Jingtang Ma
CS465
Lab3 Classification

1. View the videos at the following URLs https://www.youtube.com/watch?v=TxvEVc8YNlU https://www.youtube.com/watch?v=2cl7JiPzkBY https://www.youtube.com/watch?v=9TVVF7CS3F4
You may download the R Code for Labs and the Data Sets to use from the textbook website. http://www-bcf.usc.edu/˜gareth/ISL/

2. 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 chapters lab, except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

(a) (5 points) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

Ans: According to the plot, we can easily find that year and volume have some relation.

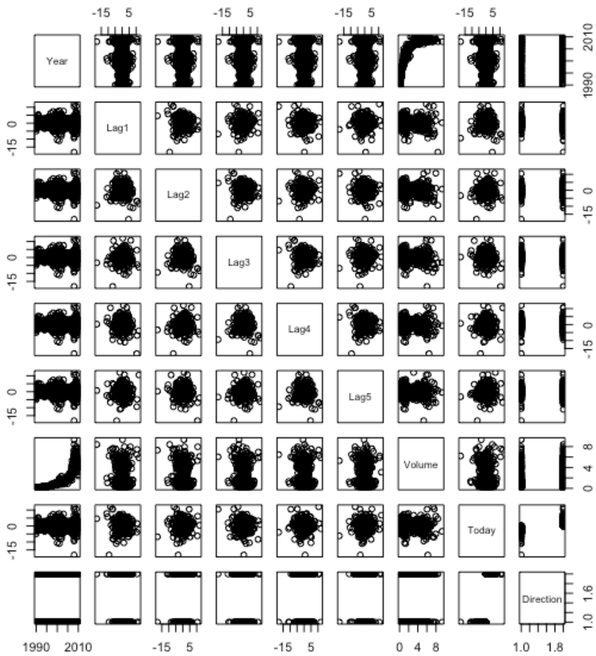
CODE:

```
require(ISLR)

### Q2
# A
attach(Weekly)
summary(Weekly)
pairs(Weekly)
```

OUTPUT:

```
      Year          Lag1               Lag2
 Min.   :1990   Min.   :-18.1950   Min.   :-18.1950
 1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540
 Median :2000   Median :  0.2410   Median :  0.2410
 Mean   :2000   Mean   :  0.1506   Mean   :  0.1511
 3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090
 Max.   :2010   Max.   : 12.0260   Max.   : 12.0260
      Lag3               Lag4
 Min.   :-18.1950   Min.   :-18.1950
 1st Qu.: -1.1580   1st Qu.: -1.1580
 Median :  0.2410   Median :  0.2380
 Mean   :  0.1472   Mean   :  0.1458
 3rd Qu.:  1.4090   3rd Qu.:  1.4090
 Max.   : 12.0260   Max.   : 12.0260
      Lag5              Volume            Today
 Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950
 1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540
 Median :  0.2340   Median :1.00268   Median :  0.2410
 Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499
 3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050
 Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260
 Direction
 Down:484
 Up  :605
```



(b) (5 points)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?

Ans: According to the summary I showed, we can find P value is leas than 5% that is Lag2, which means Lag2 is a predictor statistically significant.
CODE:

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

OUTPUT:

```
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
```

(c) (5 points) 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.

Ans: According to the confusion matrix, it shows most of the cases to go UP. And we know mean is 0.56 but there are 430 mistakes in Up.
CODE:

```
# C
glm.probs = predict(glm.fit, type = "response")
glm.pred = ifelse(glm.probs>0.5, "Up", "Down")
table(glm.pred, Direction)
mean(glm.pred==Direction)
```

OUTPUT:

```
> table(glm.pred, Direction)
        Direction
glm.pred Down  Up
    Down   54  48
    Up    430 557
> mean(glm.pred==Direction)
[1] 0.5610652
>
```

(d) (5 points) 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).

