**Linga Sai Yuvesh Venketa, Kotiala**

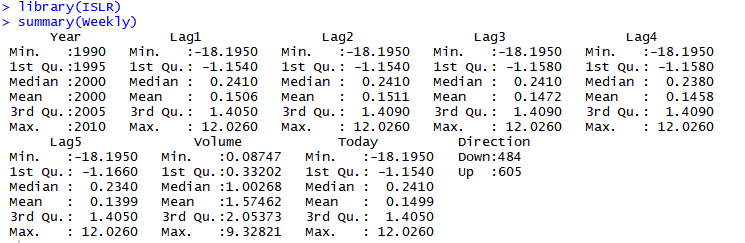
**16242113**

**ISL LAB-3**

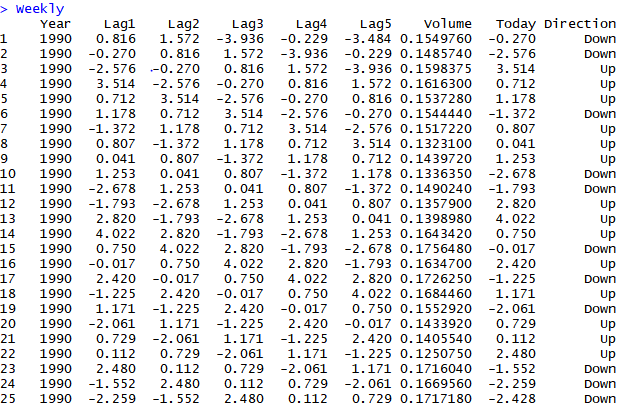
**2. This question should be answered using the Weekly data set, which is part of the ISLR package. This data is similar in nature to the Smarket data from this chapters lab, except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.**

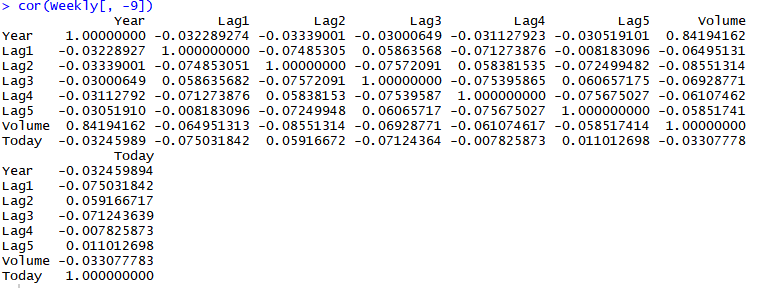
**(a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?**

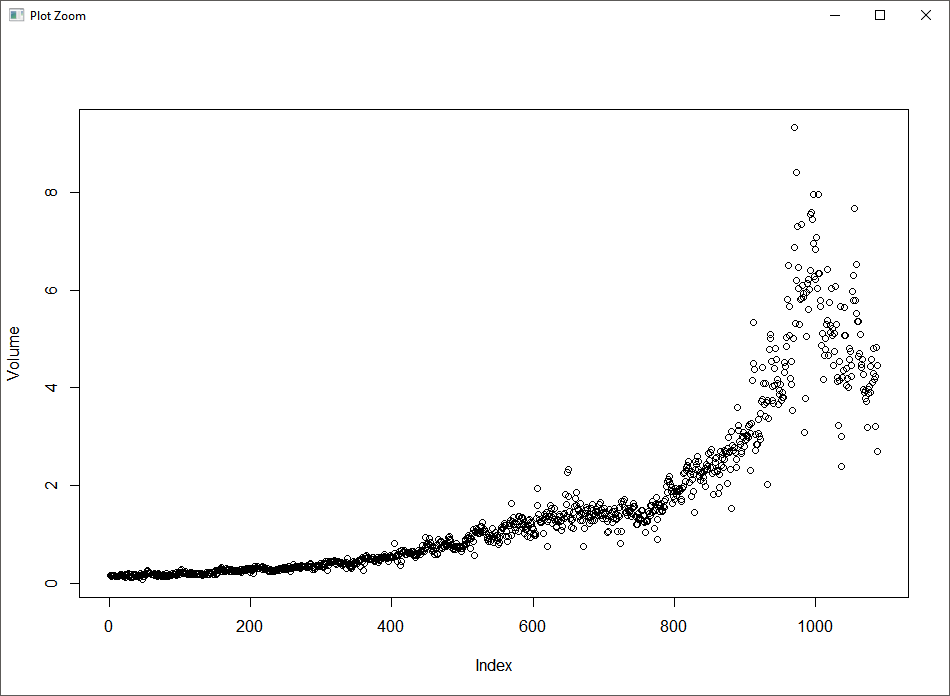
Ans.



