1. **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 1,089 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?

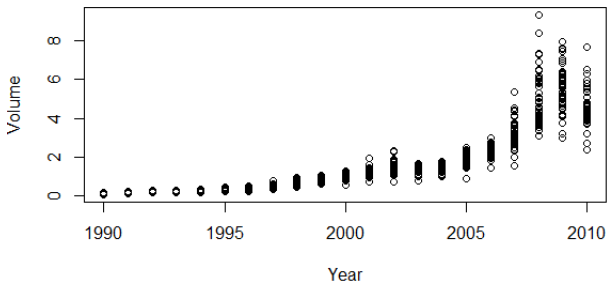
   The below figure gives us the summary of the numerical variables of the Weekly dataset.

```
> 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  
```

   Let us analyze the correlation matrix for the qualitative variables in Weekly dataset

```
> cor(weekly[,-9])
               Year         Lag1        Lag2        Lag3         Lag4         Lag5
Year     1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923 -0.030519101
Lag1    -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876 -0.008183096
Lag2    -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535 -0.072499482
Lag3    -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865  0.060657175
Lag4    -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000 -0.075675027
Lag5    -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027  1.000000000
Volume   0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617 -0.058517414
Today   -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873  0.011012698
              Volume        Today
Year     0.84194162 -0.032459894
Lag1    -0.06495131 -0.075031842
Lag2    -0.08551314  0.059166717
Lag3    -0.06928771 -0.071243639
Lag4    -0.06107462 -0.007825873
Lag5    -0.05851741  0.011012698
Volume   1.00000000 -0.033077783
Today   -0.03307778  1.000000000
```

   There is a high positive correlation between year and volume which indicates increase in volume of shares traded as the year's progress. None of the Lag variables are correlated with each other.



   A closer look at the relationship between Volume and Year. We can observe that the average number of shares traded per day has increased as the year's progressed.

(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?

The below is the summary of the logistic regression performed on the Weekly dataset with all the Lag variables and Volume with Direction as the response.

```
> 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
```

From the summary of the Logistic Regression only Lag2 (Percentage return for 2 weeks previous) variable is statistically significant. The coefficient of Lag2 is positive, which means that when the percentage return for 2 previous week's increases, it is more likely that Direction goes up or the market has a positive return.

(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

The confusion matrix is created with the below code:

```
> glm.probs<-predict(glm.fit,type="response")
> glm.preds<-rep("Down",length(glm.probs))
> glm.preds[glm.probs>0.5]="Up"
> table(glm.preds ,Direction)
         Direction
glm.preds Down  Up
     Down   54  48
     Up    430 557
```

The percentage of correct predictions is computed as (54+557)/1089=0.5611~56.11%. The training error rate is 43.89%. The model performs well when the market goes up 92.06% (557/(557+48)*100) of the time. However when the market actually goes down the prediction rate falls to 11.15% (54/(430+54)*100).

(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).

Below code is a step-by-step process to get the confusion matrix:

```
> weekly.pre09<-weekly[Year<2009,] #dataset before 2009
> weekly.post09<-weekly[Year>=2009,] #dataset after 2009
> glm.fitLag2<-glm(Direction~Lag2,data=weekly.pre09,family=binomial)
> glm.probsLag2<-predict(glm.fitLag2,weekly.post09,type="response")
> glm.predsLag2<-rep("Down",length(glm.probsLag2))
> glm.predsLag2[glm.probsLag2>0.5]="Up"
> Direction.0910<-subset(weekly.post09,select=Direction,drop=TRUE)
> table(glm.predsLag2, Direction.0910)
             Direction.0910
glm.predsLag2 Down Up
         Down    9  5
         Up     34 56
```

The fraction of correct predictions is 62.5% ((9+56/104)*100)

(e) Repeat (d) using LDA

Steps below using LDA to create the confusion matrix:

```
> library(MASS)
> lda.fitLag2<-lda(Direction~Lag2,data=weekly.pre09)
> lda.predsLag2<-predict(lda.fitLag2,weekly.post09)
> lda.class<-lda.predsLag2$class
> table(lda.class, Direction.0910)
         Direction.0910
lda.class Down Up
     Down    9  5
     Up     34 56
```

The fraction of correct predictions is 62.5% ((9+56/104)*100)

(f) Repeat (d) using QDA

```
> qda.fitLag2<-qda(Direction~Lag2,data=weekly.pre09)
> qda.predsLag2<-predict(qda.fitLag2,weekly.post09)
> qda.class<-qda.predsLag2$class
> table(qda.class, Direction.0910)
         Direction.0910
qda.class Down Up
     Down    0  0
     Up     43 61
```

The fraction of correct predictions is 58.65% ((61/104)*100). However, the model predicts correctly only the cases where the market goes up. In the case where the market goes down the prediction error is 100% i.e. it gets the prediction wrong every time the market goes down.

