Name: Hari Yadav

Class: Intro to Stat

Date : 02/22/2020

Q2. This question should be answered using the “Weekly” data set, which is part of the “ISLR” package. This data is similar in nature to the “Smarket” data from this chapter’s lab, except that it contains 1089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

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

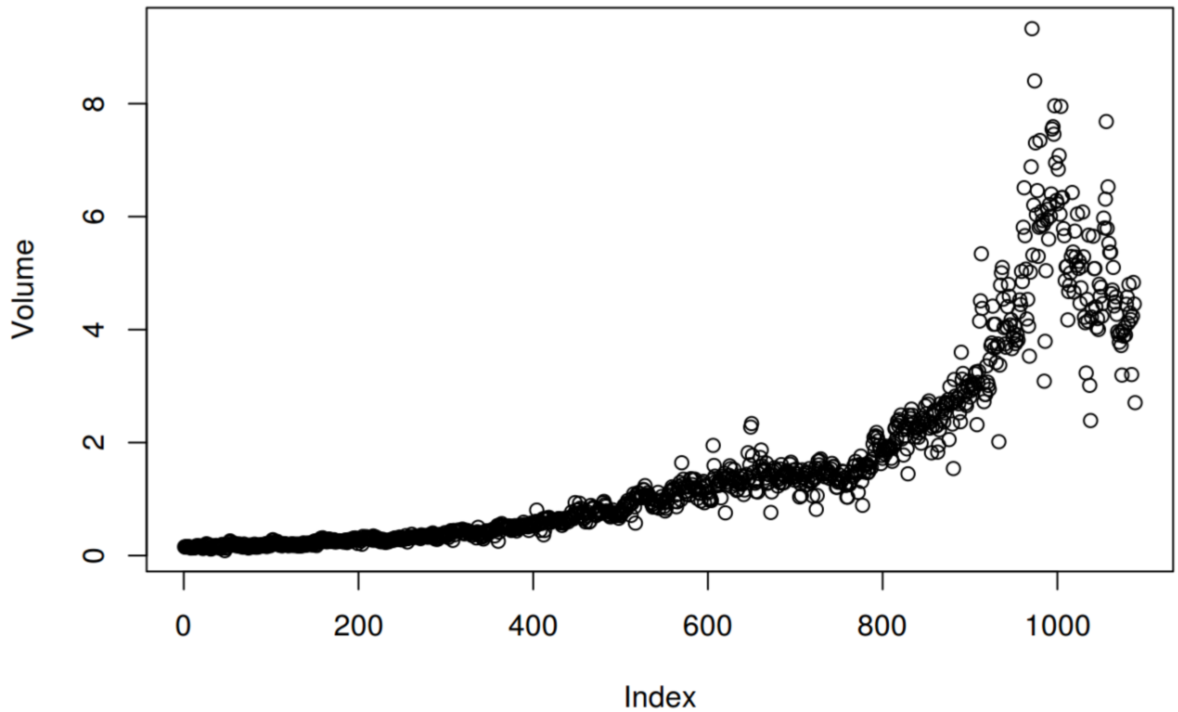
```
library(ISLR)
summary(Weekly)
```

```
##       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
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
##  Median :  0.2380   Median :  0.2340   Median :1.00268
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
##      Today          Direction
##  Min.   :-18.1950   Down:484
##  1st Qu.: -1.1540   Up  :605
##  Median :  0.2410
##  Mean   :  0.1499
##  3rd Qu.:  1.4050
##  Max.   : 12.0260
```

```
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
```

```
attach(Weekly)
plot(Volume)
```



The correlations between the "lag" variables and today's returns are close to zero. The only substantial correlation is between "Year" and "Volume". When we plot "Volume", we see that it is increasing over time.

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 ?

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

summary(fit.glm)
```

```
##
## 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
```

It would seem that "Lag2" is the only predictor statistically significant as its p-value is less than 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.

```
probs <- predict(fit.glm, type = "response")
pred.glm <- rep("Down", length(probs))
pred.glm[probs > 0.5] <- "Up"
table(pred.glm, Direction)
```

```
##         Direction
## pred.glm Down  Up
##     Down   54  48
##     Up    430 557
```

We may conclude that the percentage of correct predictions on the training data is (54+557)/1089(54+557)/1089 *wich is equal to 56.1065197%. In other words 43.8934803% is the training error rate, which is often overly optimistic. We could also say that for weeks when the market goes up, the model is right 92.0661157% of the time (*557/(48+557)557/(48+557)*). For weeks when the market goes down, the model is right only 11.1570248% of the time (*54/(54+430)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 to 2010).