Weekly percentage returns for the S&P 500 stock index between 1990 and 2010. A data frame in ISLR.



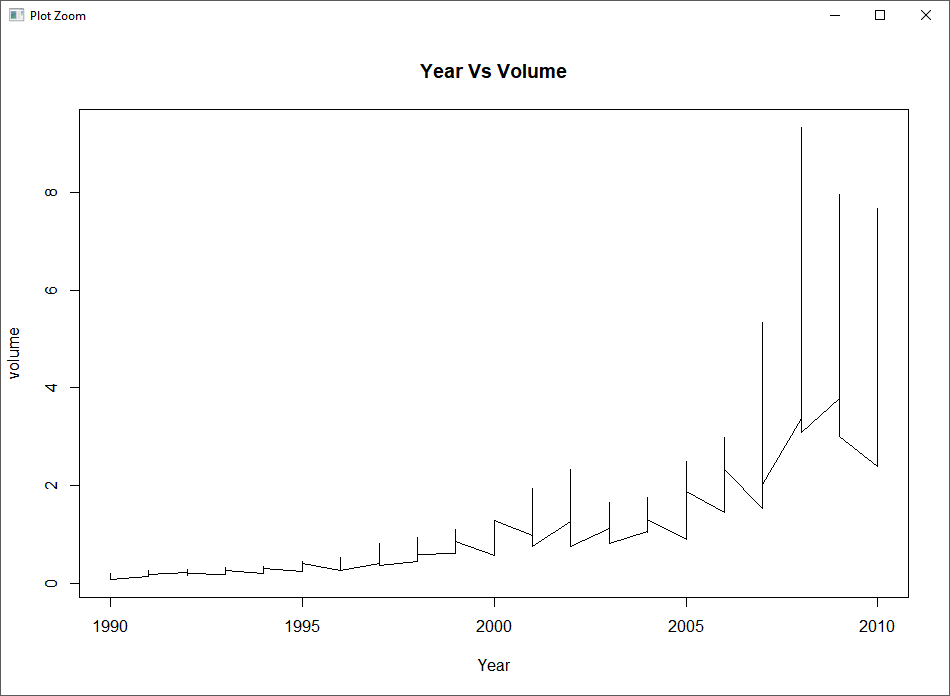




The correlations between the “lag” variables and today’s returns are close to zero. The only substantial correlation is between “Year” and “Volume”. When we plot “Volume”, we see that it is increasing over time.

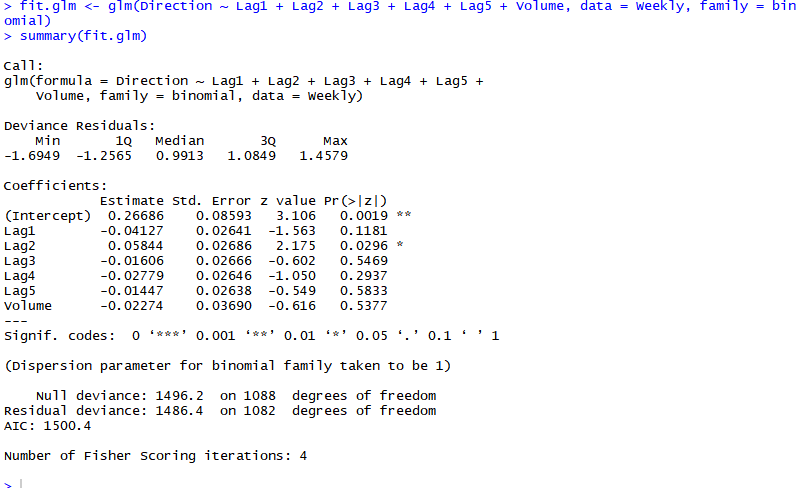
Between number of up’s and down’s

Year v/s Direction



**(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?**

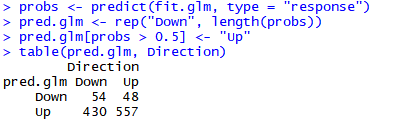
Ans.



