**STATS 202 | HW:3 | Sagar Ganapaneni | SUID# 06167633**

**Problem: 2**

**LDA Vs QDA**

1. If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?
   1. For **Training set**: we expect QDA to perform better as it fits closer to data point in training data set
   2. For **Test set:** We expect LDA to perform better on test set as QDA over fits the training data leading to high variance when applied to test set
2. If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

This case we expect QDA to perform better for both training and test data as bias component is too high with LDA model.

1. In general, as the sample size nn increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

In general, with higher sample size QDA performs better compared to LDA as variance component is not a big concern when the sample size is large.

1. True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

False. When the sample size is smaller QDA will over fit the training data, leading to higher error with Test data.

**Problem: 4**

For KNN with K=1 🡺 we have zero percent training error🡺 test error itself 36%

Whereas test error for logistic regression model is 30%, which is better than KNN.

So we choose Logistic regression over KNN with k=1 model

**Problem: 5**

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

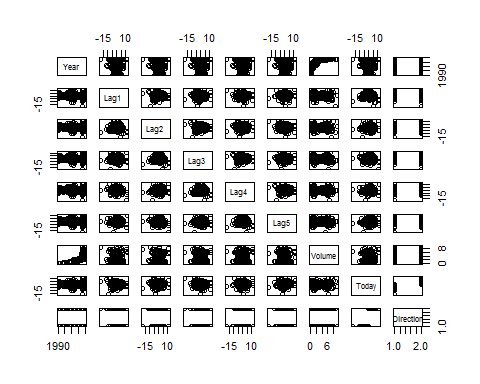
library(ISLR)

## Warning: package 'ISLR' was built under R version 3.2.5

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

pairs(Weekly)

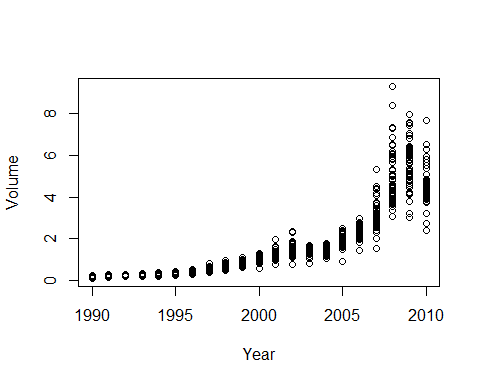


cor(Weekly[,-9])

## 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

Thera is high correlation between Year and Volume, lets plot bivariate plot for these variables

attach(Weekly)  
plot(Year,Volume)



Median volume is increasing each year

1. Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors.

glm\_model <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly, family = binomial)  
summary(glm\_model)

##   
## 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

Only **Lag2** is statistically significant variable among all other X variables

1. 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.

ods <- predict(glm\_model, type = "response")  
glm\_pred <- rep("Down", length(ods))  
glm\_pred[ods > 0.5] <- "Up"  
table(glm\_pred, Direction)

## Direction  
## glm\_pred Down Up  
## Down 54 48  
## Up 430 557

* By looking at the above confusion matrix, we can calculate training error rate: (430+48)/1089 = **43.89348 %**
* Also training error is really high when Direction is Down: 430/ (54+430): **88.84298 %**
* When the Direction is Up, the training error is better: 48/ (48+557): **7.93388%**

1. 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).

Weekly\_Pre2009 <- Weekly[Year<2009, ]  
Weekly\_Post2009 <- Weekly[Year>2008, ]  
glm\_model <- glm(Direction ~ Lag2, data = Weekly\_Pre2009, family = binomial)  
summary(glm\_model)

##   
## Call:  
## glm(formula = Direction ~ Lag2, family = binomial, data = Weekly\_Pre2009)  
##   
## 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

ods <- predict(glm\_model, Weekly\_Post2009, type = "response")  
glm\_pred <- rep("Down", length(ods))  
glm\_pred[ods > 0.5] <- "Up"  
table(glm\_pred, Weekly\_Post2009$Direction)