CODE:

```
# D
training.data = Year<=2008
test.data = Year>2008
glm.fit2 = glm(Direction~Lag2, data= Weekly, family = binomial, subset = training.data)
summary(glm.simp)

glm.testprobs = predict(glm.simp, Weekly[test.data, ], type = "response")
glm.testpred = ifelse(glm.testprobs>0.5, "Up", "Down")
table(glm.testpred, Direction[test.data])
mean(glm.testpred==Direction[test.data])
```

OUTPUT:

```
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

> glm.testprobs = predict(glm.simp, Weekly[test.data, ], type = "response")
> glm.testpred = ifelse(glm.testprobs>0.5, "Up", "Down")
> table(glm.testpred, Direction[test.data])

glm.testpred Down Up
        Down    9  5
        Up     34 56
> mean(glm.testpred==Direction[test.data])
[1] 0.625
>
```

(e) (5 points) Repeat (d) using LDA.

CODE:

```
# E
library(MASS)
lda.fit = lda(Direction ~ Lag2, data = Weekly, subset = training.data)
lda.fit
summary(lda.fit)

lda.pred = predict(lda.fit, Weekly[test.data, ])
table(lda.pred$class, Direction[test.data])
```

OUTPUT:

```
Call:
lda(Direction ~ Lag2, data = Weekly, subset = 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
> summary(lda.fit)
        Length Class  Mode
prior   2      -none- numeric
counts  2      -none- numeric
means   2      -none- numeric
scaling 1      -none- numeric
lev     2      -none- character
svd     1      -none- numeric
N       1      -none- numeric
call    4      -none- call
terms   3      terms  call
xlevels 0      -none- list
> lda.pred = predict(lda.fit, Weekly[test.data, ])
> table(lda.pred$class, Direction[test.data])

       Down Up
  Down    9  5
  Up     34 56
>
```

(f) (5 points) Repeat (d) using QDA.

CODE:

```
# E
library(MASS)
lda.fit = lda(Direction ~ Lag2, data = Weekly, subset = training.data)
lda.fit
summary(lda.fit)

lda.pred = predict(lda.fit, Weekly[test.data, ])
table(lda.pred$class, Direction[test.data])

# F
qda.fit <- qda(Direction ~ Lag2, data = Weekly, subset = training.data)
qda.fit
summary(qda.fit)
qda.pred <- predict(qda.fit, Weekly[test.data, ])
table(qda.pred$class, Direction[test.data])
```

OUTPUT:

```
> # F
> qda.fit <- qda(Direction ~ Lag2, data = Weekly, subset = training.data)
> qda.fit
Call:
qda(Direction ~ Lag2, data = Weekly, subset = training.data)

Prior probabilities of groups:
     Down        Up
0.4477157 0.5522843

Group means:
            Lag2
Down -0.03568254
Up    0.26036581
> summary(qda.fit)
        Length Class  Mode
prior   2      -none- numeric
counts  2      -none- numeric
means   2      -none- numeric
scaling 2      -none- numeric
ldet    2      -none- numeric
lev     2      -none- character
N       1      -none- numeric
call    4      -none- call
terms   3      terms  call
xlevels 0      -none- list
> qda.pred <- predict(qda.fit, Weekly[test.data, ])
> table(qda.pred$class, Direction[test.data])

       Down Up
  Down    0  0
  Up     43 61
```

(g) (5 points) Repeat (d) using KNN with K = 1.

CODE:

```
# G
library(class)
training.X = as.matrix(Lag2[training.data])
test.X = as.matrix(Lag2[test.data])
training.Direction = Direction[training.data]
set.seed(1)
knn.pred = knn(training.X, test.X, training.Direction, k = 1)
table(knn.pred, Direction[test.data])
```

OUTPUT:

```
knn.pred Down Up
    Down   21 30
    Up     22 31
>
```

(h) (5 points) Which of these methods appears to provide the best results on this data?

Ans: There are four methods: Logistic regression, LDA, QDA, and KNN. Depends on the test error rates, I think Logistic regression and LDA are better than the others. They both have 62.5% rate of true.

```
> mean(glm.testpred==Direction[test.data])
[1] 0.625
> mean(lda.pred$class==Direction[test.data])
[1] 0.625
> mean(qda.pred$class==Direction[test.data])
[1] 0.5865385
> mean(knn.pred$class==Direction[test.data])
Error in knn.pred$class : $ operator is invali
> mean(knn.pred==Direction[test.data])
[1] 0.5
>
```

(i) (5 points) 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.