It would seem that “Lag2” is the only predictor statistically significant as its p-value is less than 0.05.

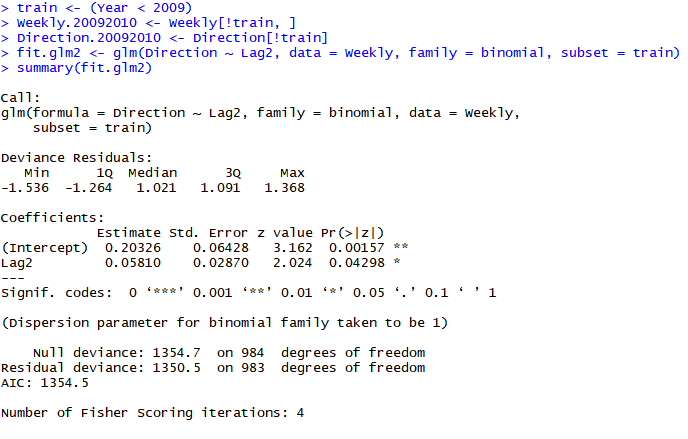
**(c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.**

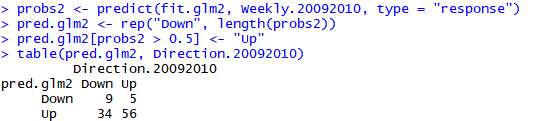
Ans.



We may conclude that the percentage of correct predictions on the training data is (54+557)/1089 which 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)). 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 to 2010).**



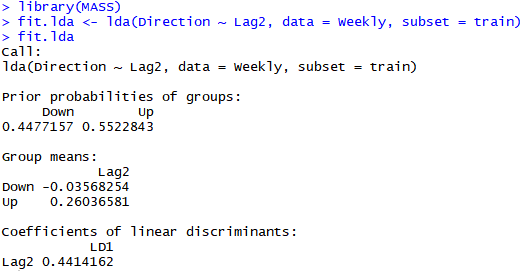


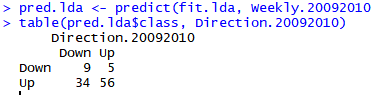
In this case, we may conclude that the percentage of correct predictions on the test data is (9+56)/104 which is equal to 62.5%. In other words, 37.5% is the test error rate. We could also say that for weeks when the market goes up, the model is right 91.8032787% of the time (56/(56+5)). For weeks when the market goes down, the model is right only 20.9302326% of the time (9/(9+34)).

**(e) Repeat (d) using LDA.**

Ans.

Here we use “MASS” library to use the LDA and QDA

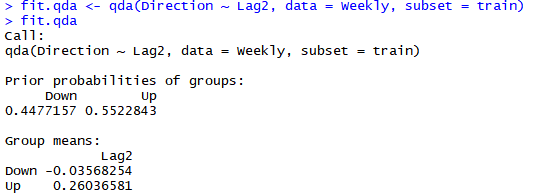


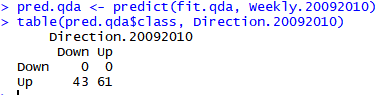


In this case, we may conclude that the percentage of correct predictions on the test data is 62.5%. In other words, 37.5% is the test error rate. We could also say that for weeks when the market goes up, the model is right 91.8032787% of the time. For weeks when the market goes down, the model is right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

**(f) Repeat (d) using QDA.**

Ans.