##   
## glm\_pred Down Up  
## Down 9 5  
## Up 34 56

* By looking at the above confusion matrix, we can calculate test error rate: 39/104= **37.5%**
* Also test error is really high when Direction is Down: 34/43: **79.0697 %**
* When the Direction is Up: 5/61: **8.19672%**

1. Repeat (d) using LDA.

library(MASS)  
lda\_model <- lda(Direction ~ Lag2, data = Weekly\_Pre2009)  
lda\_model

## Call:  
## lda(Direction ~ Lag2, data = Weekly\_Pre2009)  
##   
## 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\_model, Weekly\_Post2009)  
table(lda\_pred$class, Weekly\_Post2009$Direction)

##   
## Down Up  
## Down 9 5  
## Up 34 56

* By looking at the above confusion matrix, we can calculate test error rate: 39/104= **37.5%**
* Also test error is really high when Direction is Down: 34/43: **79.0697 %**
* When the Direction is Up: 5/61: **8.19672%**
* This results are similar to what wo got from Logistic Regression Model

1. Repeat (d) using QDA.

qda\_model <- qda(Direction ~ Lag2, data = Weekly\_Pre2009)  
qda\_model

## Call:  
## qda(Direction ~ Lag2, data = Weekly\_Pre2009)  
##   
## Prior probabilities of groups:  
## Down Up   
## 0.4477157 0.5522843   
##   
## Group means:  
## Lag2  
## Down -0.03568254  
## Up 0.26036581

qda\_pred <- predict(qda\_model, Weekly\_Post2009)  
table(qda\_pred$class, Weekly\_Post2009$Direction)

##   
## Down Up  
## Down 0 0  
## Up 43 61

* By looking at the above confusion matrix, we can calculate test error rate: 61/104= **58.6538%**
* Also test error is really high when Direction is Down being **100 %**
* Even with moderate overall test error, we don’t want to consider this model as it always predict when Direction is UP

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

set.seed(12)  
library(class)

## Warning: package 'class' was built under R version 3.2.5

## with K =5  
knn\_pred <- knn(as.matrix(Weekly\_Pre2009[,c("Lag2")]), as.matrix(Weekly\_Post2009[,c("Lag2")]), Weekly\_Pre2009$Direction, k = 1)  
table(knn\_pred, Weekly\_Post2009$Direction)

##   
## knn\_pred Down Up  
## Down 21 29  
## Up 22 32

* Test error rate: 22+29/(104) : **41.34615%**
* Also test error is really high when Direction is Down: **51.162790%**
* When the Direction is Up: 29/61: **47.54098%**

1. 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 KNN and QDA.

1. examine whether it is worth to include interactions via a forward selection scheme for LDA, which greedily minimizes the test error as it adds variables to the model one at a time.

library(MASS)  
**## step 1**  
lda\_model <- lda(Direction ~ Lag2, data = Weekly\_Pre2009)  
lda\_model

## Call:  
## lda(Direction ~ Lag2, data = Weekly\_Pre2009)  
##   
## 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\_model, Weekly\_Post2009)  
table(lda\_pred$class, Weekly\_Post2009$Direction)

##   
## Down Up  
## Down 9 5  
## Up 34 56

mean(lda\_pred$class !=Weekly\_Post2009$Direction)

## [1] **0.375**

library(MASS)  
**## step 2**  
lda\_model <- lda(Direction ~ Lag2:Lag3, data = Weekly\_Pre2009)  
lda\_model

## Call:  
## lda(Direction ~ Lag2:Lag3, data = Weekly\_Pre2009)  
##   
## Prior probabilities of groups:  
## Down Up   
## 0.4477157 0.5522843   
##   
## Group means:  
## Lag2:Lag3  
## Down -0.1937158  
## Up -0.6405132  
##   
## Coefficients of linear discriminants:  
## LD1  
## Lag2:Lag3 0.1012928

lda\_pred <- predict(lda\_model, Weekly\_Post2009)  
table(lda\_pred$class, Weekly\_Post2009$Direction)

##   
## Down Up  
## Down 0 0  
## Up 43 61

mean(lda\_pred$class !=Weekly\_Post2009$Direction)

## [1] **0.4134615**

library(MASS)  
**## step 3**  
lda\_model <- lda(Direction ~ Lag2:Lag4, data = Weekly\_Pre2009)  
lda\_model