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

```
> train.X=as.matrix(weekly.pre09$Lag1)
> train.X=as.matrix(weekly.pre09$Lag2)
> test.X=as.matrix(weekly.post09$Lag2)
> set.seed(1)
> knn.pred=knn(train.X,test.X,subset(weekly.pre09,select=Direction,drop=TRUE),k=1)
> table(knn.pred,Direction.0910)
        Direction.0910
knn.pred Down Up
    Down   21 30
    Up     22 31
```

The fraction of correct predictions is 50%((52/104)*100).

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

By comparing the test error rates, we see that logistic regression and LDA have the same error rates, which are also the minimum error rates, followed by QDA and KNN. Hence, logistic regression and LDA provide the best results on this data

(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 classifiers.

Logistic Regression:

```
> glm.fit2<-glm(Direction~Lag2+I(Lag2^2) ,data=weekly.pre09 ,family =binomial )
> glm.probs2<-predict(glm.fit2,weekly.post09,type="response")
> glm.preds2<-rep("Down",length(glm.probs2))
> glm.preds2[glm.probs2>0.5]="Up"
> table(glm.preds2, Direction.0910)
          Direction.0910
glm.preds2 Down Up
      Down    8  4
      Up     35 57
> mean(glm.preds2==Direction.0910)
[1] 0.625
```

In logistic regression, the model showed best results when we considered Lag2 and Lag2^2. This model gave a correct prediction of 62.5 % similar to the initial model in question 1(d). The closest model to this is Direction~Lag1:Lag2 with a correct prediction rate of 58.65%

LDA:

```
> lda.fit2<-lda(Direction~Lag2+I(Lag2^2),data=weekly.pre09)
> lda.preds2<-predict(lda.fit2,weekly.post09)
> lda.class2<-lda.preds2$class
> table(lda.class2, Direction.0910)
           Direction.0910
lda.class2 Down Up
      Down    7  4
      Up     36 57
> mean(lda.class2==Direction.0910)
[1] 0.6153846
```

In LDA, the model showed best results when we considered Lag2 and Lag2^2. This model gave a correct prediction of 61.54 % similar to the initial model in question 1(e).

QDA:

```
> qda.fit2<-qda(Direction~Lag2+I(Lag2^2),data=weekly.pre09)
> qda.preds2<-predict(qda.fit2,weekly.post09)
> qda.class2<-qda.preds2$class
> table(qda.class2, Direction.0910)
           Direction.0910
qda.class2 Down Up
      Down    7  3
      Up     36 58
> mean(qda.class2==Direction.0910)
[1] 0.625
```

In QDA, the model showed best results when we considered Lag2 and Lag2^2. This model gave a correct prediction of 62.5 % similar to the initial model in question 1(f).

KNN:

```
> set.seed(10)
> train.X=as.matrix(weekly.pre09$Lag2,weekly.pre09$Lag2^2)
> test.X=as.matrix(weekly.post09$Lag2,weekly.pre09$Lag2^2)
> knn.pred=knn(train.X,test.X,subset(weekly.pre09,select=Direction,drop=TRUE),k=20)
> table(knn.pred,Direction.0910)
        Direction.0910
knn.pred Down Up
    Down   20 19
    Up     23 42
> mean(knn.pred==Direction.0910)
[1] 0.5961538
```

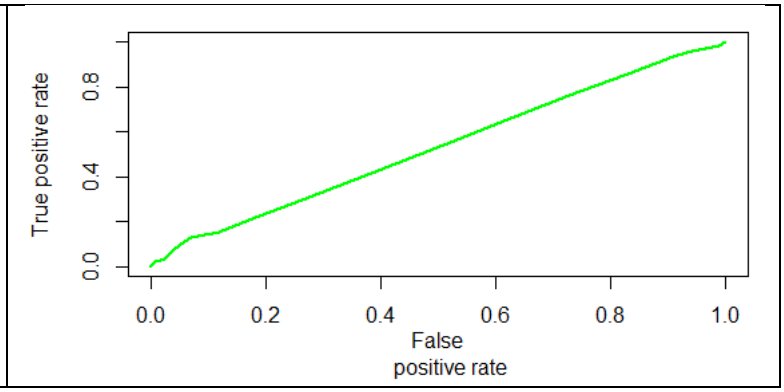
In KNN, the model showed best results when we considered Lag2 and Lag2^2 and K = 10. This model gave a correct prediction of 59.61 % similar to the initial model in question 1(g).

