3b – Aniket Maheshwari

UB Person Number: 50360266

Setting up our environment and importing important libraries:

```
### Clear the environment
rm(list = ls())


### First we will set the directory of the R script
setwd("C:/Users/anike/Desktop/Sem 1/EAS 506 Statistical Data
Mining/Homework/Homework 3")


## Loading all the libraries
library(ISLR)
library(corrplot)

## corrplot 0.90 loaded

library(MASS)
library(klaR)
library(leaps)
library(lattice)
library(ggplot2)
library(corrplot)
library(car)

## Loading required package: carData

library(caret)
library(class)
```

Importing dataset:

```
data("Weekly")
dim(Weekly)

## [1] 1089    9

str(Weekly)

## 'data.frame':    1089 obs. of  9 variables:
##  $ Year     : num  1990 1990 1990 1990 1990 1990 1990 1990 1990 1990 ...
##  $ Lag1     : num  0.816 -0.27 -2.576 3.514 0.712 ...
##  $ Lag2     : num  1.572 0.816 -0.27 -2.576 3.514 ...
##  $ Lag3     : num  -3.936 1.572 0.816 -0.27 -2.576 ...
##  $ Lag4     : num  -0.229 -3.936 1.572 0.816 -0.27 ...
```

```
## $ Lag5     : num  -3.484 -0.229 -3.936 1.572 0.816 ...
## $ Volume   : num  0.155 0.149 0.16 0.162 0.154 ...
## $ Today    : num  -0.27 -2.576 3.514 0.712 1.178 ...
## $ Direction: Factor w/ 2 levels "Down","Up": 1 1 2 2 2 1 2 2 2 1 ...

data1 <- Weekly
```

So the dataset has 1089 rows and 9 colunms. All the colummns except for the response variable are numeric varaibles. Years are between 1990 to 2010. Our response variable is a categorical value with two categories : "Up" and "Down"

Before starting EDA, first i'll check whether the data has any missing values or not:

```
NAmat = matrix(as.numeric(is.na(data1)) , ncol = 9)
#head(NAmat,50)
nonNAdx = which(rowSums(NAmat) == 0)
length(nonNAdx) ## so no missing value as length of nonNAdx is equal to number of rows in dataset

## [1] 1089

dim(data1)

## [1] 1089    9
```

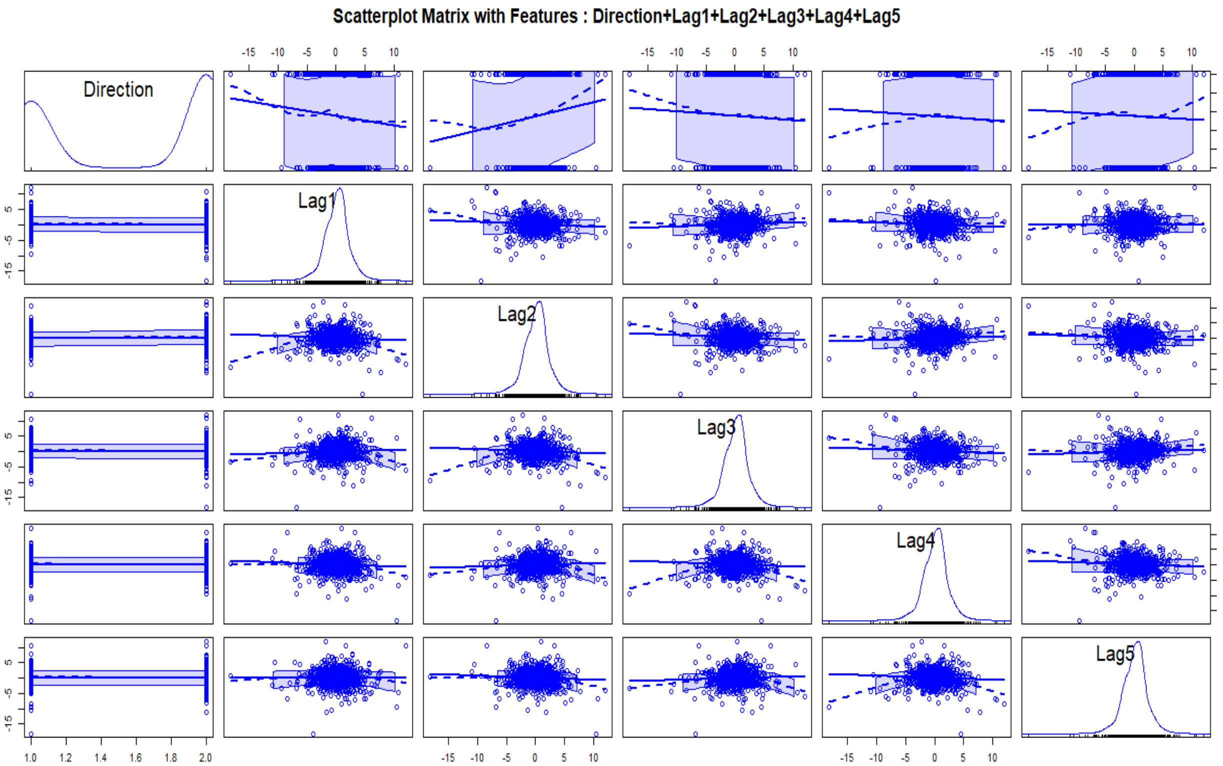
So there are no missing values in the dataset.

