FE582 Assignment 3

Mugdha

11/9/2020

Problem 1

a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

Solution:

```
library(class)    # for KNN
library(ISLR)     # for Data
```

```
## Warning: package 'ISLR' was built under R version 4.0.3
```

```
library(MASS)     # for LDA
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.0.3
```

```
head(Weekly)
```

```
##   Year   Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

```
summary(Weekly)
```

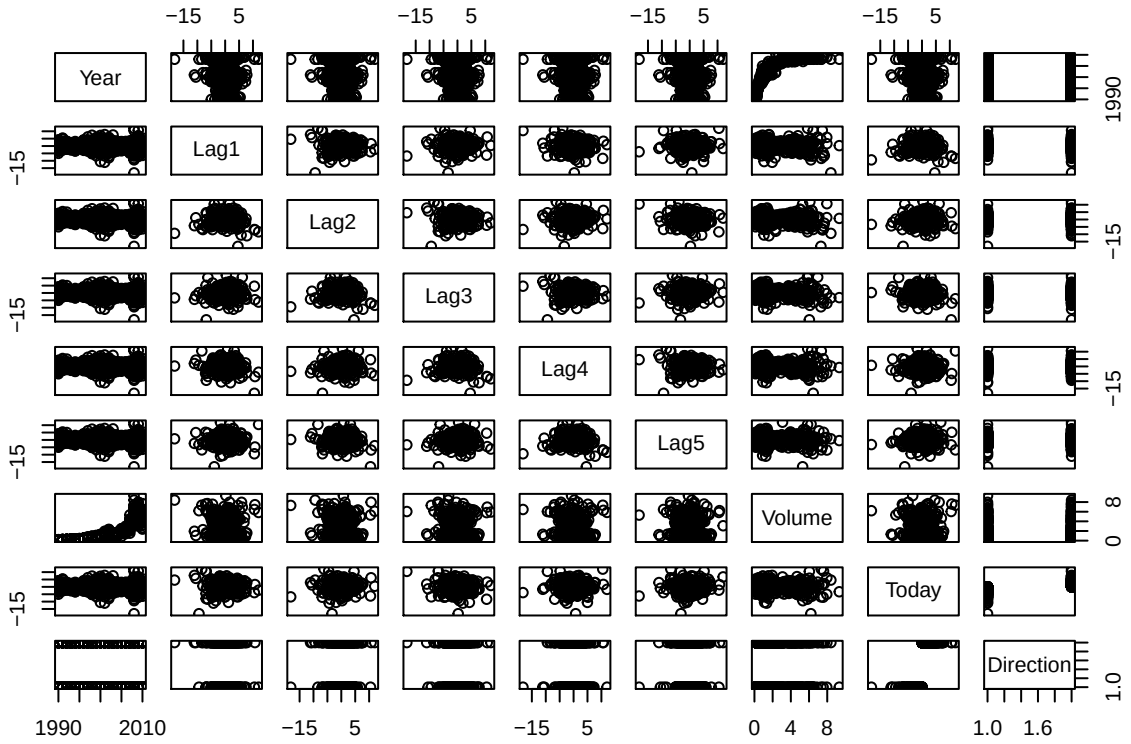
```
##       Year           Lag1               Lag2               Lag3          
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       Lag4               Lag5              Volume            Today          
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540  
```

```
##  Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260
##  Direction
##  Down:484
##  Up  :605
##
##
##
##
```

```
cor(Weekly[, -9])
```

```
##               Year         Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
## Lag5    1.000000000 -0.05851741  0.011012698
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

```
pairs(Weekly)
```



Observations: **The Summary statistics and Scatter plot don't provide any obvious patterns except that the Volume of shares traded each week has grown quite a lot over the years and flattened out in recent years. The previous 5 weeks' rate of returns have no correlation with any other variables as indicated by pairwise correlation plots as well as correlation coefficient numbers close to 0.**

b)Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

Solution:

```
glm.fit = glm(Direction ~ Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data=Weekly, family=binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
```

```
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

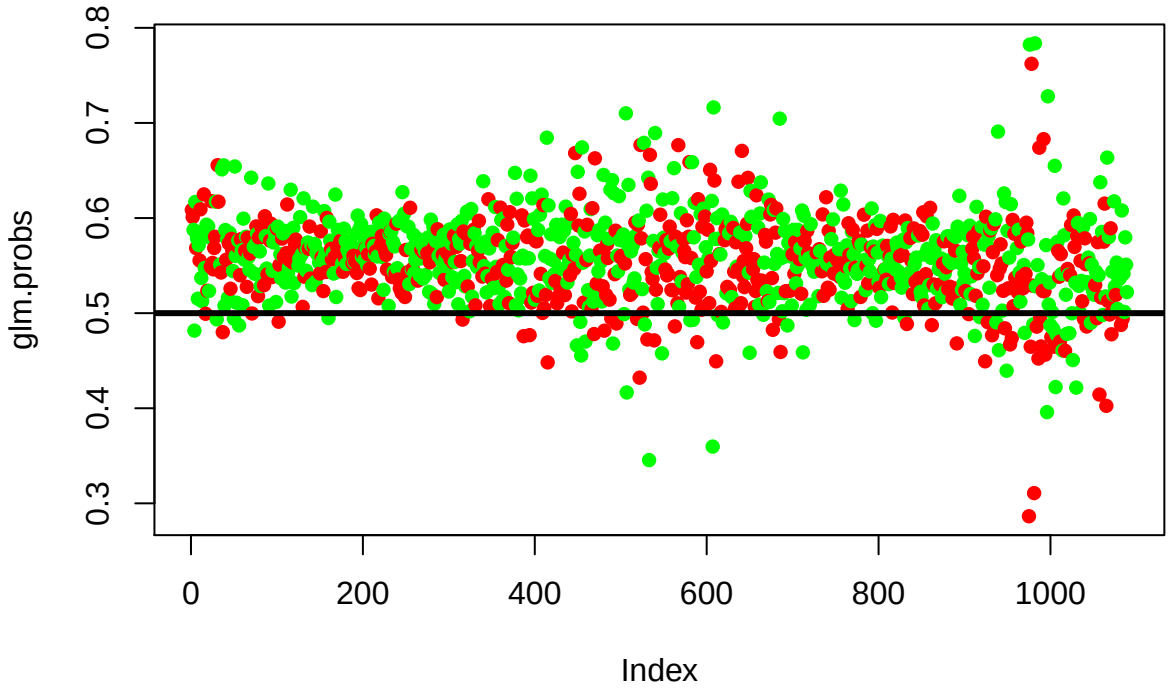
Significant predictors: **Out of the 6 predictors,Lag2 is the only significant predictor of the "direction" as it has p-value less than significance level of 0.05.**

c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.
Solution:

```
glm.probs = predict(glm.fit, type="response")
glm.preds = rep("Down", 1089)
glm.preds[glm.probs > 0.5] = "Up"
table(glm.preds, Weekly$Direction)
```

```
##
## glm.preds Down  Up
##      Down   54  48
##      Up    430 557
```

```
plot(glm.probs, col= ifelse(Weekly$Direction=="Down", "red","green"), pch=16)
abline(h = 0.5, lwd= 3)
```



Observations: **The logistic regression model using the five lag variables along with Volume as the predictors, and a prediction threshold of 0.5, correctly predicted 54 down weeks out of a total of 484 actual down weeks and 557 up days out of a total of 605 actual up weeks. 56.1% of the responses are predicted correctly.Training error rate = 43.89%, overly optimistic.For weeks when the market goes down, the model is right only 11.1570248% of the time (54/(54+430)).**

d) Now fit the logistic regression model using a training data period from 1990 to 2008,with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

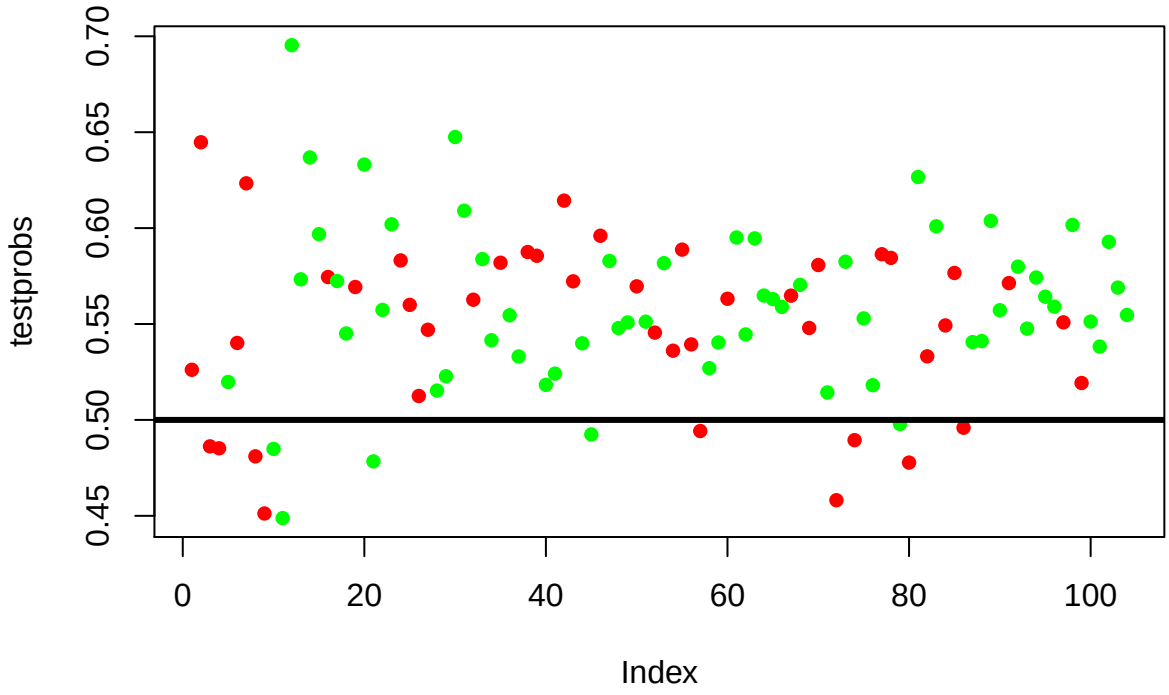
Solution:

```
training.data = Weekly[Weekly$Year<2009,]
test.data = Weekly[Weekly$Year>2008,]
simpglm = glm(Direction~Lag2, data= training.data, family = "binomial")
summary(simpglm)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2, family = "binomial", data = training.data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.536  -1.264   1.021   1.091   1.368
##
## Coefficients:
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.20326    0.06428   3.162  0.00157 **
## Lag2         0.05810    0.02870   2.024  0.04298 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1350.5  on 983  degrees of freedom
## AIC: 1354.5
##
## Number of Fisher Scoring iterations: 4
```

```
testprobs = predict(simpglm, type="response", newdata = test.data)
testdirs = Weekly$Direction[Weekly$Year>2008]
plot(testprobs, col= ifelse(Weekly$Direction[Weekly$Year>2008]=="Down", "red","green"), pch=16)
abline(h = 0.5, lwd= 3)
```




```
testpreds = rep("Down", 104)
testpreds[testprobs>0.5] = "Up"
#mean(probs)
table(testpreds, testdirs)
```

```
##          testdirs
```

```
## testpreds Down Up
##      Down    9  5
##      Up     34 56
```

Observations: **Using only Lag2 as the predictor, the model correctly predicted the market direction for 62.5% of the weeks in the data We could also say that for weeks when the market goes up, the model is right 91.8032787% of the time (56/(56+5)). For weeks when the market goes down, the model is right only 20.9302326% of the time (9/(9+34)).**

e) Repeat d) using LDA

Solution:

```
lda.fit = lda(Direction~Lag2, data= training.data)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag2, data = training.data)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
##            LD1
## Lag2 0.4414162
```

```
lda.pred = predict(lda.fit, newdata=test.data, type="response")
lda.class = lda.pred$class
table(lda.class, test.data$Direction)
```

```
##
## lda.class Down Up
##      Down    9  5
##      Up     34 56
```

Observations: **Our classifier still says that most of the samples go Up. the percentage of correct predictions on the test data is 62.5%.Error rate is 37.5%.These results are very close to those obtained with the logistic regression model**

f) Repeat d) using QDA

Solution:

```
qda.fit = qda(Direction~Lag2, data= training.data)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag2, data = training.data)
```

```
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
```

```
qda.pred = predict(qda.fit, newdata=test.data, type="response")
qda.class = qda.pred$class
table(qda.class, test.data$Direction)
```

```
##
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

Observations: **The error rate for the QDA is the worst out of all the models: 41,35%.For weeks when the market goes up, the model is right 100% of the time. For weeks when the market goes down, the model is right 0% of the time.**

g) Repeat d) using KNN with K = 1

Solution:

```
set.seed(1)
train.X = cbind(training.data$Lag2)
test.X = cbind(test.data$Lag2)
train.Y = cbind(training.data$Direction)
knn.pred = knn(train.X, test.X, train.Y, k=1)
table(knn.pred, test.data$Direction)
```

```
##
## knn.pred Down Up
##        1   21 30
##        2   22 31
```

Observations: **The percentage of correct predictions on the test data is 50%.So, 50% is the test error rate. Also, for weeks when the market goes up, the model is right 50.8196721% of the time. For weeks when the market goes down, the model is right only 48.8372093% of the time.**

h) Which of these methods appears to provide the best results on this data?