Ans: There are four different methods: Logistic regression(Lag3), LDA(Lag3), QDA(Sqrt and abs), and KNN(k=5). Depends on the test error rates, I think original Logistic regression and original LDA are the best. They both have 62.5% rate of true.

CODE&OUTPUT:

```
> # I
> ##########Logistic regression: Lag2:Lag3
> glm.fit3 = glm(Direction ~ Lag2:Lag3, data = Weekly, family = binomial, subset = training.data)
> glm.probs3 = predict(glm.fit3, Weekly[test.data, ], type = "response")
> glm.pred3 = ifelse(glm.probs3>0.5, "Up", "Down")
> table(glm.pred3, Direction[test.data])

glm.pred3 Down Up
       Up   43 61
> mean(glm.pred3 == Direction[test.data])
[1] 0.5865385
> ########### LDA: Lag2:Lag3
> lda.fit2 = lda(Direction ~ Lag2:Lag3, data = Weekly, subset = training.data)
> lda.pred2 = predict(lda.fit2, Weekly[test.data, ])
> mean(lda.pred2$class == Direction[test.data])
[1] 0.5865385
> ############ QDA: sqrt(abs(Lag2))
> qda.fit2 = qda(Direction ~ Lag2 + sqrt(abs(Lag2)), data = Weekly, subset = training.data)
> qda.pred2 = predict(qda.fit2, Weekly[test.data, ])
> table(qda.pred2$class, Direction[test.data])

       Down Up
  Down   12 13
  Up     31 48
> mean(qda.pred2$class == Direction[test.data])
[1] 0.5769231
> ############ KNN k = 5
> knn.pred2 = knn(training.X, test.X, training.Direction, k = 5)
> table(knn.pred2, Direction[test.data])

knn.pred2 Down Up
     Down   15 20
     Up     28 41
> mean(knn.pred2 == Direction[test.data])
[1] 0.5384615
>
```

**************************QUESTION2 END **********************

3. 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) (5 points) 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.

CODE:

```
### Q3
# A
attach(Auto)
mpg01 = rep(0, length(mpg))
mpg01[mpg > median(mpg)] = 1
Auto = data.frame(Auto, mpg01)
head(Auto)
```

OUTPUT:

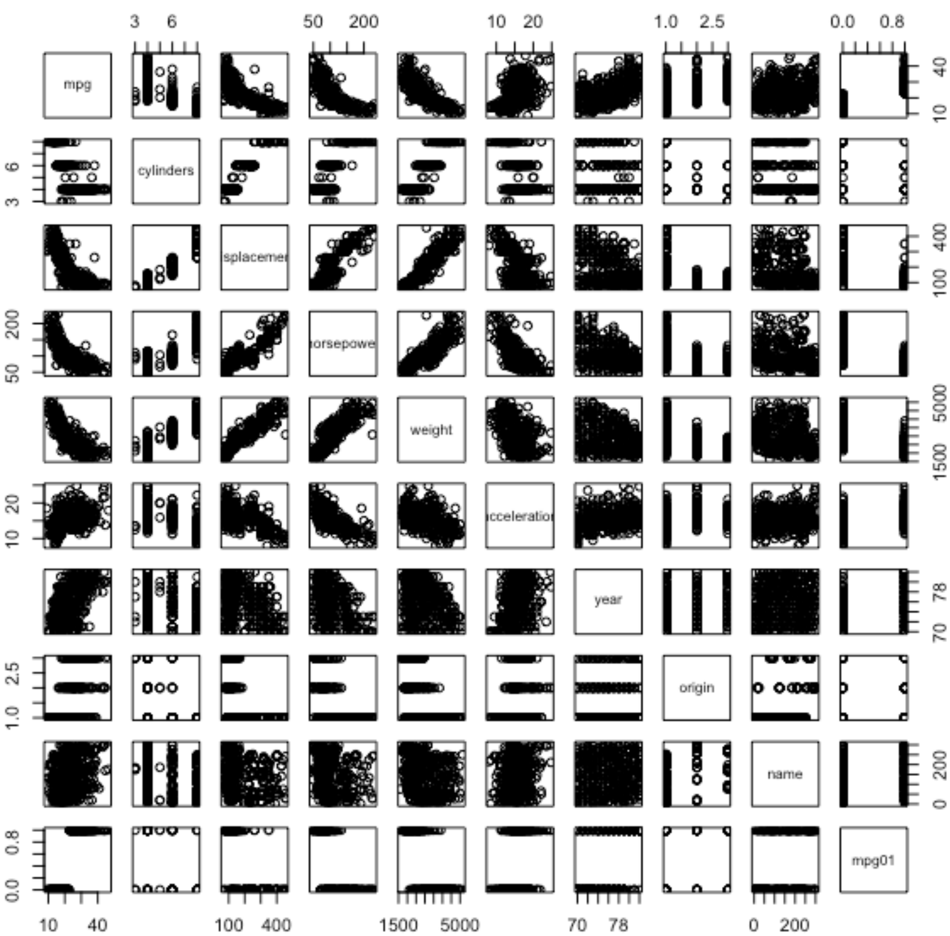
```
> head(Auto)
  mpg cylinders displacement horsepower weight
1  18         8          307        130   3504
2  15         8          350        165   3693
3  18         8          318        150   3436
4  16         8          304        150   3433
5  17         8          302        140   3449
6  15         8          429        198   4341
  acceleration year origin                      name
1         12.0   70      1 chevrolet chevelle malibu
2         11.5   70      1         buick skylark 320
3         11.0   70      1        plymouth satellite
4         12.0   70      1             amc rebel sst
5         10.5   70      1               ford torino
6         10.0   70      1          ford galaxie 500
  mpg01
1     0
2     0
3     0
4     0
5     0
6     0
> 
```

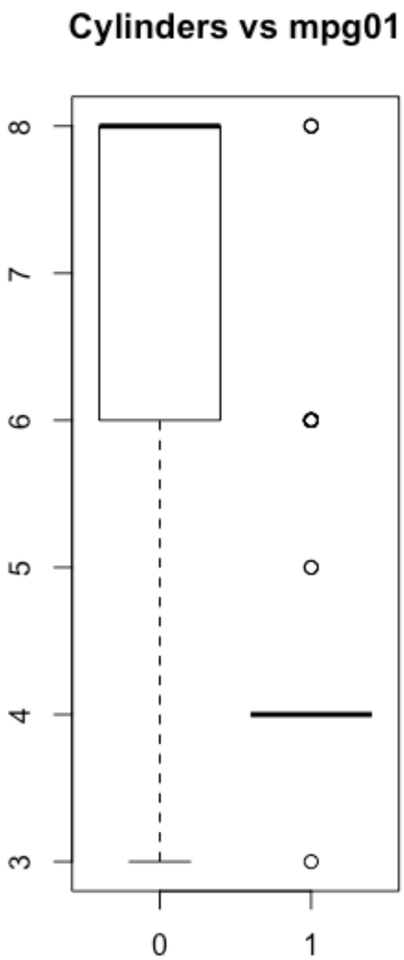
(b) (5 points) Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