## Call:  
## lda(Direction ~ Lag2:Lag4, data = Weekly\_Pre2009)  
##   
## Prior probabilities of groups:  
## Down Up   
## 0.4477157 0.5522843   
##   
## Group means:  
## Lag2:Lag4  
## Down 0.78824608  
## Up 0.04407141  
##   
## Coefficients of linear discriminants:  
## LD1  
## Lag2:Lag4 0.1287072

lda\_pred <- predict(lda\_model, Weekly\_Post2009)  
table(lda\_pred$class, Weekly\_Post2009$Direction)

##   
## Down Up  
## Down 1 4  
## Up 42 57

mean(lda\_pred$class !=Weekly\_Post2009$Direction)

## [1] **0.4423077**

library(MASS)  
## step 4  
lda\_model <- lda(Direction ~ Lag2:Lag5, data = Weekly\_Pre2009)  
lda\_model

## Call:  
## lda(Direction ~ Lag2:Lag5, data = Weekly\_Pre2009)  
##   
## Prior probabilities of groups:  
## Down Up   
## 0.4477157 0.5522843   
##   
## Group means:  
## Lag2:Lag5  
## Down -0.3132494  
## Up -0.3497535  
##   
## Coefficients of linear discriminants:  
## LD1  
## Lag2:Lag5 0.1105356

lda\_pred <- predict(lda\_model, Weekly\_Post2009)  
table(lda\_pred$class, Weekly\_Post2009$Direction)

##   
## Down Up  
## Down 0 0  
## Up 43 61

mean(lda\_pred$class !=Weekly\_Post2009$Direction)

## [1] **0.4134615**

With forward selection after Step 1, as we add more variable using forward selection the test errors are increasing, so there is no point adding variables other Lag2 in the model.

**Problem 6**

1. 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.

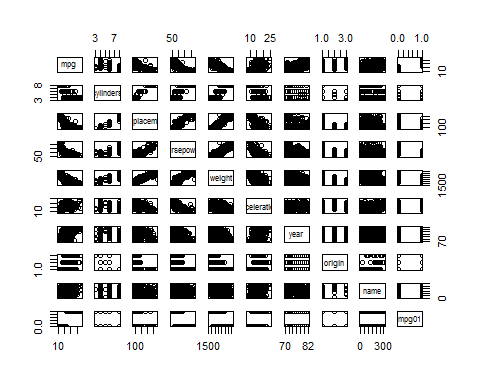
library(ISLR)

## Warning: package 'ISLR' was built under R version 3.2.5

attach(Auto)  
Auto$mpg01 <- 0  
Auto[mpg > median(mpg),]$mpg01 <- 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?

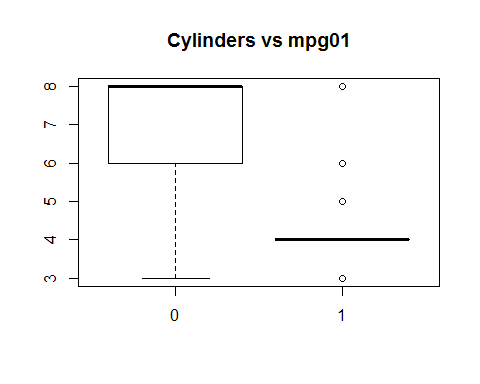
pairs(Auto)



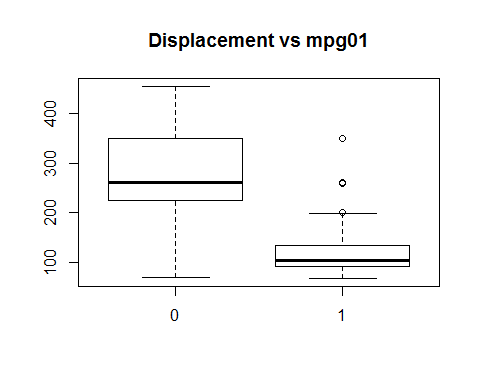
cor(Auto[,-9])