Solution: **If we compare the test error rates,logistic regression and LDA have the minimum error rates, followed by QDA and KNN.**

i) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables,method, and associated confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier.

Solution:

```
knn3.pred = knn(train.X, test.X, train.Y, k=3)
table(knn3.pred, test.data$Direction)
```

```
##
## knn3.pred Down Up
##         1   16 19
##         2   27 42
```

```
knn3.pred = knn(train.X, test.X, train.Y, k=15)
table(knn3.pred, test.data$Direction)
```

```
##
## knn3.pred Down Up
##         1   20 20
##         2   23 41
```

```
qda.fit2 = qda(Direction~Lag1 + Lag2 + Lag4, data= training.data)
qda.fit2
```

```
## Call:
## qda(Direction ~ Lag1 + Lag2 + Lag4, data = training.data)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1        Lag2       Lag4
## Down  0.289444444 -0.03568254 0.15925624
## Up   -0.009213235  0.26036581 0.09220956
```

```
qda.pred2 = predict(qda.fit2, newdata=test.data, type="response")
qda.class2 = qda.pred2$class
table(qda.class2, test.data$Direction)
```

```
##
## qda.class2 Down Up
##       Down    9 20
##       Up     34 41
```

```
lda.fit2 = lda(Direction~Lag1 + Lag2 + Lag4, data= training.data)
lda.fit2
```

```
## Call:
## lda(Direction ~ Lag1 + Lag2 + Lag4, data = training.data)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
```

```
## Group means:
##              Lag1        Lag2       Lag4
## Down  0.289444444 -0.03568254 0.15925624
## Up   -0.009213235  0.26036581 0.09220956
##
## Coefficients of linear discriminants:
##             LD1
## Lag1 -0.2984478
## Lag2  0.2960224
## Lag4 -0.1113485
```

```
lda.pred2 = predict(lda.fit2, newdata=test.data, type="response")
lda.class2 = lda.pred2$class
table(lda.class2, test.data$Direction)
```

```
##
## lda.class2 Down Up
##       Down    9  7
##       Up     34 54
```

Observations:

In these cases,the models are not much better than with only Lag2 as predictor

Problem2

a) Create a binary variable, mpg01, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the median() function. Note you may find it helpful to use the data.frame() function to create a single data set containing both mpg01 and the other Auto variables.

Solution:

```
auto = Auto
auto$mpg01 = rep(0, length(auto$mpg))
auto$mpg01[auto$mpg>median(auto$mpg)] = 1
```

b)Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

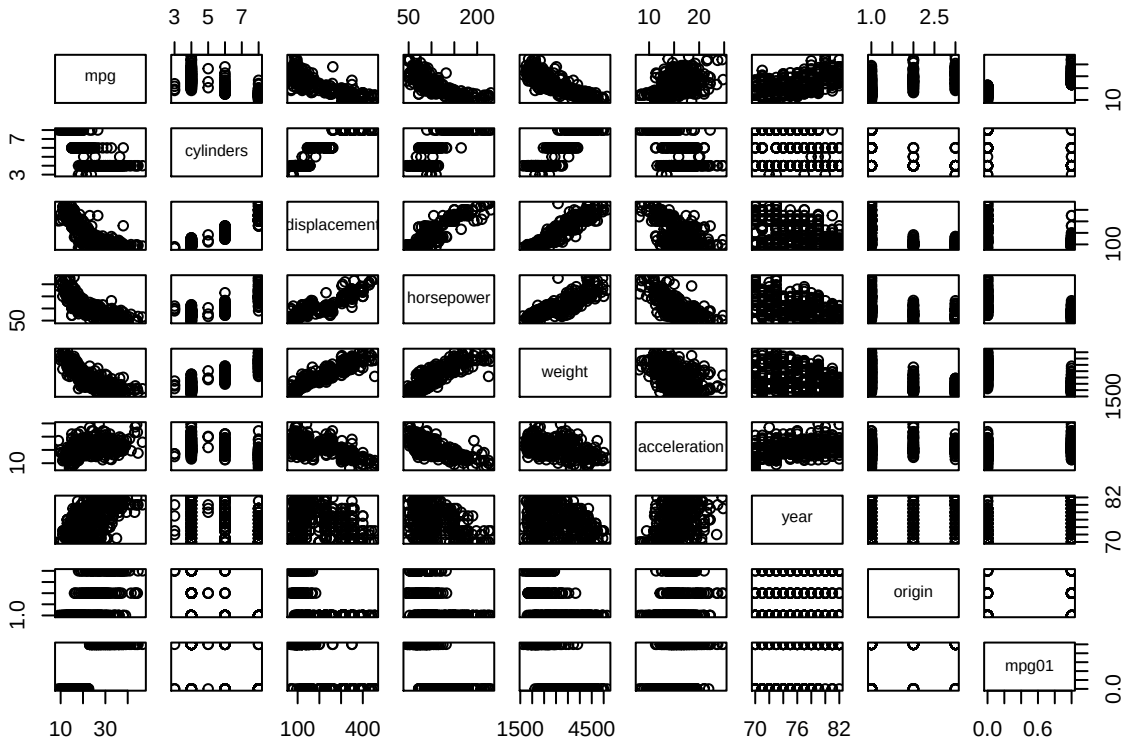
Solution:

```
cor(auto[, -9])
```

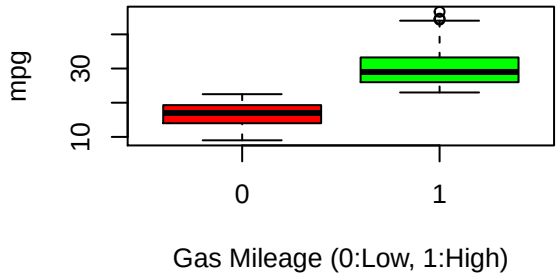
```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
```

