem 8)

In the lab, a classification tree was applied to the `Carseats` data set after converting Sales into a qualitative response variable. Now we will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable.

```
library(tree)
library(randomForest)

## randomForest 4.6-14

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

attach(Carseats)
```

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

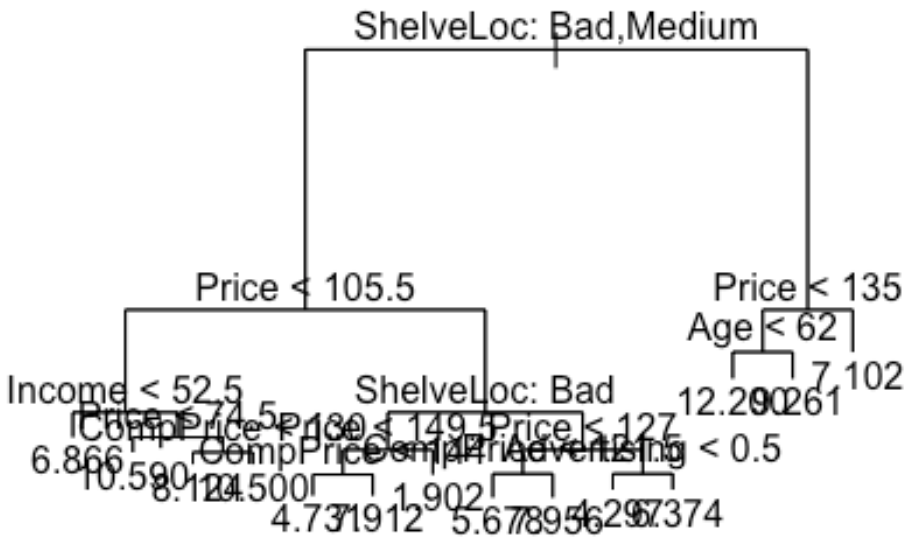
```
n=length(Sales)
sampling.tree=sample(n,n/2)
train.tree=Carseats[sampling.tree,]
test.tree=Carseats[-sampling.tree,]
```

(b) Fit a regression tree to the training set. Plot the tree, and interpret the results. What test MSE do you obtain?

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

##
## Regression tree:
## tree(formula = Sales ~ ., data = train.tree)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Income"      "CompPrice"   "Advertising"
## [6] "Age"
## Number of terminal nodes:  14
## Residual mean deviance:  2.313 = 430.2 / 186
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -5.13400 -0.85350  0.07756  0.00000  0.94970  4.06900

plot(car.tree)
text(car.tree,pretty=0)
```



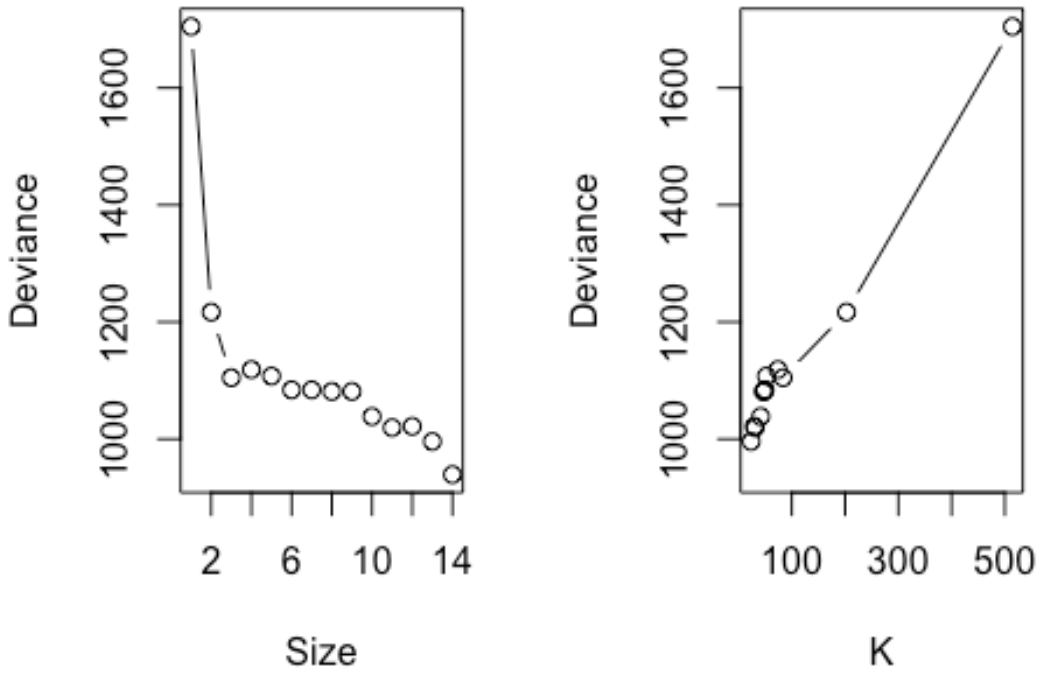

```
pred.tree=predict(car.tree,test.tree)
mean((test.tree$Sales-pred.tree)^2)

## [1] 4.452387
```

The tree method breaks the data by the Shelve Location first with Bad and Medium into one group and the remaining ones (good) in to the other group. After the dividing the training set based on ShelveLoc, the method continues to divide the data set based on different stats with the total 18 terminal nodes.

(c) Use cross-validation in order to determine the optimal level of tree complexity. Does pruning the tree improve the test MSE?