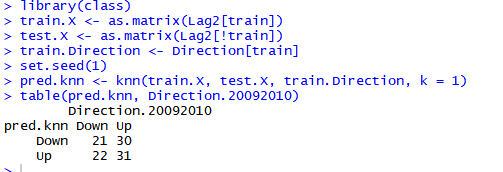




In this case, we may conclude that the percentage of correct predictions on the test data is 58.6538462%. In other words, 41.3461538% is the test error rate. We could also say that for weeks when the market goes up, the model is right 100% of the time. For weeks when the market goes down, the model is 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 = 1.**

Ans.



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.

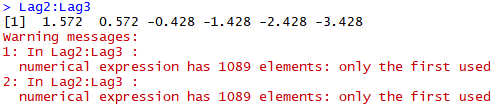
**(h) Which of these methods appears to provide the best results on this data?**

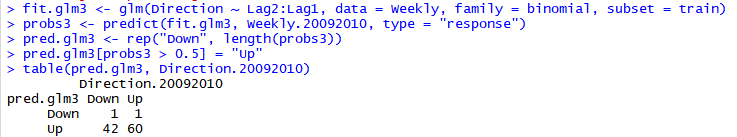
Ans.

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 K in the KNN classifier.**

Ans.







**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 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.**

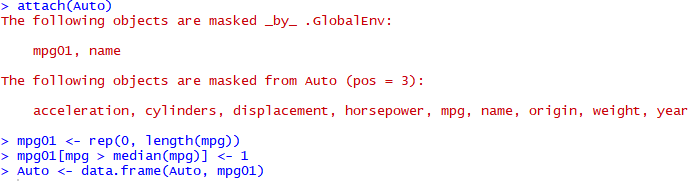
Ans.

>attach(Auto)

>mpg01 <- rep(0, length(mpg))

>mpg01[mpg > median(mpg)] <- 1

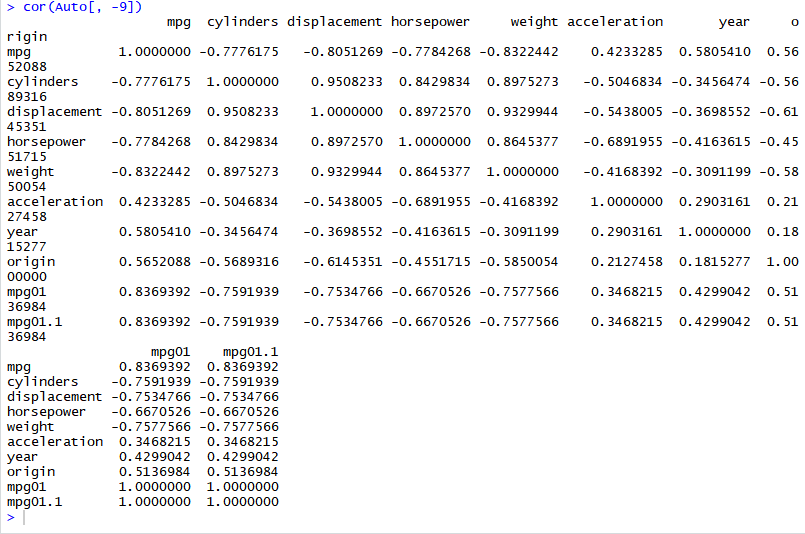
>Auto <- data.frame(Auto, mpg01)