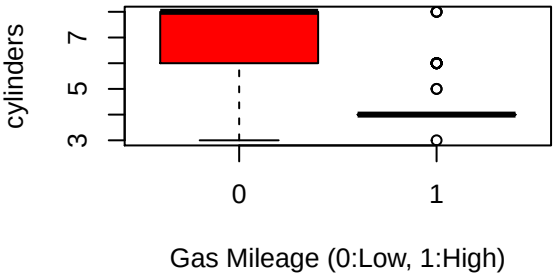
```
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
## mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566
##              acceleration       year     origin      mpg01
## mpg             0.4233285  0.5805410  0.5652088  0.8369392
## cylinders      -0.5046834 -0.3456474 -0.5689316 -0.7591939
## displacement   -0.5438005 -0.3698552 -0.6145351 -0.7534766
## horsepower     -0.6891955 -0.4163615 -0.4551715 -0.6670526
## weight         -0.4168392 -0.3091199 -0.5850054 -0.7577566
## acceleration    1.0000000  0.2903161  0.2127458  0.3468215
## year            0.2903161  1.0000000  0.1815277  0.4299042
## origin          0.2127458  0.1815277  1.0000000  0.5136984
## mpg01           0.3468215  0.4299042  0.5136984  1.0000000
```

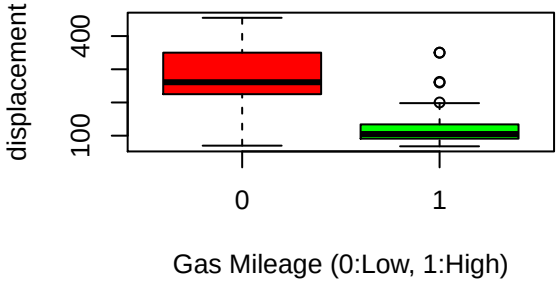
```
pairs(auto[,-9])
```

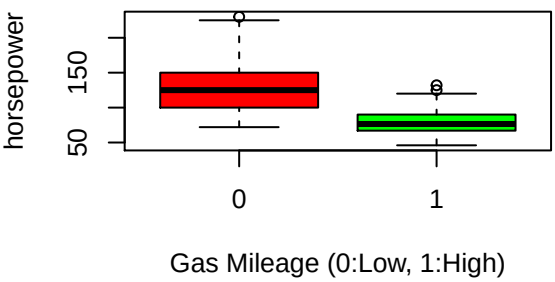


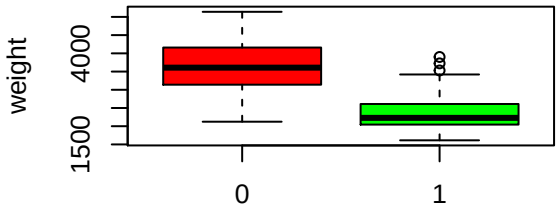

```
par(mfrow = c(2,2))
for(i in 1:(ncol(auto)-2)){
        boxplot(auto[,i] ~ as.factor(auto$mpg01),
                xlab = "Gas Mileage (0:Low, 1:High)",
                ylab = colnames(auto)[i],
                col = c("red","green"))

}
```

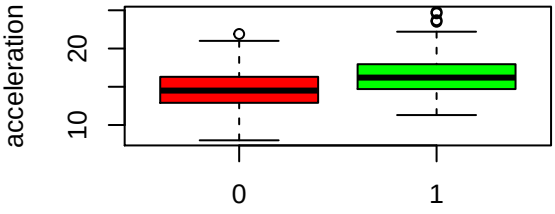

mpg
10
30
0
1
Gas Mileage (0:Low, 1:High)

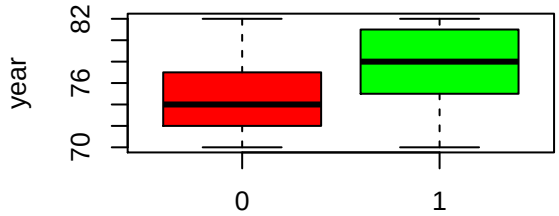

cylinders
3
5
7
0
1
Gas Mileage (0:Low, 1:High)

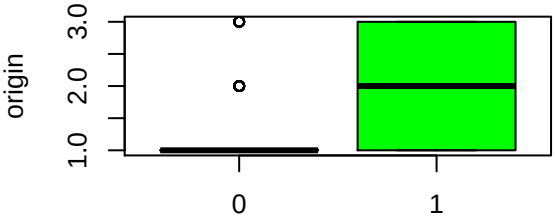

displacement
100
400
0
1
Gas Mileage (0:Low, 1:High)


horsepower
50
150
0
1
Gas Mileage (0:Low, 1:High)









Observations:

Correlation plots and Boxplots are plotted to compare the overall distributions for each variables between cars with above-median mpg and those with below median-mpg.It can be observed that there is a strong negative correlation between mpg01 and Cylinders,Displacement,Weight and Horsepower.So these features are most likely to properly predict mpg01

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

Solution:

```
train <- (auto$year %% 2 == 0)
auto.train <- auto[train, ]
auto.test <- auto[!train, ]
```

d) Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in b). What is the test error of the model obtained?

Solution:

```
lda.fit2 = lda(mpg01~displacement+weight+cylinders+year, data=auto.train)
lda.fit2
```

```
## Call:
## lda(mpg01 ~ displacement + weight + cylinders + year, data = auto.train)
##
```

```
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   displacement   weight cylinders     year
## 0     271.7396 3604.823  6.812500 74.10417
## 1     111.6623 2314.763  4.070175 77.78947
##
## Coefficients of linear discriminants:
##                       LD1
## displacement  0.003661069
## weight       -0.001176417
## cylinders    -0.658136542
## year          0.080406655
```

```
pred.lda2 = predict(lda.fit2, newdata = auto.test, type="response")$class
table(pred.lda2, auto.test$mpg01)
```

```
##
## pred.lda2  0  1
##         0 87  7
##         1 13 75
```

```
mean(pred.lda2 != auto.test$mpg01)
```

```
## [1] 0.1098901
```

Observations:

It can be concluded that there is a test error rate of 10.98901%.

e) Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in b). What is the test error of the model obtained?