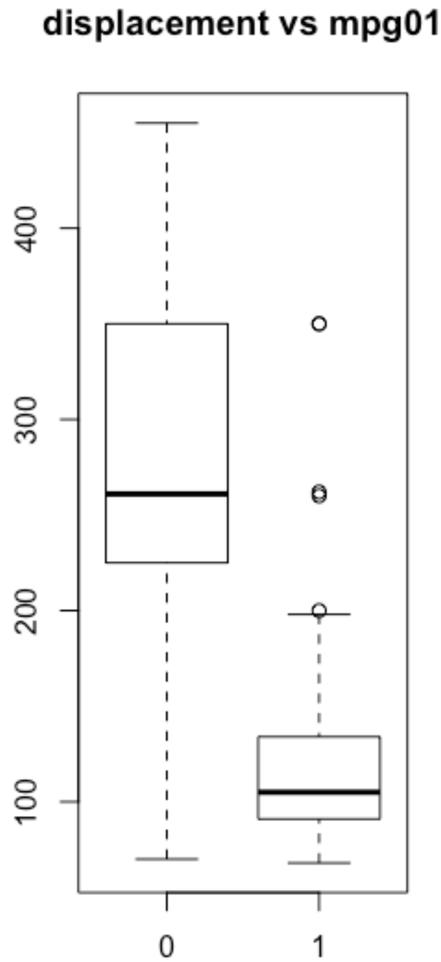
CODE:

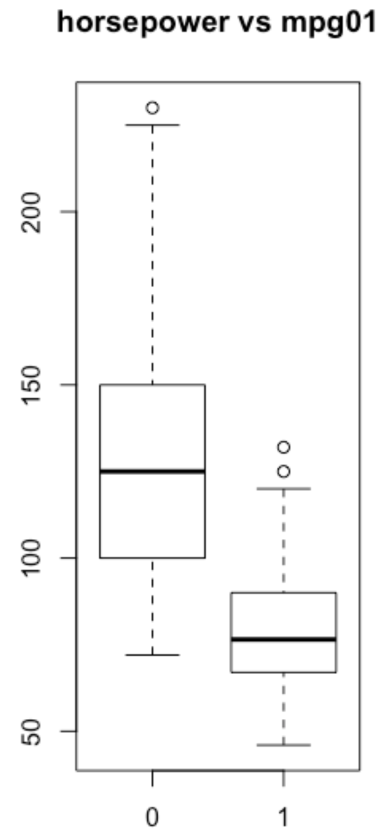
```
# B
pairs(Auto)
boxplot(cylinders ~ mpg01, data = Auto, main = "Cylinders vs mpg01")
boxplot(displacement ~ mpg01, data = Auto, main = "displacement vs mpg01")
boxplot(horsepower ~ mpg01, data = Auto, main = "horsepower vs mpg01")
boxplot(weight ~ mpg01, data = Auto, main = "weight vs mpg01")
boxplot(acceleration ~ mpg01, data = Auto, main = "acceleration vs mpg01")
boxplot(year ~ mpg01, data = Auto, main = "year vs mpg01")
boxplot(origin ~ mpg01, data = Auto, main = "origin vs mpg01")
```

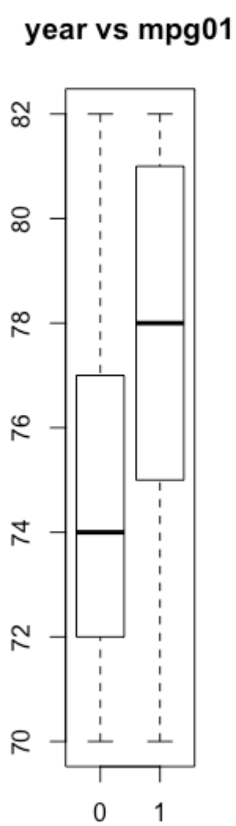
OUTPUT:

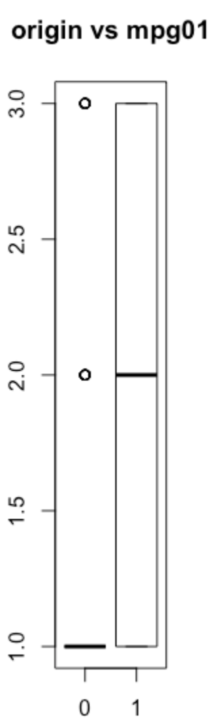


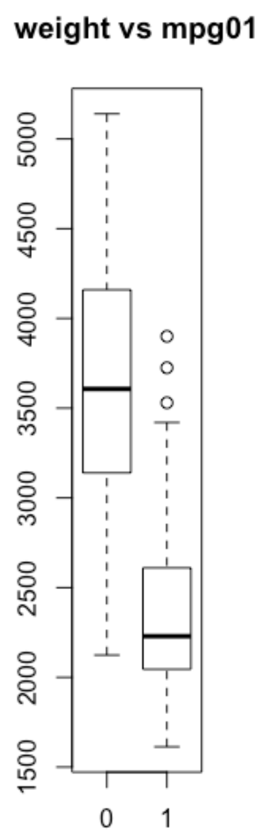

Cylinders vs mpg01
3
4
5
6
7
8
0
1

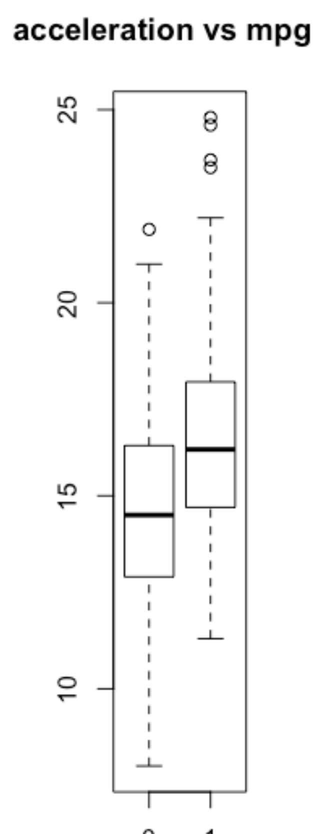

displacement vs mpg01
100
200
300
400
0
1


horsepower vs mpg01
50
100
150
200
0
1


year vs mpg01
70
72
74
76
78
80
82
0
1


origin vs mpg01
1.0
1.5
2.0
2.5
3.0
0
1


weight vs mpg01
1500
2000
2500
3000
3500
4000
4500
5000
0
1


acceleration vs mpg
10
15
20
25
0
1

Ans: According to the boxplots above, I think there are cylinders, horsepower, displacement and weight to be useful in predicting mpg01.

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

```
# C
trainid = (year %% 2 == 0)
train = Auto[trainid, ]
test = Auto[!trainid, ]
test.mpg01 = mpg01[!trainid]
```

(d) (5 points) 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?

Ans: The test error rate is 12.64%

CODE:

```
# D
lda.fit = lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = trainid)
lda.fit
lda.pred = predict(lda.fit, test)
table(lda.pred$class, test.mpg01)
lda.mean = mean(lda.pred$class == test.mpg01)
paste( "Rate of Test Error: " ,(1-lda.mean)*100, "%")
```

OUTPUT:

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

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
> lda.pred = predict(lda.fit, test)
> table(lda.pred$class, test.mpg01)
   test.mpg01
     0  1
  0 86  9
  1 14 73
> lda.mean = mean(lda.pred$class == test.mpg01)
> paste( "Rate of Test Error: " ,(1-lda.mean)*100, "%")
[1] "Rate of Test Error:  12.6373626373626 %"
> |
```

(e) (5 points) 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?

Ans: The test error rate is 13.19%

CODE:

```
# E
qda.fit = qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = trainid)
qda.fit
qda.pred = predict(qda.fit, test)
table(qda.pred$class, test.mpg01)
qda.mean = mean(qda.pred$class == test.mpg01)
paste( "Rate of Test Error: " ,(1-qda.mean)*100, "%")
```

OUTPUT:

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

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
> qda.pred = predict(qda.fit, test)
> table(qda.pred$class, test.mpg01)
   test.mpg01
     0  1
  0 89 13
  1 11 69
> qda.mean = mean(qda.pred$class == test.mpg01)
> paste( "Rate of Test Error: " ,(1-qda.mean)*100, "%")
[1] "Rate of Test Error:  13.1868131868132 %"
> 
```

(f) (5 points) 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?