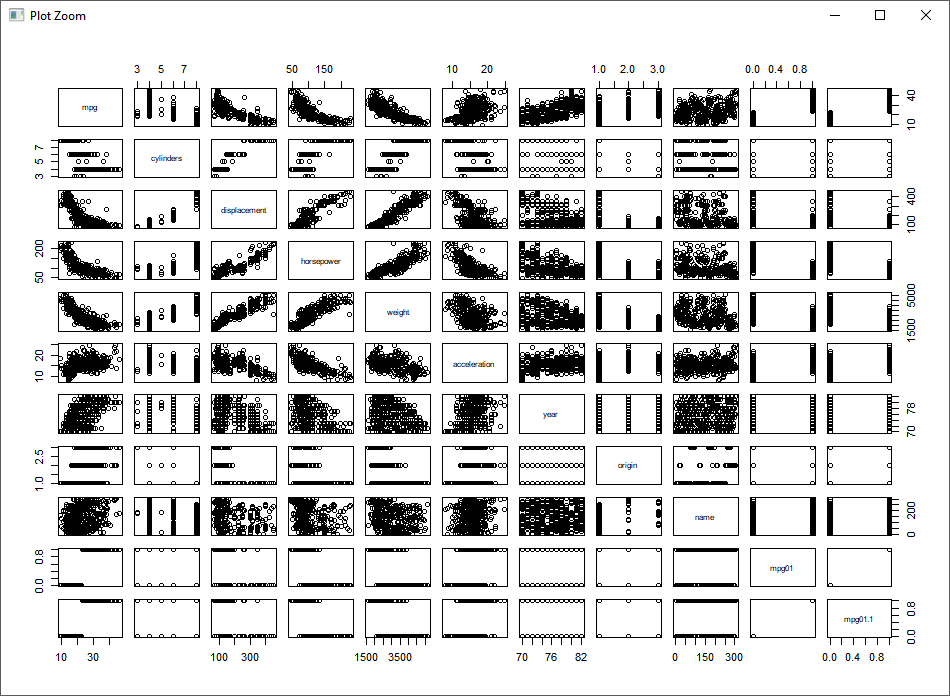
**(b) Explore the data graphically in order to investigate the association between “mpg01” and the other features. Which of the other features seem most likely to be useful in predicting “mpg01”? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.**

Ans.

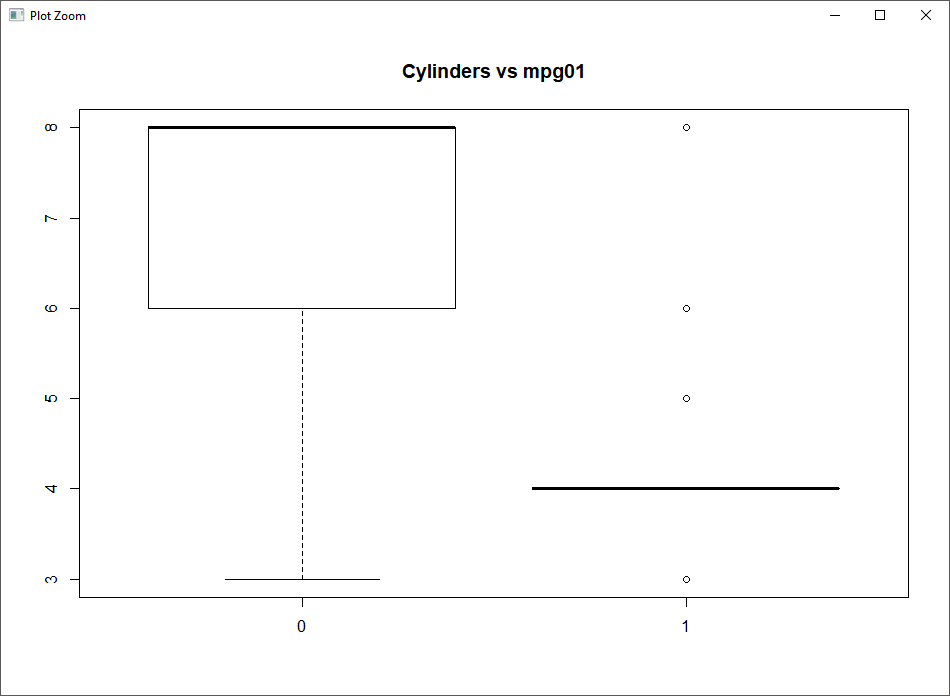
>cor(Auto[, -9])



>pairs(Auto)

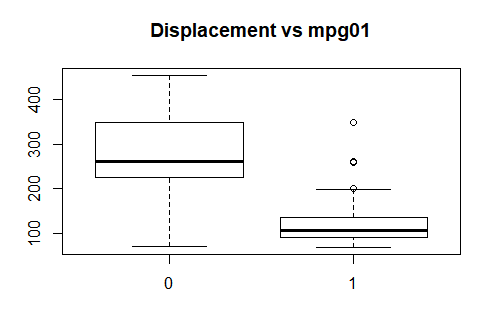


>boxplot(cylinders ~ mpg01, data = Auto, main = "Cylinders vs mpg01")