Solution:

```
qda.fit2 = qda(mpg01~displacement+weight+cylinders+year, data=auto ,subset=train)
qda.fit2
```

```
## Call:
## qda(mpg01 ~ displacement + weight + cylinders + year, data = auto,
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   displacement   weight cylinders     year
## 0     271.7396 3604.823  6.812500 74.10417
## 1     111.6623 2314.763  4.070175 77.78947
```

```r
pred.qda2 = predict(qda.fit2, newdata = auto.test, type="response")$class
table(pred.qda2, auto.test$mpg01)
```

```
##
## pred.qda2  0  1
##         0 89 11
##         1 11 71
```

```r
mean(pred.qda2 != auto.test$mpg01)
```

```
## [1] 0.1208791
```

Observations:

It can be concluded that there is a test error rate of 12.08791%.

f)Perform logistic regression on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in b).What is the test error of the model obtained?

Solution:

```r
logisticreg = glm(mpg01~displacement+weight+cylinders+year, family="binomial", data=auto)
summary(logisticreg)
```

```
##
## Call:
## glm(formula = mpg01 ~ displacement + weight + cylinders + year,
##     family = "binomial", data = auto)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.32519  -0.12750   0.01732   0.26133   3.13675
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.812e+01  4.664e+00  -3.886 0.000102 ***
## displacement -3.481e-03  9.345e-03  -0.373 0.709490
## weight       -4.359e-03  8.823e-04  -4.941 7.78e-07 ***
## cylinders    -2.770e-01  3.824e-01  -0.724 0.468813
## year          4.259e-01  7.263e-02   5.864 4.52e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 543.43  on 391  degrees of freedom
## Residual deviance: 165.20  on 387  degrees of freedom
## AIC: 175.2
##
## Number of Fisher Scoring iterations: 7
```

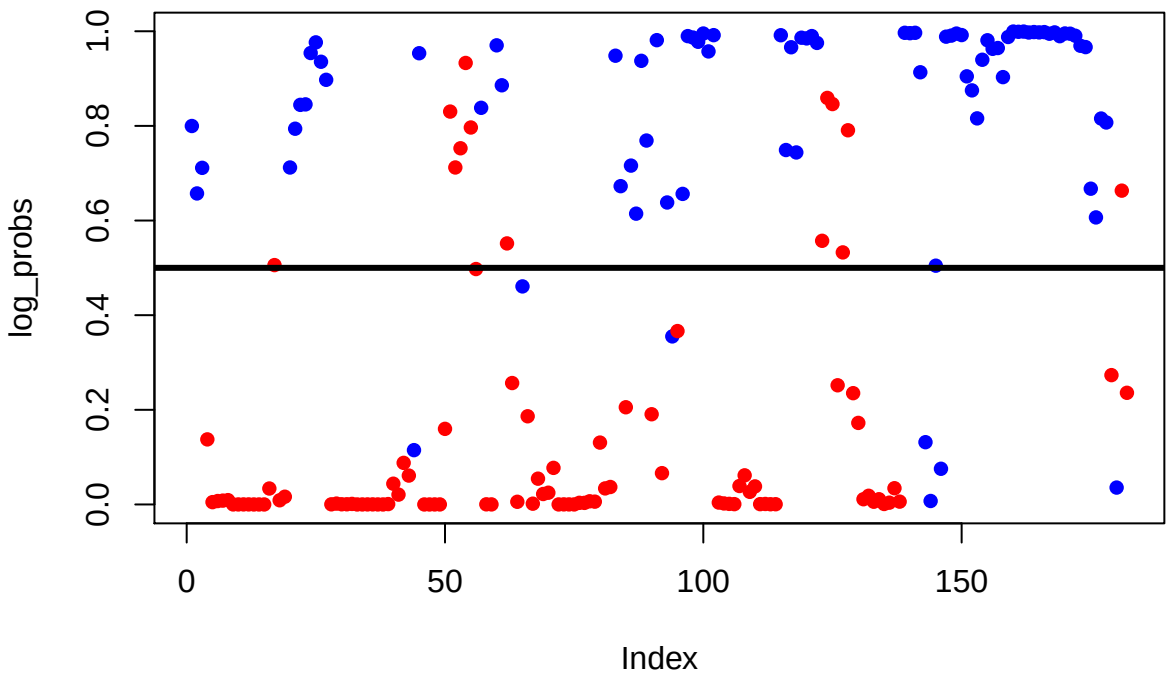
```
log_probs = predict(logisticreg, auto.test, type="response")
log_preds = rep(0, dim(auto.test)[1])
log_preds[log_probs>0.5]=1
table(log_preds,auto.test$mpg01 )
```

```
##
## log_preds  0  1
##         0 87  7
##         1 13 75
```

```
mean(log_preds != auto.test$mpg01)
```

```
## [1] 0.1098901
```

```
par(mfrow=c(1,1))
plot(log_probs, col= ifelse(auto.test$mpg01==0, "red", "blue"), pch = 16)
abline(h=0.5, lwd=3)
```



Observations:

It can be concluded that there is a test error rate of 10.98901%.

g) Perform KNN on the training data, with several values of K, in order to predict mpg01.Use only the variables that seemed most associated with mpg01 in (b). What test errors do you obtain? Which value of K seems to perform the best on this data set?

Solution:

```
train_auto_X <- cbind(auto$cylinders,
                      auto$displacement,
                      auto$horsepower,
                      auto$weight
                      )[train,]

test_auto_X <- cbind(
  auto$cylinders,
  auto$displacement,
  auto$horsepower,
  auto$weight
)[!train,]

set.seed(1)
knn.pred2 = knn(train_auto_X, test_auto_X, auto.train$mpg01, k=1)
tab = table(knn.pred2, auto.test$mpg01)
error.rate = round((sum(tab[1,2], tab[2,1])/dim(auto.test)[1])*100, 2)
paste("The error rate is of", error.rate, "%")
```

```
## [1] "The error rate is of 15.38 %"
```

```
knn.pred2 = knn(train_auto_X, test_auto_X, auto.train$mpg01, k=10)
tab = table(knn.pred2, auto.test$mpg01)
error.rate = round((sum(tab[1,2], tab[2,1])/dim(auto.test)[1])*100, 2)
paste("The error rate is of", error.rate, "%")
```

```
## [1] "The error rate is of 16.48 %"
```

```
knn.pred2 = knn(train_auto_X, test_auto_X, auto.train$mpg01, k=100)
tab = table(knn.pred2, auto.test$mpg01)
error.rate = round((sum(tab[1,2], tab[2,1])/dim(auto.test)[1])*100, 2)
paste("The error rate is of", error.rate, "%")
```

```
## [1] "The error rate is of 14.29 %"
```

Observations:

The error rate is 14.29% for K=100 so this value seems to perform the best on this data set

Problem 3

a) Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations.