Ans: The test error rate is 12.09%

CODE:

```
# F
glm.fit = glm(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, family = binomial, subset = trainid)
summary(glm.fit)
glm.probs = predict(glm.fit, test, type = "response")
glm.pred = rep(0, length(glm.probs))
glm.pred[glm.probs > 0.5] = 1
table(glm.pred, test.mpg01)
glm.mean = mean(glm.pred == test.mpg01)
paste( "Rate of Test Error: " ,(1-glm.mean)*100, "%")
```

OUTPUT:

```
Call:
glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower,
    family = binomial, data = Auto, subset = trainid)

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

> glm.probs = predict(glm.fit, test, type = "response")
> glm.pred = rep(0, length(glm.probs))
> glm.pred[glm.probs > 0.5] = 1
> table(glm.pred, test.mpg01)
        test.mpg01
glm.pred  0  1
       0 89 11
       1 11 71
> glm.mean = mean(glm.pred == test.mpg01)
> paste( "Rate of Test Error: " ,(1-glm.mean)*100, "%")
[1] "Rate of Test Error:  12.0879120879121 %"
>
```

(g) (5 points) 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?

Ans: I tried K=5, k=10, k=200. When K=5, the test error rate is 14.84%. When K=10, the test error rate is 16.48%. When K=200, the test error rate is 54.95%. To conclude that when K=5, it is the best to perform on this data set.

CODE:

```
# G
training.X = cbind(cylinders, weight, displacement, horsepower)[trainid, ]
test.X = cbind(cylinders, weight, displacement, horsepower)[!trainid, ]
training.mpg01 = mpg01[trainid]
set.seed(1)
####### K = 5
knn.pred = knn(training.X, test.X, training.mpg01, k = 5)
table(knn.pred, test.mpg01)
knn.mean = mean(knn.pred == test.mpg01)
paste( "Rate of Test Error: " ,(1-knn.mean)*100, "%")
####### K = 10
knn.pred = knn(training.X, test.X, training.mpg01, k = 10)
table(knn.pred, test.mpg01)
knn.mean = mean(knn.pred == test.mpg01)
paste( "Rate of Test Error: " ,(1-knn.mean)*100, "%")
####### K = 200
knn.pred = knn(training.X, test.X, training.mpg01, k = 200)
table(knn.pred, test.mpg01)
knn.mean = mean(knn.pred == test.mpg01)
paste( "Rate of Test Error: " ,(1-knn.mean)*100, "%")
```

OUTPUT:

```
> # G
> training.X = cbind(cylinders, weight, displacement, horsepower)[trainid, ]
> test.X = cbind(cylinders, weight, displacement, horsepower)[!trainid, ]
> training.mpg01 = mpg01[trainid]
> set.seed(1)
> ####### K = 5
> knn.pred = knn(training.X, test.X, training.mpg01, k = 5)
> table(knn.pred, test.mpg01)
        test.mpg01
knn.pred  0  1
       0 82  9
       1 18 73
> knn.mean = mean(knn.pred == test.mpg01)
> paste( "Rate of Test Error: " ,(1-knn.mean)*100, "%")
[1] "Rate of Test Error:  14.8351648351648 %"
> ####### K = 10
> knn.pred = knn(training.X, test.X, training.mpg01, k = 10)
> table(knn.pred, test.mpg01)
        test.mpg01
knn.pred  0  1
       0 77  7
       1 23 75
> knn.mean = mean(knn.pred == test.mpg01)
> paste( "Rate of Test Error: " ,(1-knn.mean)*100, "%")
[1] "Rate of Test Error:  16.4835164835165 %"
> ####### K = 200
> knn.pred = knn(training.X, test.X, training.mpg01, k = 200)
> table(knn.pred, test.mpg01)
        test.mpg01
knn.pred   0   1
       0   0   0
       1 100  82
> knn.mean = mean(knn.pred == test.mpg01)
> paste( "Rate of Test Error: " ,(1-knn.mean)*100, "%")
[1] "Rate of Test Error:  54.9450549450549 %"
> 
```

***********************************END***********************************
****

ALL CODE ARE ON MY GITHUB:
https://github.com/arthurmjt/CS465_Introduction-to-Statistical-Learning.git