Of the four methods Logistic regression gave the best results with Lag2 and Lag2^2 as predictors with a least test error of 37.5%.

2. Perform ROC analysis and present the results for logistic regression and LDA used for the best model chosen in Question 1(i).

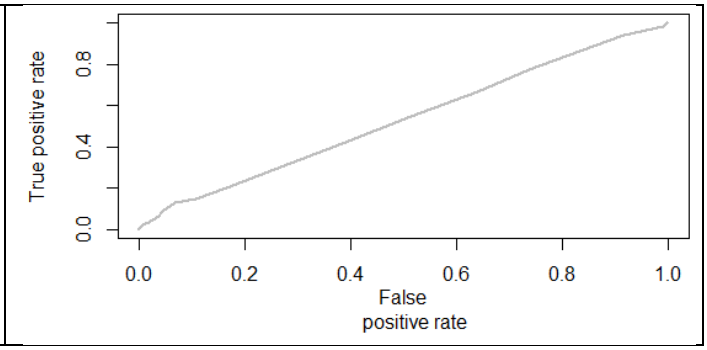
Logistic Regression:

```
> s=0.5
> roc.curve(s,print=TRUE)
          Predicted
Observed    0   1
    Down   28 456
    Up     24 581
       FPR       TPR
0.9421488 0.9603306
```



LDA:

```
> threshold=0.5
> roc.curve1(threshold,print=TRUE)
          Predicted
Observed    0   1
    Down   25 459
    Up     24 581
       FPR       TPR
0.9483471 0.9603306
```



The sensitivity is for both LR and LDA 0.96, which is the percentage of times the model predicts when the market moved "Up". However our model has a very high FPR of 0.942 and 0.948 for LR and LDA respectively. This means it wrongly predicts market going "Down" as "Up"

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) 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 that 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.

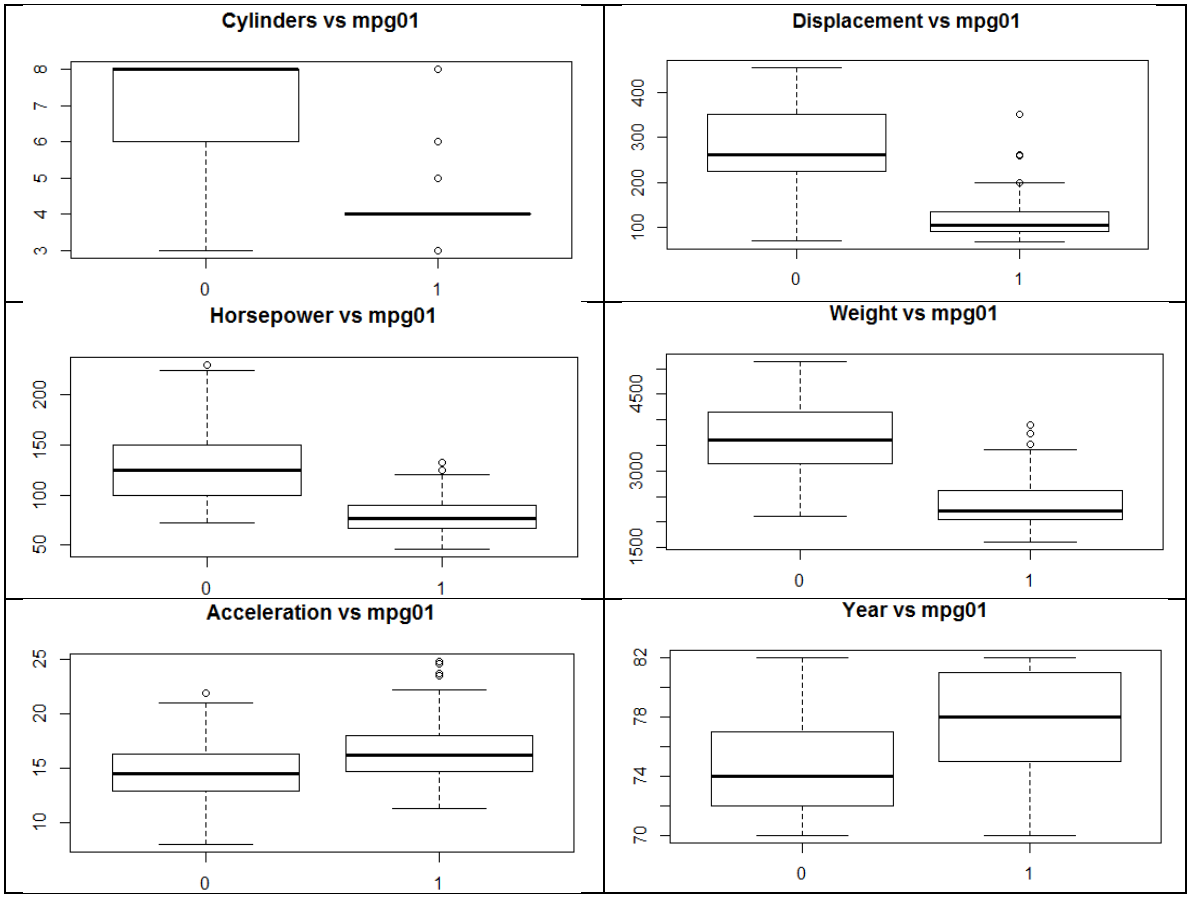
```
> mpg01 <- rep(0, length(mpg))
> mpg01[mpg > median(mpg)] <- 1
> Auto <- data.frame(Auto, mpg01)
> names(Auto)
 [1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
 [6] "acceleration" "year"         "origin"       "name"         "mpg01"
```