Part A)

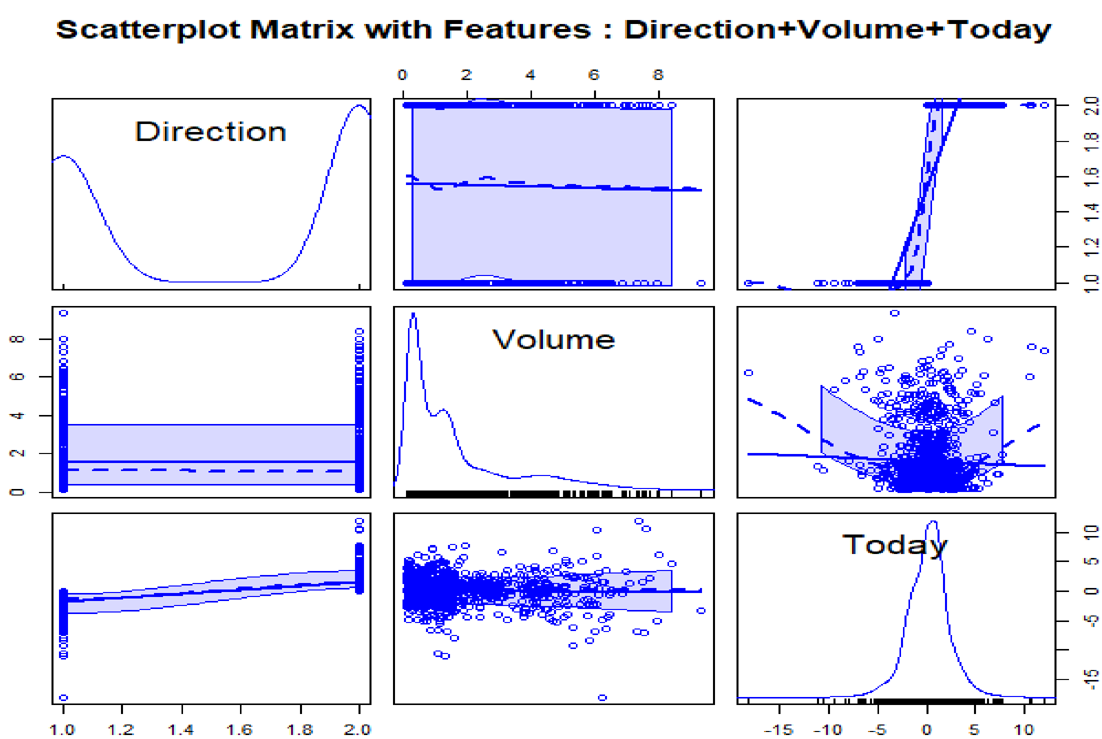
Visualizing the data set:

a) Scatter-Plots:

```
scatterplotMatrix(~Direction+Lag1+Lag2+Lag3+Lag4+Lag5, data=data1, main="Scatterplot Matrix with Features : Direction+Lag1+Lag2+Lag3+Lag4+Lag5")
```