```
train <- (Year < 2009)

Weekly.20092010 <- Weekly[!train, ]

Direction.20092010 <- Direction[!train]

fit.glm2 <- glm(Direction ~ Lag2, data = Weekly, family = binomial, subset = 
train)

summary(fit.glm2)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2, family = binomial, data = Weekly,
##     subset = train)
##
## 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
```

```
probs2 <- predict(fit.glm2, Weekly.20092010, type = "response")
pred.glm2 <- rep("Down", length(probs2))
pred.glm2[probs2 > 0.5] <- "Up"
table(pred.glm2, Direction.20092010)
```

```
##           Direction.20092010
## pred.glm2 Down Up
##      Down    9  5
##      Up     34 56
```

In this case, we may conclude that the percentage of correct predictions on the test data is $(9+56)/104$ *wich is equal to 62.5%. In other words 37.5% is the test error rate. 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.

```
library(MASS)
fit.lda <- lda(Direction ~ Lag2, data = Weekly, subset = train)
fit.lda
```

```
## Call:
## lda(Direction ~ Lag2, data = Weekly, subset = train)
##
## 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
pred.lda <- predict(fit.lda, Weekly.20092010)
table(pred.lda$class, Direction.20092010)
##       Direction.20092010
##        Down Up
##   Down    9  5
##   Up     34 56
```

In this case, we may conclude that the percentage of correct predictions on the test data is 62.5%. In other words 37.5% is the test error rate. We could also say that for weeks when the market goes up, the model is right 91.8032787% of the time. For weeks when the market goes down, the model is right only 20.9302326% of the time. These results are very close to those obtained with the logistic regression model which is not surpising.

f. Repeat (d) using QDA.

```
fit.qda <- qda(Direction ~ Lag2, data = Weekly, subset = train)
fit.qda
## Call:
## qda(Direction ~ Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
pred.qda <- predict(fit.qda, Weekly.20092010)
table(pred.qda$class, Direction.20092010)
##       Direction.20092010
##        Down Up
##   Down    0  0
##   Up     43 61
```

In this case, we may conclude that the percentage of correct predictions on the test data is 58.6538462%. In other words 41.3461538% is the test error rate. We could also say that 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 only 0% of the time. We may note, that QDA achieves a correctness of 58.6538462% even though the model chooses "Up" the whole time !

g. Repeat (d) using KNN with K=1K=1.

```
library(class)
train.X <- as.matrix(Lag2[train])
test.X <- as.matrix(Lag2[!train])
train.Direction <- Direction[train]
set.seed(1)
pred.knn <- knn(train.X, test.X, train.Direction, k = 1)
table(pred.knn, Direction.20092010)
```

```
##         Direction.20092010
## pred.knn Down Up
##     Down   21 30
##     Up     22 31
```

In this case, we may conclude that the percentage of correct predictions on the test data is 50%. In other words 50% is the test error rate. We could also say that 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 ?

If we compare the test error rates, we see that 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 KK in the KNN classifier.

```
# Logistic regression with Lag2:Lag1
fit.glm3 <- glm(Direction ~ Lag2:Lag1, data = Weekly, family = binomial, subs
et = train)
probs3 <- predict(fit.glm3, Weekly.20092010, type = "response")
pred.glm3 <- rep("Down", length(probs3))
```

```r
pred.glm3[probs3 > 0.5] = "Up"
table(pred.glm3, Direction.20092010)
```

```
##           Direction.20092010
## pred.glm3 Down Up
##      Down    1  1
##      Up     42 60
```

```r
mean(pred.glm3 == Direction.20092010)
```

```
## [1] 0.5865385
```

```r
# LDA with Lag2 interaction with Lag1
fit.lda2 <- lda(Direction ~ Lag2:Lag1, data = Weekly, subset = train)
pred.lda2 <- predict(fit.lda2, Weekly.20092010)
mean(pred.lda2$class == Direction.20092010)
```