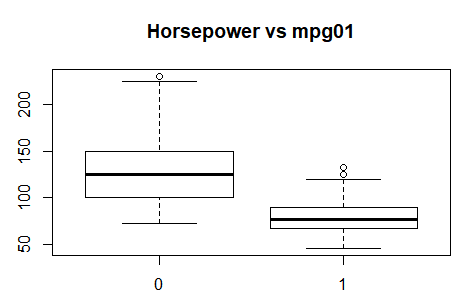
>boxplot(displacement ~ mpg01, data = Auto, main = "Displacement vs mpg01")

## here we get a box plot for the values of 0 and 1 when plotted between Displacement and mpg01



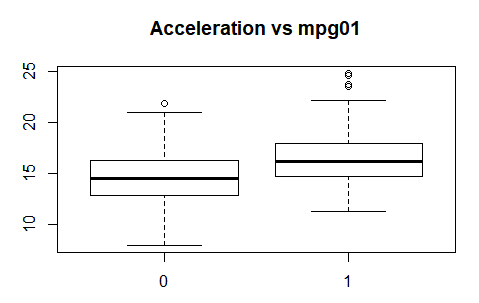
>boxplot(horsepower ~ mpg01, data = Auto, main = "Horsepower vs mpg01")

## here we get a box plot for the values of 0 and 1 when plotted between Horsepower and mpg01



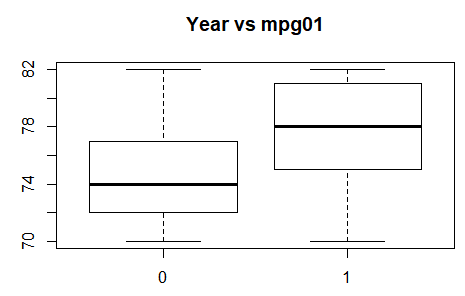
>boxplot(acceleration ~ mpg01, data = Auto, main = "Acceleration vs mpg01")

## here we get a box plot for the values of 0 and 1 when plotted between Acceleration and mpg01



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

## here we get a box plot for the values of 0 and 1 when plotted between Year and 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.**

Ans.

>train <- (year %% 2 == 0)

>Auto.train <- Auto[train, ]

>Auto.test <- Auto[!train, ]

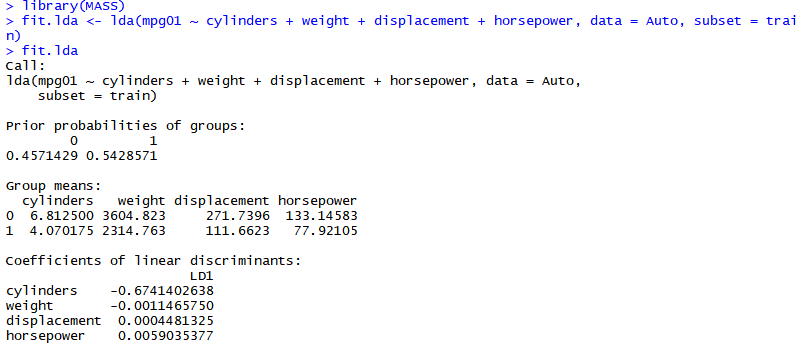
>mpg01.test <- mpg01[!train]

**(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?**

Ans.

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

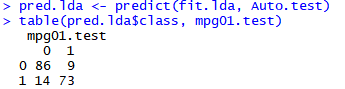
>fit.lda



For LDA to work we should import the library named “mass”

>pred.lda <- predict(fit.lda, Auto.test)

>table(pred.lda$class, mpg01.test)



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



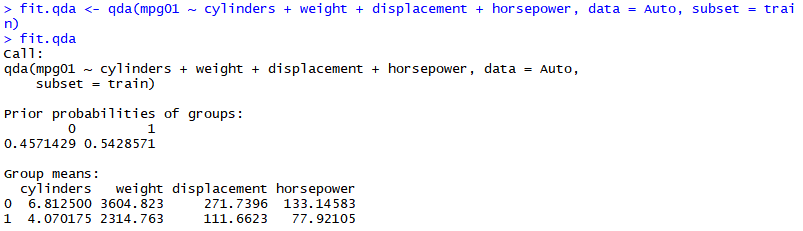
We may conclude that we have a test error rate of 12.63736%.