## 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

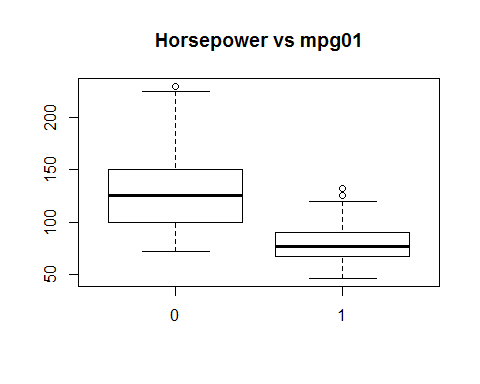
boxplot(cylinders ~ mpg01, data = Auto, main = "Cylinders vs mpg01")



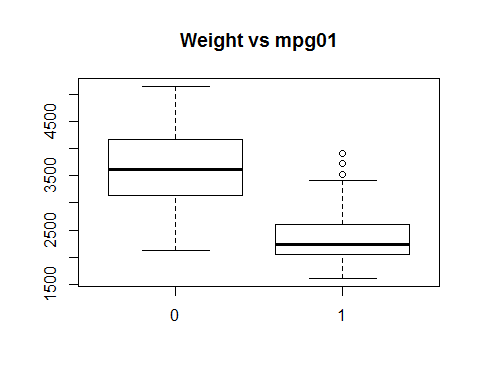
boxplot(displacement ~ mpg01, data = Auto, main = "Displacement vs mpg01")



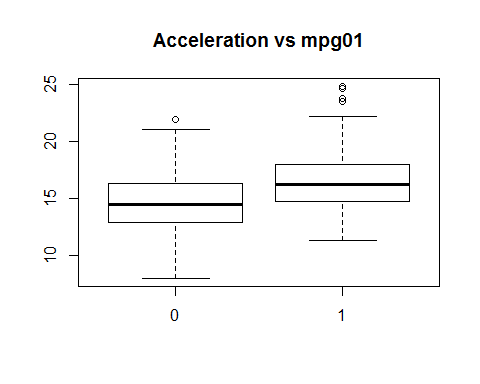
boxplot(horsepower ~ mpg01, data = Auto, main = "Horsepower vs mpg01")



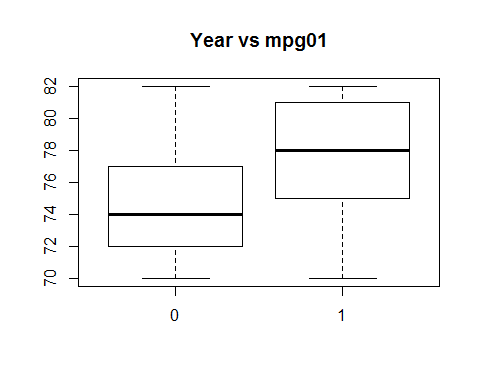
boxplot(weight ~ mpg01, data = Auto, main = "Weight vs mpg01")



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



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



By looking at correlation matrix, scatterplot and boxplots we can say there is some relationship between mpg01 and cylinders, weight, displacement and horsepower.

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

## add rnum coloumn  
Auto$rnum<-seq(1,nrow(Auto),1)  
## split data  
Auto\_train <- Auto[Auto$rnum %% 2 ==0, ]  
Auto\_test<- Auto[Auto$rnum %% 2 !=0, ]  
##drop runm coloumn  
Auto\_train<-Auto\_train[,!(names(Auto\_train) %in% c("rnum"))]  
Auto\_test<-Auto\_test[,!(names(Auto\_test) %in% c("rnum"))]  
Auto<-Auto[,!(names(Auto) %in% c("rnum"))]

1. 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?

library(MASS)  
lda\_model <- lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto\_train)  
lda\_model

## Call:  
## lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto\_train)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.4897959 0.5102041   
##   
## Group means:  
## cylinders weight displacement horsepower  
## 0 6.760417 3653.583 273.500 132.4271  
## 1 4.170000 2305.070 114.045 78.8700  
##   
## Coefficients of linear discriminants:  
## LD1  
## cylinders -0.4297440160  
## weight -0.0011996694  
## displacement 0.0003516146  
## horsepower 0.0021885992

lda\_pred <- predict(lda\_model, Auto\_test)  
table(lda\_pred$class, Auto\_test$mpg01)