```
## [1] 0.5769231
```

```r
# QDA with sqrt(abs(Lag2))
fit.qda2 <- qda(Direction ~ Lag2 + sqrt(abs(Lag2)), data = Weekly, subset = t
rain)
pred.qda2 <- predict(fit.qda2, Weekly.20092010)
table(pred.qda2$class, Direction.20092010)
```

```
##       Direction.20092010
##        Down Up
##   Down   12 13
##   Up     31 48
```

```r
mean(pred.qda2$class == Direction.20092010)
```

```
## [1] 0.5769231
```

```r
# KNN k =10
pred.knn2 <- knn(train.X, test.X, train.Direction, k = 10)
table(pred.knn2, Direction.20092010)
```

```
##           Direction.20092010
## pred.knn2 Down Up
##      Down   17 18
##      Up     26 43
```

```r
mean(pred.knn2 == Direction.20092010)
```

```
## [1] 0.5769231
```

```r
# KNN k = 100
pred.knn3 <- knn(train.X, test.X, train.Direction, k = 100)
```

```
table(pred.knn3, Direction.20092010)
##           Direction.20092010
## pred.knn3 Down Up
##      Down    9 12
##      Up     34 49
mean(pred.knn3 == Direction.20092010)
## [1] 0.5576923
```

Out of these combinations, the original logistic regression and LDA have the best performance in terms of test error rates.

Q3. In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the "Auto" data set.

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.

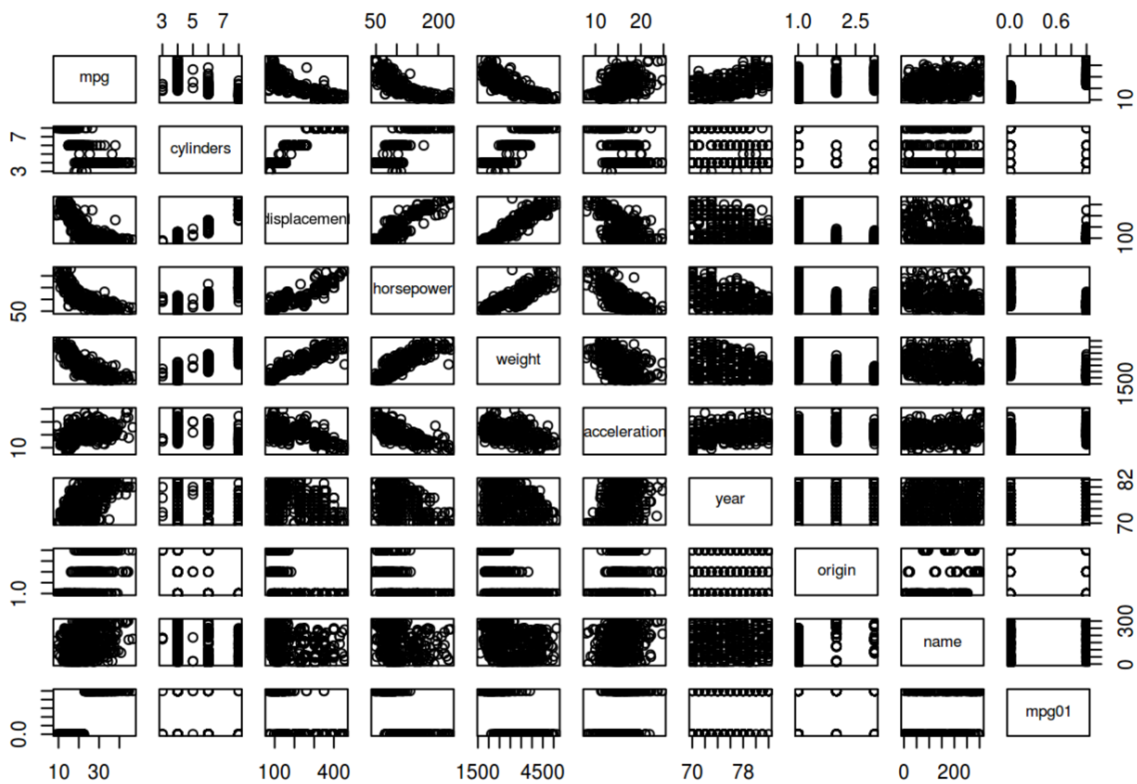
```
attach(Auto)
mpg01 <- rep(0, length(mpg))
mpg01[mpg > median(mpg)] <- 1
Auto <- data.frame(Auto, mpg01)
```

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 predictiong "mpg01" ? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

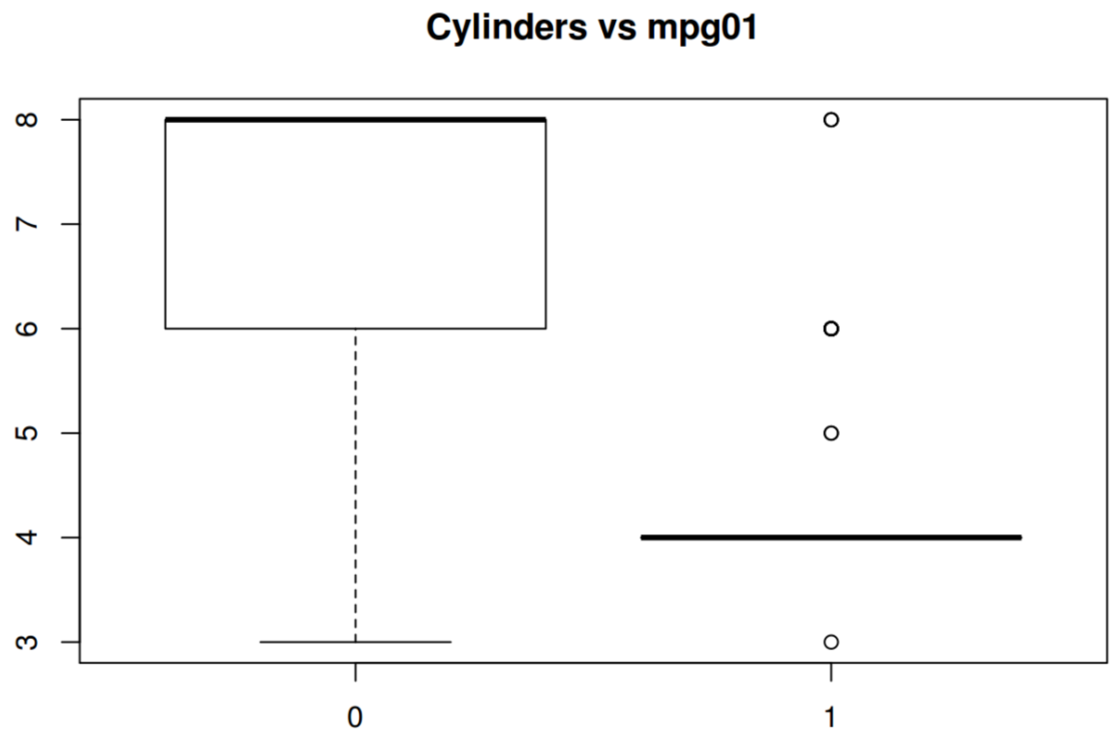
```
cor(Auto[, -9])
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  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
```

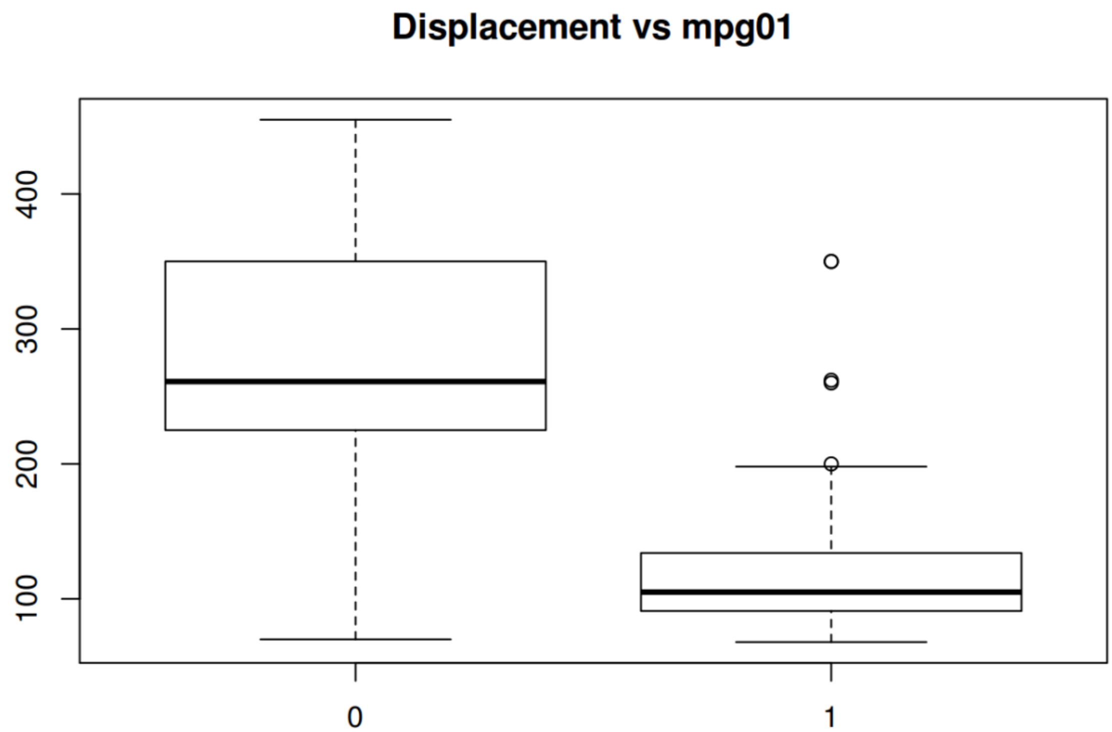
```
pairs(Auto)
```



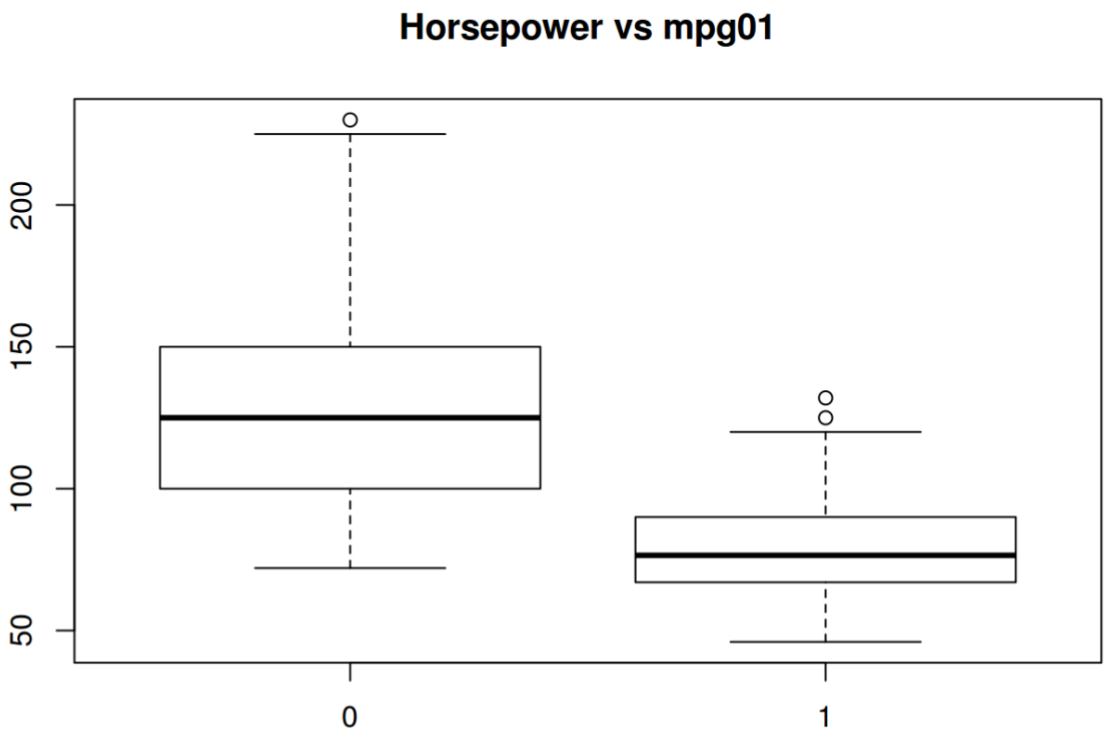