**(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?**

Ans.

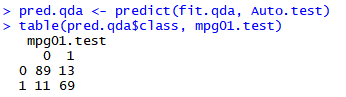
>fit.qda <- qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = train)

>fit.qda



>pred.qda <- predict(fit.qda, Auto.test)

>table(pred.qda$class, mpg01.test)



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



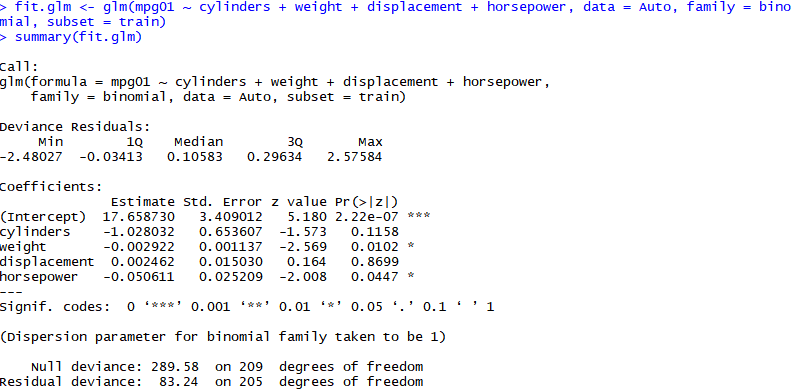
We may conclude that we have a test error rate of 13.1868132%.

**(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?**

Ans.

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

>summary(fit.glm)

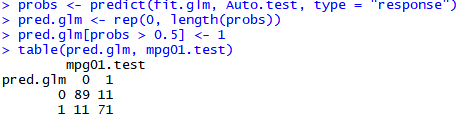


>probs <- predict(fit.glm, Auto.test, type = "response")

>pred.glm <- rep(0, length(probs))

>pred.glm[probs > 0.5] <- 1

>table(pred.glm, mpg01.test)



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



We may conclude that we have a test error rate of 12.08791%.

**(g) Perform KNN on the training data, with several values of K, 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 K seems to perform the best on this data set?**

Ans.

>train.X <- cbind(cylinders, weight, displacement, horsepower)[train, ]

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

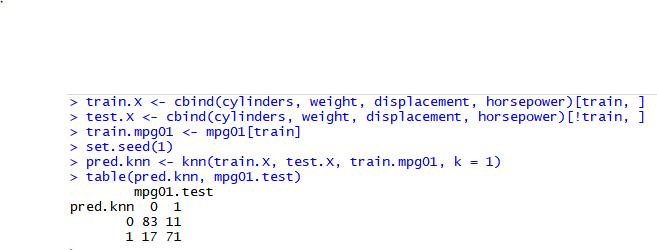
>train.mpg01 <- mpg01[train]

>set.seed(1)

**##k=1**

>pred.knn <- knn(train.X, test.X, train.mpg01, k = 1)

>table(pred.knn, mpg01.test)



>mean(pred.knn != mpg01.test)

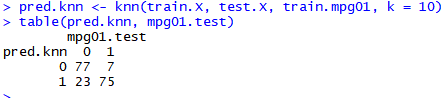


We may conclude that we have a test error rate of 15.3846154% for K=1.

**## K=10**

>pred.knn <- knn(train.X, test.X, train.mpg01, k = 10)

>table(pred.knn, mpg01.test)



>mean(pred.knn != mpg01.test)

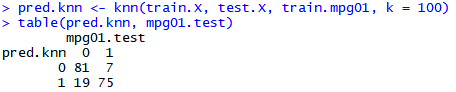


We may conclude that we have a test error rate of 16.4835165% for K=10.

**##k=100**

>pred.knn <- knn(train.X, test.X, train.mpg01, k = 100)

>table(pred.knn, mpg01.test)



>mean(pred.knn != mpg01.test)



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.

GitHub Link : <https://github.com/Yuvesh95/R-programming>