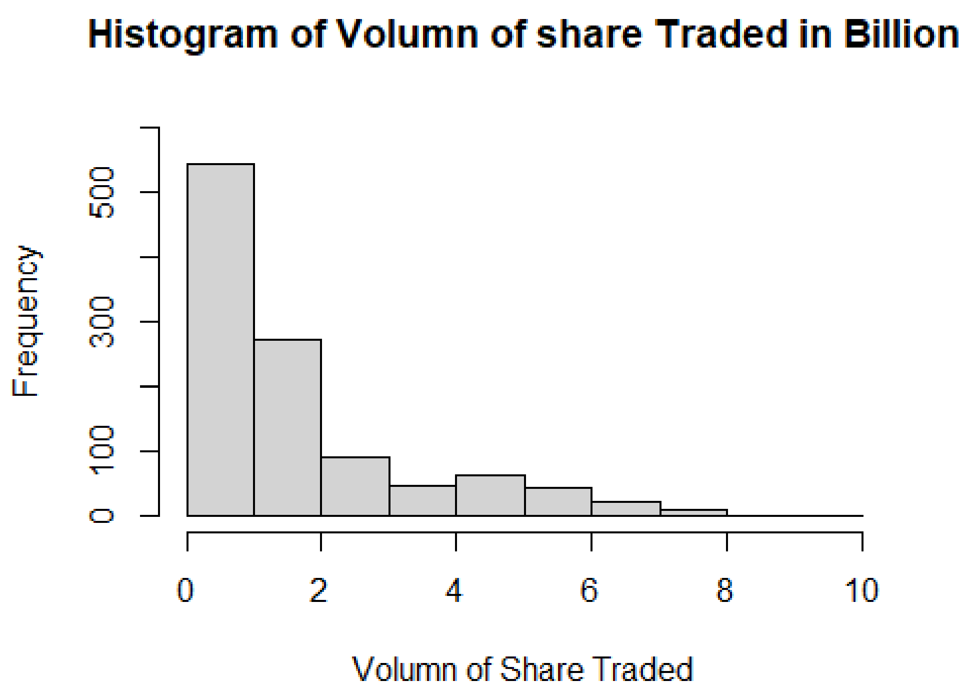

```
scatterplotMatrix(~Direction+Volume+Today, data=data1, main="Scatterplot Matrix with Features : Direction+Volume+Today")
```



From the plots above it can be told that except volume feature all the other features have normal distribution. All the features are also co-related to each other as the points in all the graph forms a cluster in the center of the graph meaning the points of both the feature in a given graph are overlapping each other.

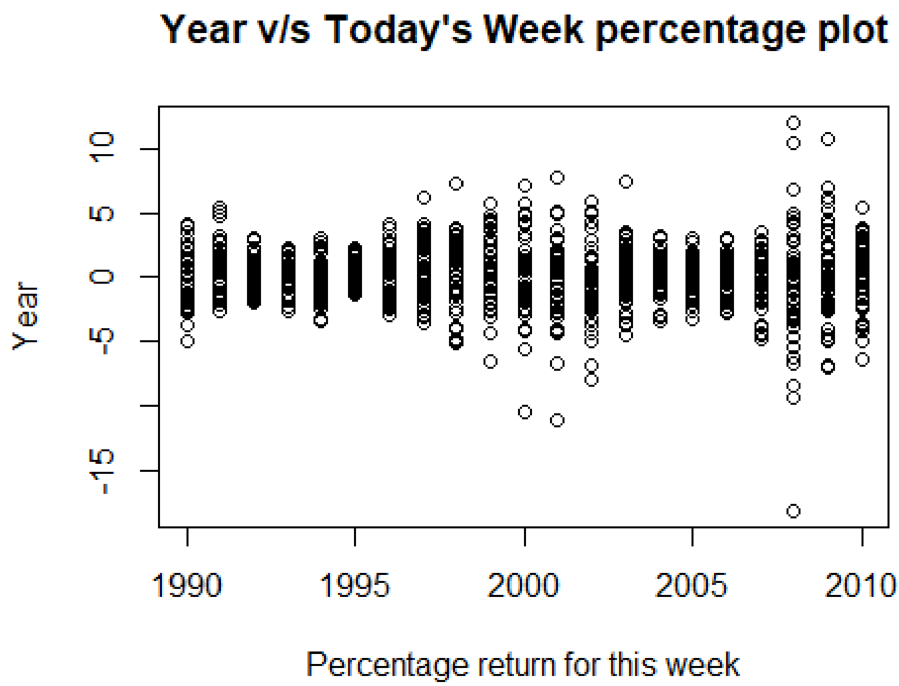
b) Histogram:

```
hist(Weekly$Volume , ylim = c(0,600) , xlab = "Volumn of Share Traded"  ,
main = "Histogram of Volumn of share Traded in Billion")
```



c) pair-plots:

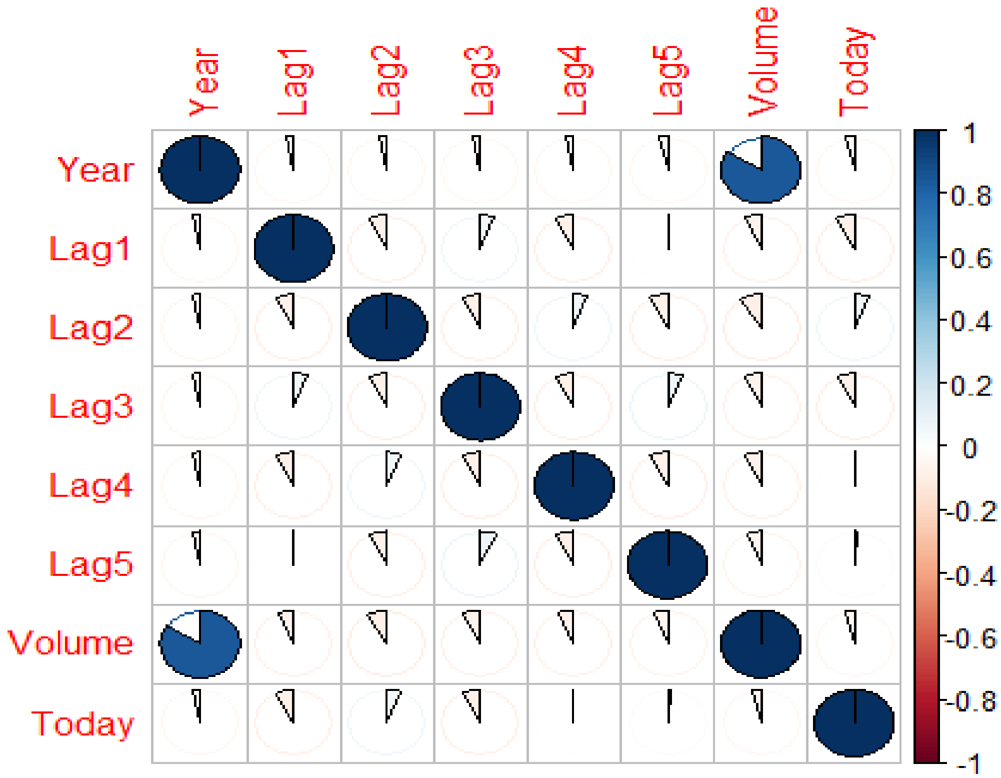
```
plot(Weekly$Year,Weekly$Today , xlab = "Percentage return for this week" ,
ylab = "Year" , main= "Year v/s Today's Week percentage plot")
```



So, 2008 has the most volume of highest number of shares traded. This make sense because of the Stock Market Crash.

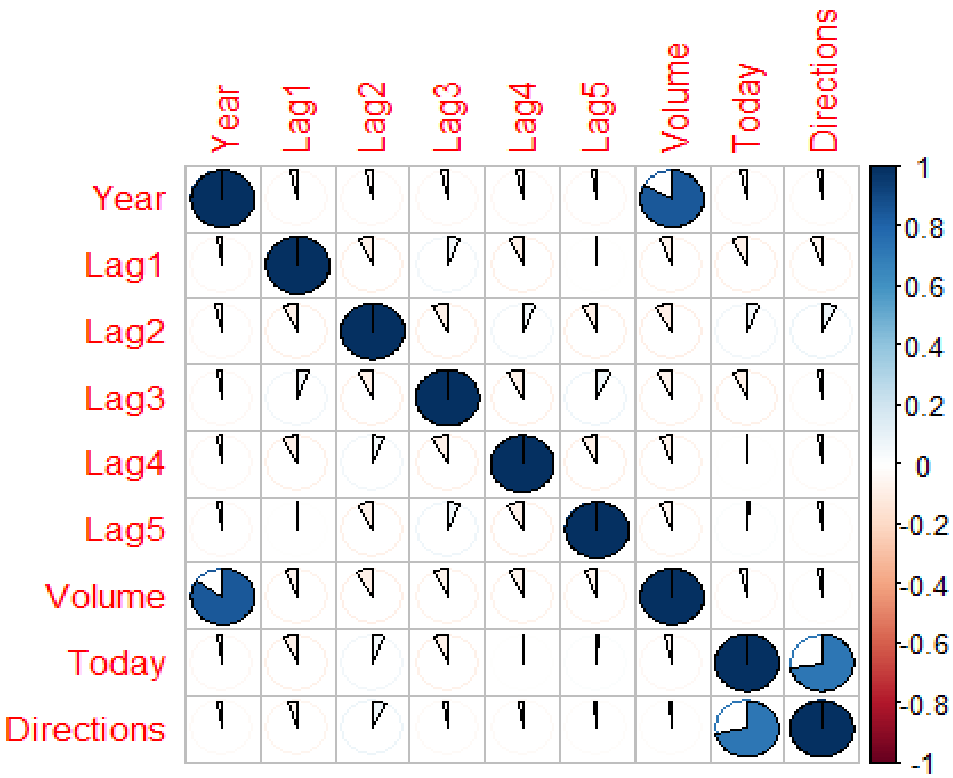
d) Co-Relation Plot:

```
corrplot(cor(data1[,1:8]), method="pie")
```