```
boxplot(cylinders ~ mpg01, data = Auto, main = "Cylinders vs mpg01")
```



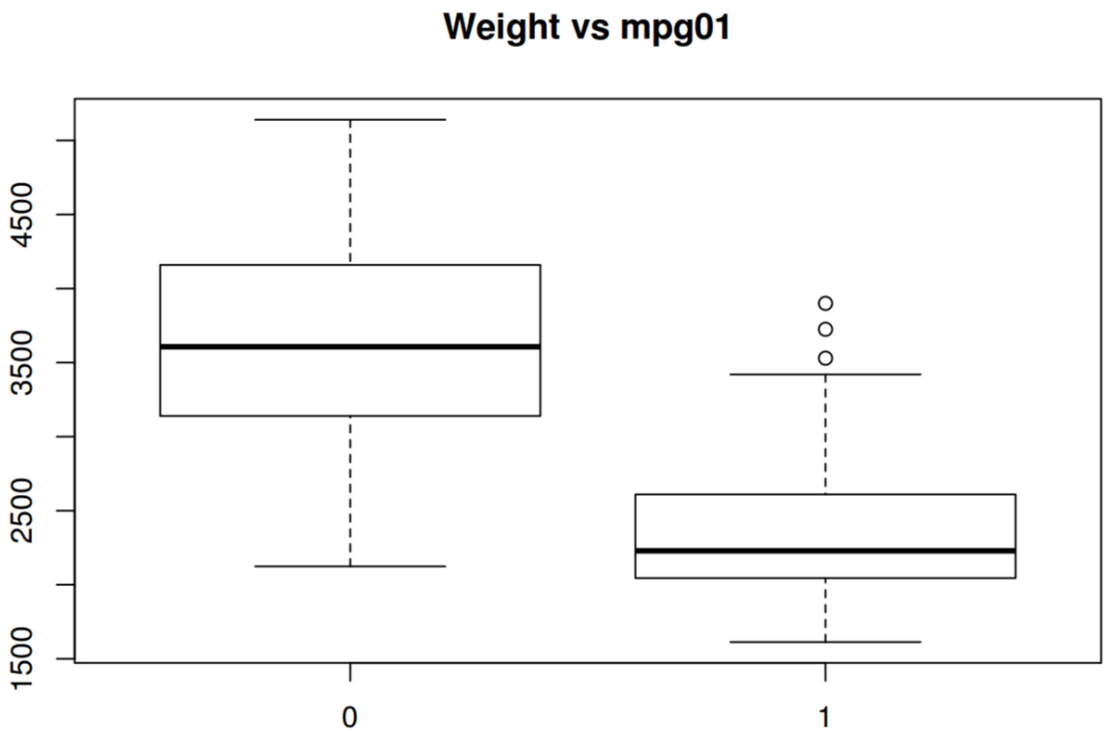