```
car.tree.cv=cv.tree(car.tree,FUN = prune.tree)
par(mfrow = c(1, 2))
plot(car.tree.cv$size, car.tree.cv$dev, type =
"b",xlab="Size",ylab="Deviance")
plot(car.tree.cv$k, car.tree.cv$dev, type = "b",xlab="K",ylab="Deviance")
```




```
optimal=car.tree.cv$size[which.min(car.tree.cv$dev)]
pruned.car.tree = prune.tree(car.tree, best = optimal)
pred.pruned=predict(pruned.car.tree,test.tree)
mean((test.tree$Sales-pred.pruned)^2)

## [1] 4.452387
```

The MSE does not improve as the optimal number of nodes is still the maximum

(d) Use the bagging approach in order to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important.

```
bag.car=randomForest(Sales~.,data=train.tree,mtry=10,ntree=1000,importance=T)
bag.pred=predict(bag.car,test.tree)
mean((test.tree$Sales-bag.pred)^2)

## [1] 2.888286

importance(bag.car)

##                %IncMSE IncNodePurity
## CompPrice   30.6269443    136.603047
## Income      14.9089733    112.767271
## Advertising 15.1743462     81.546307
## Population  -2.1069189     66.084403
```

```
## Price       69.9945237    460.676451
## ShelveLoc   83.8107500    587.656571
## Age         22.0803521    130.433324
## Education   -2.4731018     43.987612
## Urban        1.2099727      9.433934
## US           0.3940582      4.340877
```

Price and ShelveLoc are the most important variables.

Question 5 (based on JWTH Chapter 8, Problem 10)

Use boosting (and bagging) to predict Salary in the Hitters data set

```
library(gbm)

## Warning: package 'gbm' was built under R version 3.5.2

## Loaded gbm 2.1.5

attach(Hitters)
```

(a) Remove the observations for which salary is unknown, and then log-transform the salaries

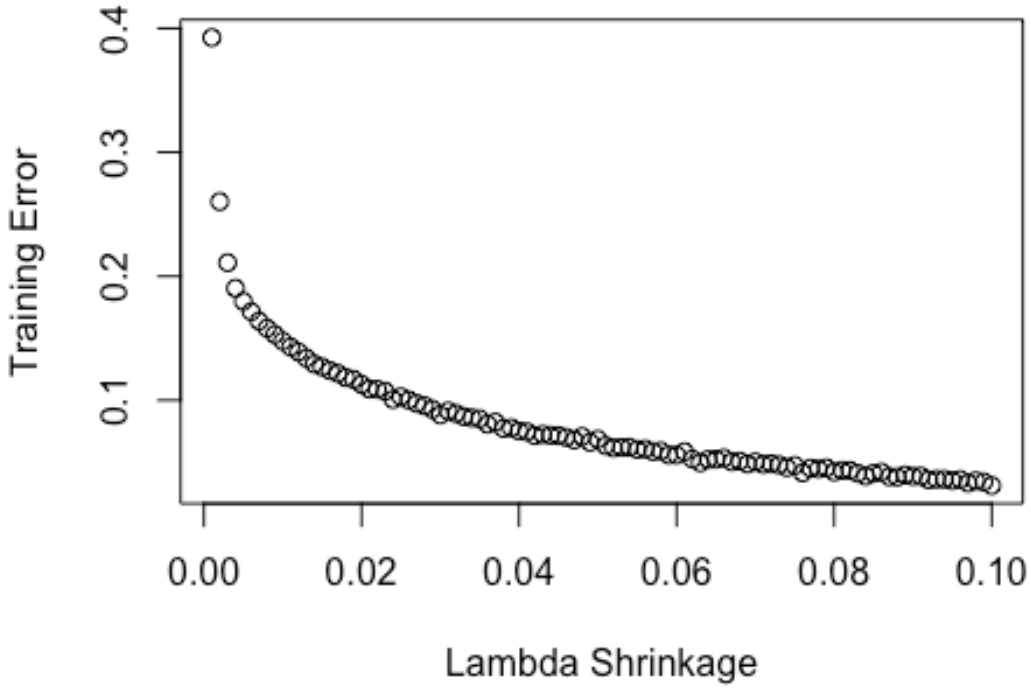
```
Hitters = Hitters[-which(is.na(Hitters$Salary)), ]
Hitters$Salary=log(Hitters$Salary)
```

(b) Split the data into training and testing sets for cross validation purposes.