So, Volume and Year are highly co-related to each other that means volumes of share trading as increased over the years. All the other features are positively corelated to each other as well. Now, In this Co-relation plot Direction could not be taken as a feature because it's not numeric. So to add Direction to my co-relation plot i first converted it into numeric.

```
Directions <- ifelse(data1$Direction == "Up", 1 , 0)
data2 <- data1[,1:8]
data2 <- cbind(data2 , Directions)
## correlation plot with target feature
corrplot(cor(data2), method="pie")
```



Our target feature, Direction, is highly corelated to Today variable but that's not much of a suprise as Today feature tells the Percentage return for this week.

Part B)

Logistic Regression: Logistic Regression uses linear regression with the addition of sigmoid function which helps in returning output in between 0-1 range. Here as i have two categorical features 'UP' and 'DOWN', if the logistic regression returns value lower than 0.5 I'll classify that as 'DOWN' and it returns value more than that then I'll classify that as 'UP'.

R uses glm.net to do logistic regression.

```
logistic_reg <- glm(Direction~Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,data
= data1, family = binomial)
logistic_reg

##
## Call:  glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = data1)
##
## Coefficients:
## (Intercept)         Lag1         Lag2         Lag3         Lag4
Lag5
##     0.26686     -0.04127      0.05844     -0.01606     -0.02779     -
0.01447
##      Volume
##    -0.02274
```

```
##
## Degrees of Freedom: 1088 Total (i.e. Null);  1082 Residual
## Null Deviance:       1496
## Residual Deviance: 1486  AIC: 1500
```

```
summary(logistic_reg)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = data1)
##
## 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
```

Here, it seems like Lag2 is the most significant feature with the significance code 0.05.

Part C)

Fitting the Logistic Model and Finding accuracy and error.

```
pred_model = predict(logistic_reg, type="response")

pred_values = rep("Down", length(data1$Direction))
pred_values[pred_model > 0.5] = "Up"

confusion_matrix <- table(pred_values, data1$Direction)
confusion_matrix1 <- confusionMatrix(confusion_matrix)
confusion_matrix1
```

```
## Confusion Matrix and Statistics
##
##
## pred_values Down  Up
##        Down   54  48
##        Up    430 557
##
##                Accuracy : 0.5611
##                  95% CI : (0.531, 0.5908)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : 0.369
##
##                   Kappa : 0.035
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.11157
##             Specificity : 0.92066
##          Pos Pred Value : 0.52941
##          Neg Pred Value : 0.56434
##              Prevalence : 0.44444
##          Detection Rate : 0.04959
##    Detection Prevalence : 0.09366
##       Balanced Accuracy : 0.51612
##
##        'Positive' Class : Down
##
```

So the confusion matrix of logistic regression model tells me that we have 54 points in TRUE Positive , 48 points in False negative , 430 points in False Positive and 557 points in true negative.

```
accuracy_logistic_model <- 100 * confusion_matrix1$overall[1]
##Accuracy is : 56.11
round(accuracy_logistic_model, digits = 2)

## Accuracy
##    56.11

rounded_acc <- round(accuracy_logistic_model, digits = 2)
error = 100 -  as.numeric(rounded_acc)
error

## [1] 43.89
```

We get the accuracy of 56.11% and error rate of 43.89% from logistic regression model.