(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 predicting mpg01? Scatterplots and Boxplots may be useful tools to answer this question. Describe your findings.

Finding the correlation matrix between the variables

```
> cor(Auto[,-9])
                     mpg  cylinders displacement horsepower     weight acceleration
mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442    0.4233285
cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273   -0.5046834
displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944   -0.5438005
horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377   -0.6891955
weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000   -0.4168392
acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392    1.0000000
year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199    0.2903161
origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054    0.2127458
mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566    0.3468215
                    year     origin      mpg01
mpg           0.5805410  0.5652088  0.8369392
cylinders    -0.3456474 -0.5689316 -0.7591939
displacement -0.3698552 -0.6145351 -0.7534766
horsepower   -0.4163615 -0.4551715 -0.6670526
weight       -0.3091199 -0.5850054 -0.7577566
acceleration  0.2903161  0.2127458  0.3468215
year          1.0000000  0.1815277  0.4299042
origin        0.1815277  1.0000000  0.5136984
mpg01         0.4299042  0.5136984  1.0000000
```

From this correlation matrix we can see a high correlation between mpg01 and cylinders, displacement, horsepower, weight, year, acceleration



From the box plots we can derive the following conclusions:

- As the number of cylinders increases mpg decreases and falls below median.
- As the engine displacement increases mpg decreases and falls below median.
- As the horsepower increases mpg decreases and falls below median.
- As the weight increases mpg decreases and falls below median.
- Higher the acceleration higher the mpg and likely to be above median.
- As the manufacturing year increases mpg increases and likely to be above median.

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

Splitting the training and test data in the ratio of 75:25

```
> train_sample_size <- floor(0.75 * nrow(Auto))
> set.seed(123)
> train_index <- sample(seq_len(nrow(Auto)), size = train_sample_size)
> train_data<-Auto[train_index,]
> test_data<-Auto[-train_index]
```

(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?

```
> lda.Auto<-lda(mpg01~cylinders+displacement+horsepower+weight+acceleration+year,data=train
_data)
> lda.Auto
Call:
lda(mpg01 ~ cylinders + displacement + horsepower + weight +
    acceleration + year, data = train_data)

Prior probabilities of groups:
        0         1
0.4863946 0.5136054

Group means:
  cylinders displacement horsepower   weight acceleration    year
0  6.790210     276.6434  131.88811 3659.091     14.54685 74.3007
1  4.205298     116.4536   79.33113 2336.086     16.42848 77.6755

Coefficients of linear discriminants:
                    LD1
cylinders    -0.424524318
displacement  0.001026357
horsepower    0.011549043
weight       -0.001468573
acceleration  0.013879772
year          0.129595079
> lda.predAuto<-predict(lda.Auto,test_data)
> lda.classAuto<-lda.predAuto$class
> table(lda.classAuto, mpg01.test)
             mpg01.test
lda.classAuto  0  1
            0 44  1
            1  9 44
> mean(lda.classAuto != mpg01.test)
[1] 0.1020408
```