##   
## 0 1  
## 0 83 6  
## 1 17 90

mean(lda\_pred$class !=Auto\_test$mpg01)

## [1] 0.1173469

Hence the error rate is **11.73%**

1. Perform QDA on the training data in order to predict "mpg01"

qda\_model <- qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto\_train)  
qda\_model

## Call:  
## qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto\_train)  
##   
## Prior probabilities of groups:  
## 0 1   
## 0.4897959 0.5102041   
##   
## Group means:  
## cylinders weight displacement horsepower  
## 0 6.760417 3653.583 273.500 132.4271  
## 1 4.170000 2305.070 114.045 78.8700

qda\_pred <- predict(qda\_model, Auto\_test)  
table(qda\_pred$class, Auto\_test$mpg01)

##   
## 0 1  
## 0 89 9  
## 1 11 87

mean(qda\_pred$class != Auto\_test$mpg01)

## [1] 0.1020408

Hence the error rate with QDA model is **10.20%** which is lower compared to LDA

1. Perform logistic regression on the training data in order to predict "mpg01"

glm\_model <- glm(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto\_train, family = binomial)  
summary(glm\_model)

##   
## Call:  
## glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower,   
## family = binomial, data = Auto\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.2416 -0.1088 0.1038 0.3176 2.9731   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 13.1401211 2.5218930 5.210 1.88e-07 \*\*\*  
## cylinders -0.0665897 0.4846865 -0.137 0.8907   
## weight -0.0025557 0.0009355 -2.732 0.0063 \*\*   
## displacement -0.0084554 0.0109024 -0.776 0.4380   
## horsepower -0.0431963 0.0197508 -2.187 0.0287 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 271.632 on 195 degrees of freedom  
## Residual deviance: 97.301 on 191 degrees of freedom  
## AIC: 107.3  
##   
## Number of Fisher Scoring iterations: 7

ods <- predict(glm\_model, Auto\_test, type = "response")  
glm\_pred <- rep(0, length(ods))  
glm\_pred[ods > 0.5] <- 1  
table(glm\_pred, Auto\_test$mpg01)

##   
## glm\_pred 0 1  
## 0 88 8  
## 1 12 88

mean(glm\_pred != Auto\_test$mpg01)

## [1] 0.1020408

Test Error rate here is:**10.20 %**

1. Perform KNN on the training data, with several values of K, in order to predict "mpg01"

set.seed(12)  
library(class)

## Warning: package 'class' was built under R version 3.2.5

## with K =5  
knn\_pred <- knn(Auto\_train[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_test[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_train$mpg01, k = 5)  
table(knn\_pred, Auto\_test$mpg01)

##   
## knn\_pred 0 1  
## 0 86 17  
## 1 14 79

mean(knn\_pred != Auto\_test$mpg01)

## [1] 0.1581633

## with K =10  
knn\_pred <- knn(Auto\_train[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_test[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_train$mpg01, k = 10)  
table(knn\_pred, Auto\_test$mpg01)

##   
## knn\_pred 0 1  
## 0 85 18  
## 1 15 78

mean(knn\_pred != Auto\_test$mpg01)

## [1] 0.1683673

## with K =100  
knn\_pred <- knn(Auto\_train[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_test[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_train$mpg01, k = 100)  
table(knn\_pred, Auto\_test$mpg01)

##   
## knn\_pred 0 1  
## 0 82 7  
## 1 18 89

mean(knn\_pred != Auto\_test$mpg01)

## [1] 0.127551

## with K =125  
knn\_pred <- knn(Auto\_train[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_test[,c("cylinders", "weight", "displacement", "horsepower")], Auto\_train$mpg01, k =125)  
table(knn\_pred, Auto\_test$mpg01)

##   
## knn\_pred 0 1  
## 0 78 4  
## 1 22 92

mean(knn\_pred != Auto\_test$mpg01)

## [1**] 0.1326531**

For KNN with K=125, the error rate is lower, hence better performance model among other KNN models