```
boxplot(displacement ~ mpg01, data = Auto, main = "Displacement vs mpg01")
```



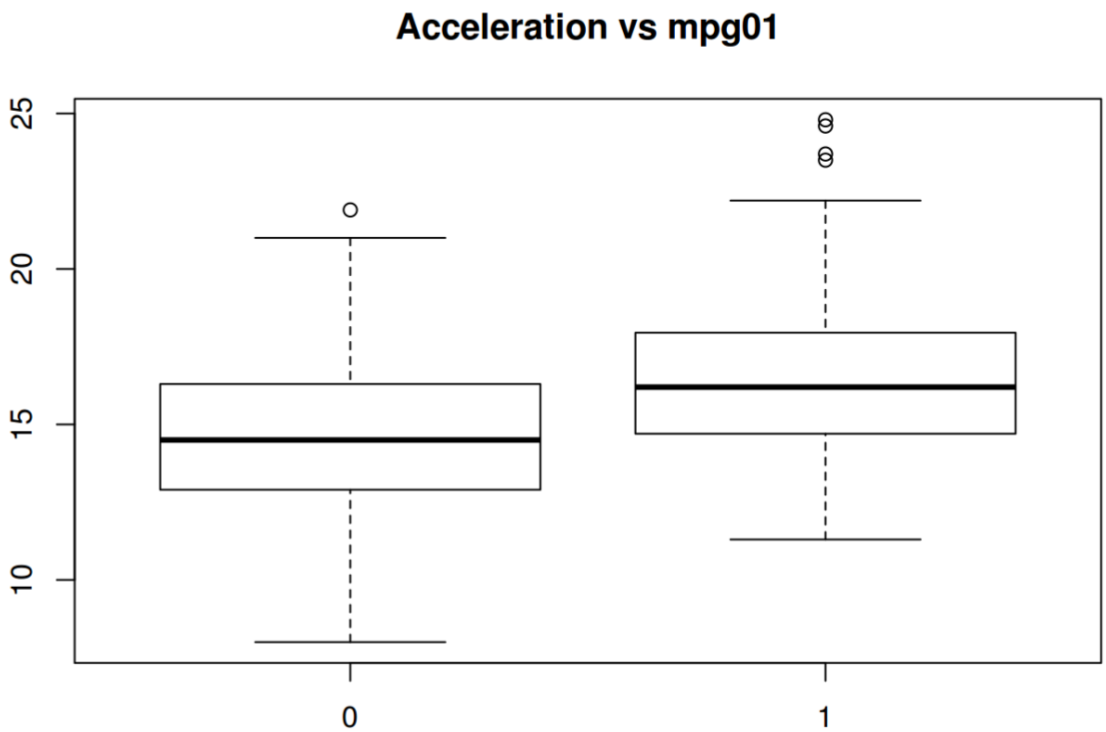