Solution:

```
set.seed(15)
tr <- sample(1:nrow(OJ), 800)
oj.tr <- OJ[tr,]
oj.te <- OJ[-tr,]
```

b) Fit a tree to the training data, with Purchase as the response and the other variables as predictors. Use the summary() function to produce summary statistics about the tree, and describe the results obtained. What is the training error rate? How many terminal nodes does the tree have?

Solution:

```
tree.oj <- tree(Purchase ~., oj.tr)
summary(tree.oj)
```

```
##
## Classification tree:
## tree(formula = Purchase ~ ., data = oj.tr)
## Variables actually used in tree construction:
## [1] "LoyalCH"       "PriceDiff"     "ListPriceDiff"
## Number of terminal nodes:  8
## Residual mean deviance:  0.7283 = 576.8 / 792
## Misclassification error rate: 0.1562 = 125 / 800
```

Observations:

The tree uses 3 variables LoyalCH,PriceDiff and ListPriceDiff for construction.It has 8 terminal nodes.Training error rate (misclassification error) for the tree is 0.1562

c) Type in the name of the tree object in order to get a detailed text output. Pick one of the terminal nodes, and interpret the information displayed.

Solution:

```
tree.oj
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 800 1074.00 CH ( 0.60375 0.39625 )
##    2) LoyalCH < 0.5036 353  420.80 MM ( 0.28329 0.71671 )
##      4) LoyalCH < 0.276142 174  120.00 MM ( 0.10920 0.89080 )
##        8) LoyalCH < 0.0356415 62   10.24 MM ( 0.01613 0.98387 ) *
##        9) LoyalCH > 0.0356415 112   98.75 MM ( 0.16071 0.83929 ) *
##      5) LoyalCH > 0.276142 179  246.50 MM ( 0.45251 0.54749 )
##       10) PriceDiff < 0.065 73   79.24 MM ( 0.23288 0.76712 ) *
##       11) PriceDiff > 0.065 106  142.30 CH ( 0.60377 0.39623 ) *
##    3) LoyalCH > 0.5036 447  367.20 CH ( 0.85682 0.14318 )
##      6) LoyalCH < 0.753545 180  219.90 CH ( 0.70000 0.30000 )
##       12) PriceDiff < 0.265 106  146.60 CH ( 0.52830 0.47170 )
##         24) ListPriceDiff < 0.235 61   79.76 MM ( 0.36066 0.63934 ) *
##         25) ListPriceDiff > 0.235 45   50.05 CH ( 0.75556 0.24444 ) *
##       13) PriceDiff > 0.265 74   31.12 CH ( 0.94595 0.05405 ) *
##      7) LoyalCH > 0.753545 267   85.31 CH ( 0.96255 0.03745 ) *
```

Observations:

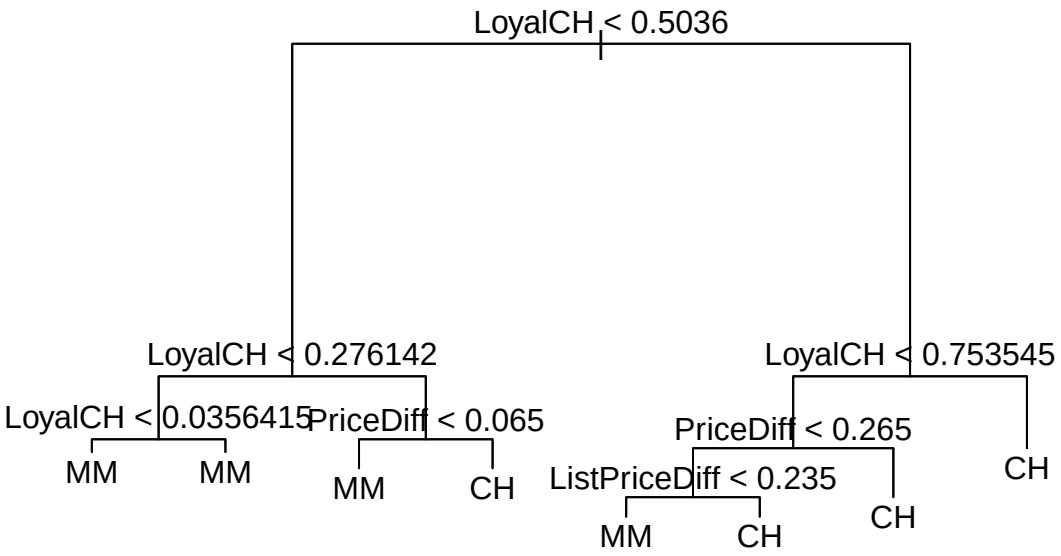
**Node 11 is a terminal node as indicated by '*'.The split criteria is PriceDiff is split on $>$ 0.065.There are 106 points in this terminal node.The deviance is 142.30.The prediction at this node is CH. About 60.37% of

the observations in this node have CH as value of Sales. The remaining 39.62% points have MM as value of Sales.**

d) Create a plot of the tree, and interpret the results

Solution:

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



Observations:

Based on the plot above,the most important splitting variables is LoyalCH as first 4 splits are based on this variable.If LoyalCH<0.035,the tree predicts MM and if LoyalCH>0.753,the tree predicts CH.For intermediate values of LoyalCH,the decision depends on the value of PriceDiff and ListPriceDiff

e)Predict the response on the test data, and produce a confusion matrix comparing the test labels to the predicted test labels. What is the test error rate?

Solution:

```
oj.pred <-  predict(tree.oj, oj.te, type="class")
table(oj.te$Purchase,oj.pred)
```

```
##     oj.pred
##       CH  MM
##   CH 137  33
##   MM  29  71
```

```r
mean( oj.te$Purchase == oj.pred)
```

```
## [1] 0.7703704
```

Observations: **The test error rate observed is 22.9%.Also, 77% of the observations are classified correctly.**

f) Apply the cv.tree() function to the training set in order to determine the optimal tree size.