```
hitters.sampling=sample(length(Hitters$Salary),length(Hitters$Salary)/2)
hitters.train=Hitters[hitters.sampling,]
hitters.test=Hitters[-hitters.sampling,]
```

(c) Perform boosting on the training set with 1000 trees for a range of values of the shrinkage parameter λ. Produce a plot with different shrinkage parameters on the x-axis and the corresponding training set MSE on the y-axis

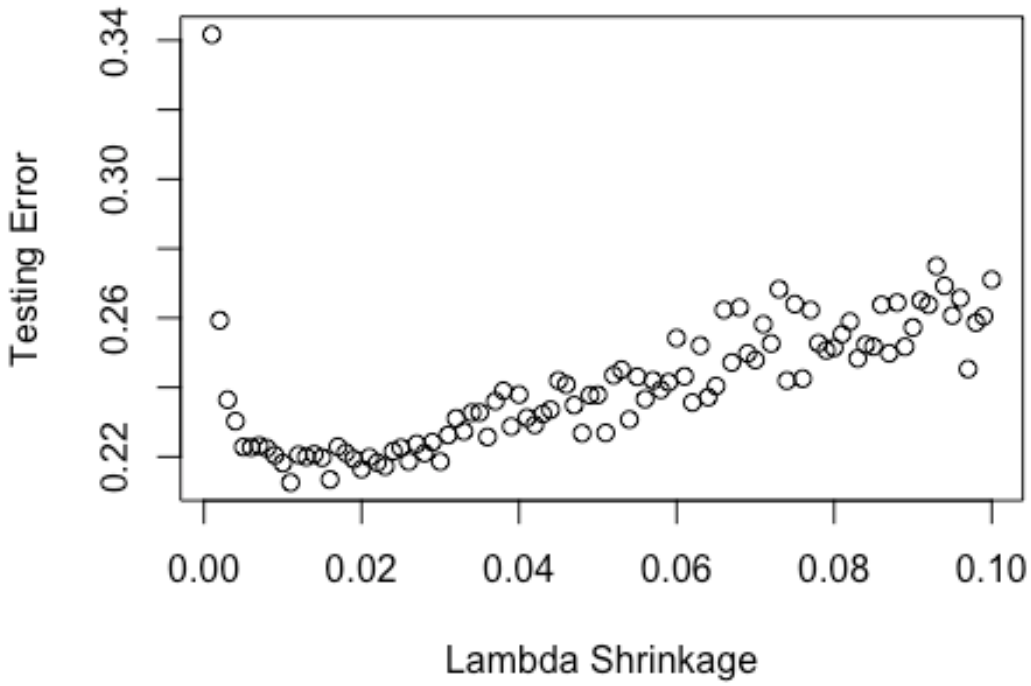
```
lambdas = seq(0.001, 0.1, by = 0.001)
len=length(lambdas)
train.err=rep(NA,len)
test.err=rep(NA,len)
for(i in c(1:len)){
  boost=gbm(Salary~.,data=hitters.train,
            distribution = "gaussian",
            n.trees = 1000,
            shrinkage = lambdas[i])
  train.pred=predict(boost, hitters.train, n.trees=1000)
  test.pred=predict(boost, hitters.test, n.trees=1000)
  train.err[i]=mean((hitters.train$Salary-train.pred)^2)
  test.err[i]=mean((hitters.test$Salary-test.pred)^2)
}
plot(lambdas,train.err,ylab="Training Error",xlab="Lambda Shrinkage")
```



(d)

Produce a plot similar to the last one, but this time using the test set MSE

```
plot(lambdas,test.err,ylab="Testing Error",xlab="Lambda Shrinkage")
```



(e) Fit the model using two other regression techniques (from previous classes) and compare the MSE of those techniques to the results of these boosted trees.

```
#first technique is the linear regression
hit.lm=lm(Salary~.,data=hitters.train)
hit.pred.lm=predict(hit.lm,hitters.test)
mean((hitters.test$Salary-hit.pred.lm)^2)
```

```
## [1] 0.4547007
```

```
#the second one is GAM
hit.gam=gam(Salary~s(AtBat,5)
                   +s(Hits,5)
                   +s(HmRun,5)
                   +s(RBI,5)
                   +s(Walks,5)
                   +s(Years,5)
                   +s(CAtBat,5)
                   +s(CHits,5)
                   +s(CHmRun,5)
                   +s(CRuns,5)
                   +s(CRBI,5)
                   +s(CWalks,5)
                   +s(PutOuts,5)
```

```
                    +s(Assists,5)
                    +s(Errors,5)
                    +League
                    +Division
                    +NewLeague,data = hitters.train)
hit.pred.gam=predict(hit.gam,hitters.test)
mean((hitters.test$Salary-hit.pred.gam)^2)

## [1] 0.2161251
```

The testing errors of the Linear Regression technique is worse than most of the lambda values from the boosted trees, only better than $\lambda = 0.001$, while the GAM outperforms most of the boosted trees.

(f) Reproduce (c) and (d), but this time use bagging instead of boosting and compare to the boosted MSE's and the MSE's from (e)

```
hit.bag=randomForest(Salary ~ ., data = hitters.train, ntree = 1000, mtry =
19)
hit.bag.pred=predict(hit.bag,hitters.test)
mean((hitters.test$Salary-hit.bag.pred)^2)

## [1] 0.1780143
```

The MSE for bagged trees is much better than Linear Regression and GAM models