```r
boxplot(horsepower ~ mpg01, data = Auto, main = "Horsepower vs mpg01")
```



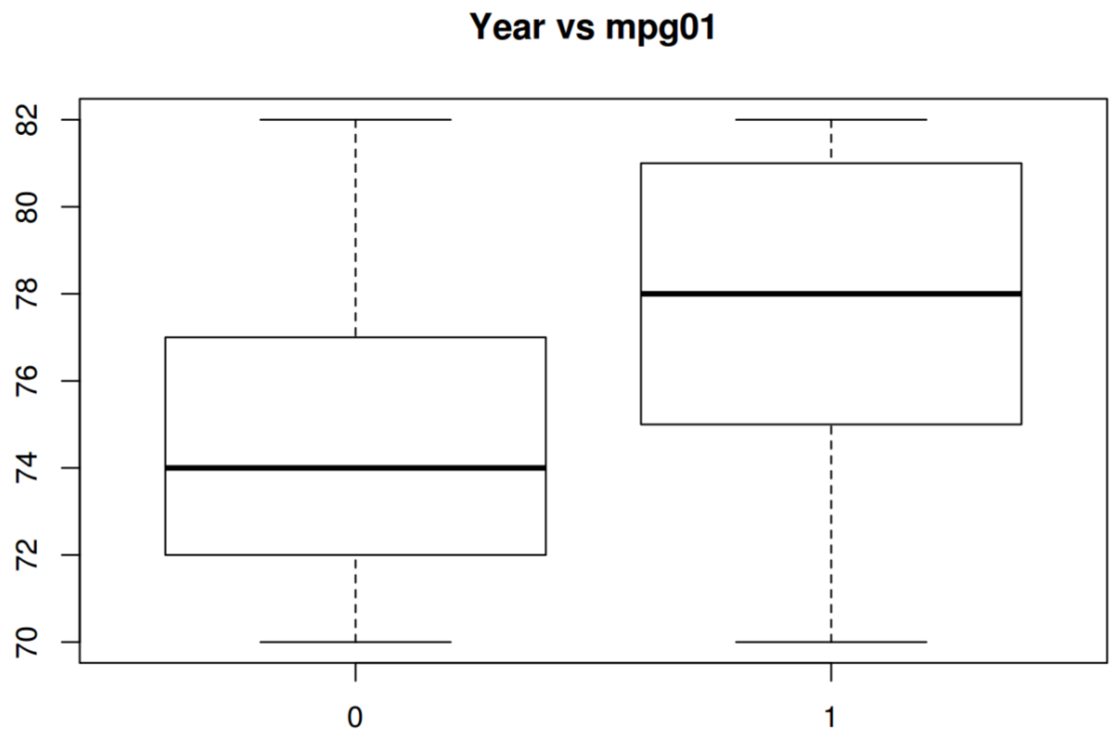

```
boxplot(weight ~ mpg01, data = Auto, main = "Weight vs mpg01")
```




```
boxplot(acceleration ~ mpg01, data = Auto, main = "Acceleration vs mpg01")
```




```r
boxplot(year ~ mpg01, data = Auto, main = "Year vs mpg01")
```



We may conclude that there exists some association between "mpg01" and "cylinders", "weight", "displacement" and "horsepower".

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