The test error of LDA is 10.20408 %

(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?

```
> qda.Auto<-qda(mpg01~cylinders+displacement+horsepower+weight+acceleration+year,data=train_d
ata)
> qda.Auto
Call:
qda(mpg01 ~ cylinders + displacement + horsepower + weight +
    acceleration + year, data = train_data)

Prior probabilities of groups:
        0         1
0.4863946 0.5136054

Group means:
  cylinders displacement horsepower   weight acceleration    year
0  6.790210     276.6434  131.88811 3659.091     14.54685 74.3007
1  4.205298     116.4536   79.33113 2336.086     16.42848 77.6755
> qda.predAuto<-predict(qda.Auto,test_data)
> qda.classAuto<-qda.predAuto$class
> table(qda.classAuto, mpg01.test)
             mpg01.test
qda.classAuto  0  1
            0 46  2
            1  7 43
> mean(qda.classAuto != mpg01.test)
[1] 0.09183673
```

The test error of QDA is 9.183673%

(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?

```
> glm.Auto<-glm(mpg01~cylinders+displacement+horsepower+weight+acceleration+year,data=train
_data ,family =binomial )
> summary(glm.Auto)

Call:
glm(formula = mpg01 ~ cylinders + displacement + horsepower +
    weight + acceleration + year, family = binomial, data = train_data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.29689  -0.09721   0.02579   0.21439   2.96400

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -20.884869   7.224877  -2.891  0.00384 **
cylinders     0.030457   0.451005   0.068  0.94616
displacement  0.002089   0.011615   0.180  0.85728
horsepower   -0.020707   0.025792  -0.803  0.42207
weight       -0.005445   0.001397  -3.897 9.75e-05 ***
acceleration  0.061972   0.165287   0.375  0.70771
year          0.484498   0.098262   4.931 8.19e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 407.35  on 293  degrees of freedom
Residual deviance: 116.21  on 287  degrees of freedom
AIC: 130.21

Number of Fisher Scoring iterations: 8
> glm.probsAuto<-predict(glm.Auto,test_data,type="response")
> glm.predsAuto<-rep(0,length(glm.probsAuto))
> glm.predsAuto[glm.probsAuto>0.5]=1
> table(glm.predsAuto, mpg01.test)
             mpg01.test
glm.predsAuto  0  1
            0 46  4
            1  7 41
> mean(glm.predsAuto!=mpg01.test)
[1] 0.1122449
```

The test error of Logistic Regression is 11.22449%

(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?

K=10:

```
> set.seed(1)
> mpg01.train<-mpg01[train_index]
> train.Auto=as.matrix(train_data$cylinders,train_data$displacement,train_data$horsepower,tra
in_data$weight,train_data$acceleration,train_data$year)
> test.Auto=as.matrix(test_data$cylinders,test_data$displacement,test_data$horsepower,test_da
ta$weight,test_data$acceleration,test_data$year)
> knn.predAuto=knn(train.Auto,test.Auto,mpg01.train,k=10)
> table(knn.predAuto,mpg01.test)
           mpg01.test
knn.predAuto  0  1
           0 47  2
           1  6 43
> mean(knn.predAuto!=mpg01.test)
[1] 0.08163265
```

Test error: 8.163265%

K=100:

```
> set.seed(1)
> mpg01.train<-mpg01[train_index]
> train.Auto=as.matrix(train_data$cylinders,train_data$displacement,train_data$horsepower,t
rain_data$weight,train_data$acceleration,train_data$year)
> test.Auto=as.matrix(test_data$cylinders,test_data$displacement,test_data$horsepower,test_
data$weight,test_data$acceleration,test_data$year)
> knn.predAuto=knn(train.Auto,test.Auto,mpg01.train,k=100)
> table(knn.predAuto,mpg01.test)
           mpg01.test
knn.predAuto  0  1
           0 25  0
           1 28 45
> mean(knn.predAuto!=mpg01.test)
[1] 0.2857143
```

Test error: 28.57143%

K=200:

```
> set.seed(1)
> mpg01.train<-mpg01[train_index]
> train.Auto=as.matrix(train_data$cylinders,train_data$displacement,train_data$horsepower,t
rain_data$weight,train_data$acceleration,train_data$year)
> test.Auto=as.matrix(test_data$cylinders,test_data$displacement,test_data$horsepower,test_
data$weight,test_data$acceleration,test_data$year)
> knn.predAuto=knn(train.Auto,test.Auto,mpg01.train,k=200)
> table(knn.predAuto,mpg01.test)
           mpg01.test
knn.predAuto  0  1
           0  0  0
           1 53 45
> mean(knn.predAuto!=mpg01.test)
[1] 0.5408163
```

Test error: 54.08163%

K=10 seems to be the best performance. 10 nearest neighbors.