Solution:

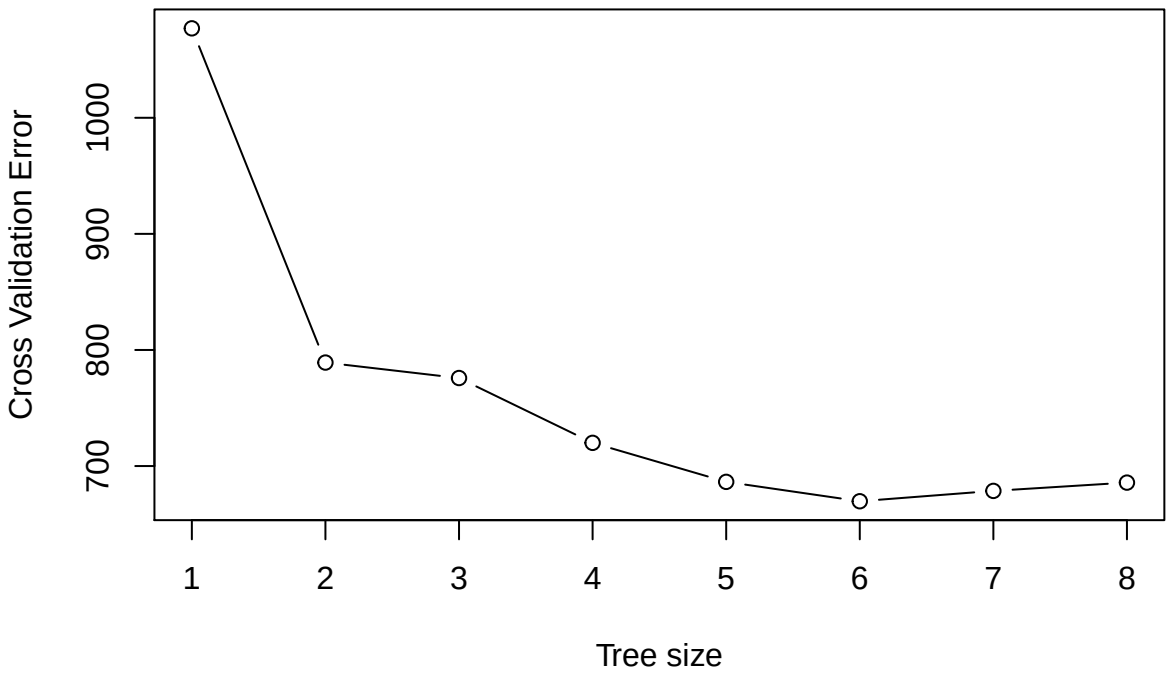
```r
set.seed(50)
cv.oj <- cv.tree(tree.oj, FUN = prune.tree)
cv.oj
```

```
## $size
## [1] 8 7 6 5 4 3 2 1
##
## $dev
## [1]  685.7503  678.6075  669.6925  686.3934  720.0006  775.8217  789.1557
## [8] 1077.0366
##
## $k
## [1]      -Inf  11.01178  16.79105  24.94349  42.18181  54.26682  61.93042
## [8] 286.38608
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

g) Produce a plot with tree size on the x-axis and cross-validated classification error rate on the y-axis.

Solution:

```r
plot(cv.oj$size, cv.oj$dev, type = "b", xlab = "Tree size", ylab = "Cross Validation Error")
```



Observations:

Based on the plot, tree of size 6 has the lowest cross validation error

Problem 4

a)Create a training set consisting of the first 1,000 observations, and a test set consisting of the remaining observations.

Solution:

```
set.seed(1)
train <- 1:1000
Caravan$Purchase <- ifelse(Caravan$Purchase == "Yes", 1, 0)
Caravan.train <- Caravan[train, ]
Caravan.test <- Caravan[-train, ]
```

b) Fit a boosting model to the training set with Purchase as the response and the other variables as predictors. Use 1,000 trees, and a shrinkage value of 0.01. Which predictors appear to be the most important?

Solution:

```
set.seed(1)
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.0.3
```

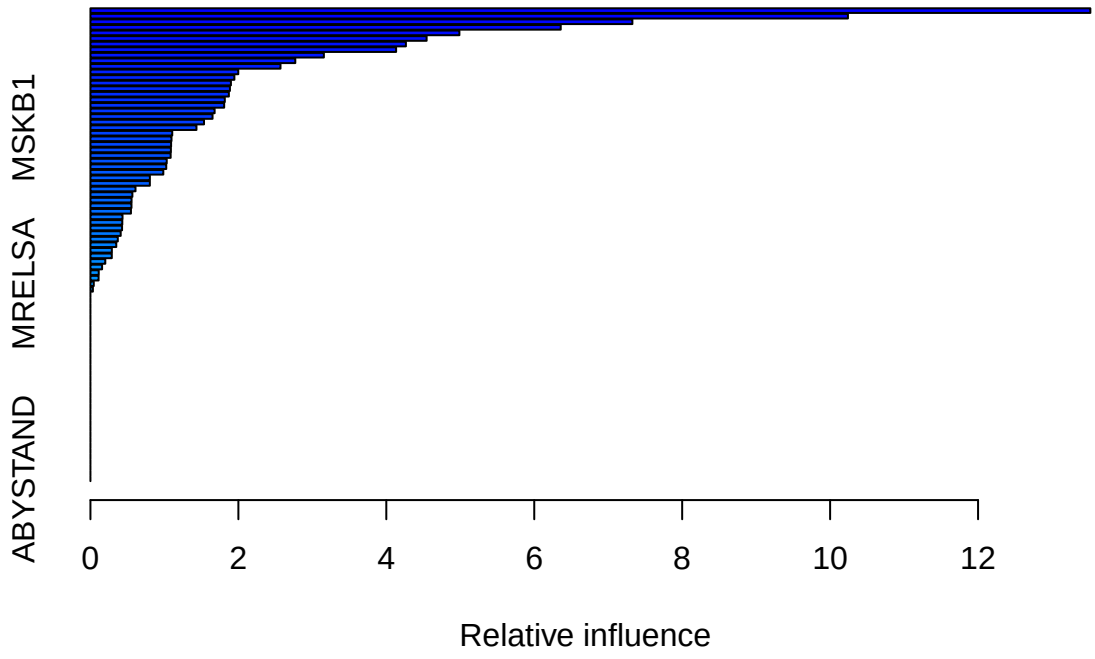
```
## Loaded gbm 2.1.8
```