```
train <- (year %% 2 == 0)
Auto.train <- Auto[train, ]
Auto.test <- Auto[!train, ]
mpg01.test <- mpg01[!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 ?

```
fit.lda <- lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = 
Auto, subset = train)
fit.lda
```

```
## Call:
## lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto,
```

```
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders   weight displacement horsepower
## 0  6.812500 3604.823     271.7396  133.14583
## 1  4.070175 2314.763     111.6623   77.92105
##
## Coefficients of linear discriminants:
##                        LD1
## cylinders    -0.6741402638
## weight       -0.0011465750
## displacement  0.0004481325
## horsepower    0.0059035377
```

```
pred.lda <- predict(fit.lda, Auto.test)
table(pred.lda$class, mpg01.test)
```

```
##    mpg01.test
##      0  1
##   0 86  9
##   1 14 73
```

```
mean(pred.lda$class != mpg01.test)
```

```
## [1] 0.1263736
```

We may conclude that we have a test error rate of 12.6373626%.

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 ?

```
fit.qda <- qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = 
Auto, subset = train)
fit.qda
```

```
## Call:
## qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto,
```

```
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders   weight displacement horsepower
## 0  6.812500 3604.823     271.7396  133.14583
## 1  4.070175 2314.763     111.6623   77.92105
```

```
pred.qda <- predict(fit.qda, Auto.test)
table(pred.qda$class, mpg01.test)
```

```
##    mpg01.test
##      0  1
##   0 89 13
##   1 11 69
```

```
mean(pred.qda$class != mpg01.test)
```

```
## [1] 0.1318681
```

We may conclude that we have a test error rate of 13.1868132%.

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 ?

```
fit.glm <- glm(mpg01 ~ cylinders + weight + displacement + horsepower, data =
Auto, family = binomial, subset = train)
summary(fit.glm)
```

```
##
## Call:
## glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower,
##     family = binomial, data = Auto, subset = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.48027  -0.03413   0.10583   0.29634   2.57584
##
```

```
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  17.658730   3.409012   5.180 2.22e-07 ***
## cylinders    -1.028032   0.653607  -1.573   0.1158
## weight       -0.002922   0.001137  -2.569   0.0102 *
## displacement  0.002462   0.015030   0.164   0.8699
## horsepower   -0.050611   0.025209  -2.008   0.0447 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 289.58  on 209  degrees of freedom
## Residual deviance:  83.24  on 205  degrees of freedom
## AIC: 93.24
##
## Number of Fisher Scoring iterations: 7
```

```
probs <- predict(fit.glm, Auto.test, type = "response")
pred.glm <- rep(0, length(probs))
pred.glm[probs > 0.5] <- 1
table(pred.glm, mpg01.test)
```

```
##         mpg01.test
## pred.glm  0  1
##        0 89 11
##        1 11 71
```

```
mean(pred.glm != mpg01.test)
```

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

We may conclude that we have a test error rate of 12.0879121%.

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

```
train.X <- cbind(cylinders, weight, displacement, horsepower)[train, ]
test.X <- cbind(cylinders, weight, displacement, horsepower)[!train, ]
```

```
train.mpg01 <- mpg01[train]
set.seed(1)
pred.knn <- knn(train.X, test.X, train.mpg01, k = 1)
table(pred.knn, mpg01.test)
##          mpg01.test
## pred.knn  0  1
##         0 83 11
##         1 17 71
mean(pred.knn != mpg01.test)
## [1] 0.1538462
```

We may conclude that we have a test error rate of 15.3846154% for $K=1$*.*

```
pred.knn <- knn(train.X, test.X, train.mpg01, k = 10)
table(pred.knn, mpg01.test)
##          mpg01.test
## pred.knn  0  1
##         0 77  7
##         1 23 75
mean(pred.knn != mpg01.test)
## [1] 0.1648352
```

We may conclude that we have a test error rate of 16.4835165% for $K=10$*.*

```
pred.knn <- knn(train.X, test.X, train.mpg01, k = 100)
table(pred.knn, mpg01.test)
##          mpg01.test
## pred.knn  0  1
##         0 81  7
##         1 19 75
mean(pred.knn != mpg01.test)
## [1] 0.1428571
```

We may conclude that we have a test error rate of 14.2857143% for $K=100$*. So, a* K *value of 100 seems to perform the best.*