```
boost.caravan <- gbm(Purchase ~ ., data = Caravan.train, distribution = "gaussian", n.trees = 1000, shr
```

```
## Warning in gbm.fit(x = x, y = y, offset = offset, distribution = distribution, :
## variable 50: PVRAAUT has no variation.
```

```
## Warning in gbm.fit(x = x, y = y, offset = offset, distribution = distribution, :
## variable 71: AVRAAUT has no variation.
```

```
summary(boost.caravan)
```




```
##                var     rel.inf
## PPERSAUT PPERSAUT 13.51824557
## MKOOPKLA MKOOPKLA 10.24062778
## MOPLHOOG MOPLHOOG  7.32689780
## MBERMIDD MBERMIDD  6.35820558
## PBRAND     PBRAND  4.98826360
## ABRAND     ABRAND  4.54504653
## MGODGE     MGODGE  4.26496875
## MINK3045 MINK3045  4.13253907
## PWAPART   PWAPART  3.15612877
## MAUT1       MAUT1  2.76929763
## MOSTYPE   MOSTYPE  2.56937935
```

```
## MAUT2       MAUT2  1.99879666
## MSKA         MSKA  1.94618539
## MBERARBG MBERARBG  1.89917331
## PBYSTAND PBYSTAND  1.88591514
## MINKGEM   MINKGEM  1.87131472
## MGODOV     MGODOV  1.81673309
## MGODPR     MGODPR  1.80814745
## MFWEKIND MFWEKIND  1.67884570
## MSKC         MSKC  1.65075962
## MBERHOOG MBERHOOG  1.53559951
## MSKB1       MSKB1  1.43339514
## MOPLMIDD MOPLMIDD  1.10617074
## MHHUUR     MHHUUR  1.09608784
## MRELGE     MRELGE  1.09039794
## MINK7512 MINK7512  1.08772012
## MZFONDS   MZFONDS  1.08427551
## MGODRK     MGODRK  1.03126657
## MINK4575 MINK4575  1.02492795
## MZPART     MZPART  0.98536712
## MRELOV     MRELOV  0.80356854
## MFGEKIND MFGEKIND  0.80335689
## MBERARBO MBERARBO  0.60909852
## APERSAUT APERSAUT  0.56707821
## MGEMOMV   MGEMOMV  0.55589456
## MOSHOOFD MOSHOOFD  0.55498375
## MAUT0       MAUT0  0.54748481
## PMOTSCO   PMOTSCO  0.43362597
## MSKB2       MSKB2  0.43075446
## MSKD         MSKD  0.42751490
## MINK123M MINK123M  0.40920707
## MINKM30   MINKM30  0.36996576
## MHKOOP     MHKOOP  0.34941518
## MBERBOER MBERBOER  0.28967068
## MFALLEEN MFALLEEN  0.28877552
## MGEMLEEF MGEMLEEF  0.20084195
## MOPLLAAG MOPLLAAG  0.15750616
## MBERZELF MBERZELF  0.11203381
## PLEVEN     PLEVEN  0.11030994
## MRELSA     MRELSA  0.04500507
## MAANTHUI MAANTHUI  0.03322830
## PWABEDR   PWABEDR  0.00000000
## PWALAND   PWALAND  0.00000000
## PBESAUT   PBESAUT  0.00000000
## PVRAAUT   PVRAAUT  0.00000000
## PAANHANG PAANHANG  0.00000000
## PTRACTOR PTRACTOR  0.00000000
## PWERKT     PWERKT  0.00000000
## PBROM       PBROM  0.00000000
## PPERSONG PPERSONG  0.00000000
## PGEZONG   PGEZONG  0.00000000
## PWAOREG   PWAOREG  0.00000000
## PZEILPL   PZEILPL  0.00000000
## PPLEZIER PPLEZIER  0.00000000
## PFIETS     PFIETS  0.00000000
```

```
## PINBOED   PINBOED  0.00000000
## AWAPART   AWAPART  0.00000000
## AWABEDR   AWABEDR  0.00000000
## AWALAND   AWALAND  0.00000000
## ABESAUT   ABESAUT  0.00000000
## AMOTSCO   AMOTSCO  0.00000000
## AVRAAUT   AVRAAUT  0.00000000
## AAANHANG AAANHANG  0.00000000
## ATRACTOR ATRACTOR  0.00000000
## AWERKT     AWERKT  0.00000000
## ABROM       ABROM  0.00000000
## ALEVEN     ALEVEN  0.00000000
## APERSONG APERSONG  0.00000000
## AGEZONG   AGEZONG  0.00000000
## AWAOREG   AWAOREG  0.00000000
## AZEILPL   AZEILPL  0.00000000
## APLEZIER APLEZIER  0.00000000
## AFIETS     AFIETS  0.00000000
## AINBOED   AINBOED  0.00000000
## ABYSTAND ABYSTAND  0.00000000
```

Observations:

PPERSAUT and MKOOPKLA appear to be the most important predictors here.

c) Use the boosting model to predict the response on the test data. Predict that a person will make a purchase if the estimated probability of purchase is greater than 20 %. Form a confusion matrix. What fraction of the people predicted to make a purchase do in fact make one? How does this compare with the results obtained from applying KNN or logistic regression to this data set?

Solution:

```
probs.test <- predict(boost.caravan, Caravan.test, n.trees = 1000, type = "response")
pred.test <- ifelse(probs.test > 0.2, 1, 0)
table(Caravan.test$Purchase, pred.test)
```

```
##    pred.test
##        0    1
##   0 4493   40
##   1  278   11
```

Observations:

21.56% of people predicted to make purchase actually end up making one.(11/40+11)

```
logit.caravan <- glm(Purchase ~ ., data = Caravan.train, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
probs.test2 <- predict(logit.caravan, Caravan.test, type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
pred.test2 <- ifelse(probs.test > 0.2, 1, 0)
table(Caravan.test$Purchase, pred.test2)
```

```
##    pred.test2
##        0    1
##   0 4493   40
##   1  278   11
```

21.56% of people predicted to make purchase using logistic regression actually end up making one.This is in fact same as Boosting.