Part D)

Splitting The data: Now I'll split the data into train and test set according to the year feature. All the data points that are between 1990 - 2008 will be in my training set and points from 2009 & 2010 will be in test set.

```
train_data <- subset(data1 , Year < 2009)
test_data <- subset(data1, Year > 2008)
head(train_data , 3)
```

```
##   Year   Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
```

I'll fit the train data using Lag2 as the only predictor on the test set using logistic regression.

```
training_log_reg <- glm(Direction~Lag2, data = train_data , family =
binomial)

## now I'll predict the fitted model on data of 2009 and 2010 (test data)
test_prediction <- predict(training_log_reg , test_data , type = "response")

#computing confusion matrix
y_test <- rep("Down", length(test_data$Direction))
y_test[test_prediction > 0.5] = "Up"
test_confusion_matrix <- table(y_test , test_data$Direction)
test_confusion_matrix1 <- confusionMatrix(test_confusion_matrix)
test_confusion_matrix1
```

```
## Confusion Matrix and Statistics
##
##
## y_test Down Up
##   Down    9  5
##   Up     34 56
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.2439
##
##                   Kappa : 0.1414
##
##  Mcnemar's Test P-Value : 7.34e-06
##
##             Sensitivity : 0.20930
##             Specificity : 0.91803
##          Pos Pred Value : 0.64286
##          Neg Pred Value : 0.62222
```

```
##               Prevalence : 0.41346
##           Detection Rate : 0.08654
##     Detection Prevalence : 0.13462
##        Balanced Accuracy : 0.56367
##
##         'Positive' Class : Down
##
```

```
#Accuracy is: 62.5
round(test_confusion_matrix1$overall[1]*100 , digits = 2)
```

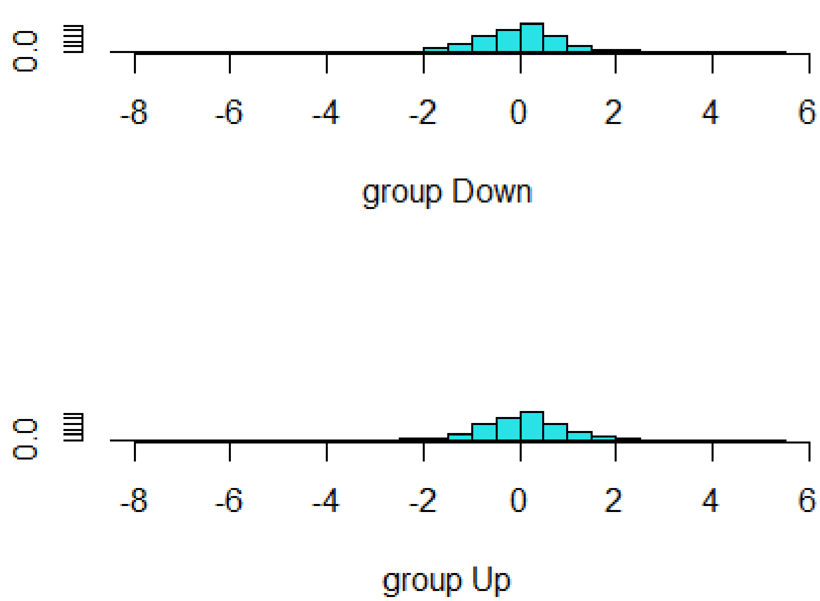
```
## Accuracy
##     62.5
```

I get the accuracy of 62.5% and error rate of 37.5% from logistic regression model with just Lag2 as predictor. This is much better than the model with all the features as predictors.

Part E)

LDA: Linear Discriminant analysis is a true decision boundary discovery algorithm. It assumes that the class has common covariance and it's decision boundary is linear separating the class.

```
train_lda <- lda(Direction~Lag2 , data = train_data)
plot(train_lda)
```



This plot tells me that the two classes are overlapping. This will cause a lot of misclassification in the LDA model.

Fitting LDA to test set:

```
test_pred_lda <- predict(train_lda, test_data)
confusion_matrix_lda <- table(test_pred_lda$class , test_data$Direction)
confusion_matrix_lda1 <- confusionMatrix(confusion_matrix_lda)
# Accuracy is : 62.5
round(confusion_matrix_lda1$overall[1]*100 , digits = 2)

## Accuracy
##     62.5
```

I get the accuracy of 62.5% and error rate of 37.5% from LDA model with just Lag2 as predictor. This is exactly the same accuracy that i got from logistic regression with just Lag2 as predictor.

Part F)

KNN: I'll use the same train and test dataset on KNN with K = 1 neighbour.

```
x_train <- subset(train_data , select = -c(9))
x_test <- subset(test_data , select = -c(9))
set.seed(1)
testing_knn <- knn(x_train , x_test , train_data$Direction , k=1)
confusion_matrix_knn <- table(testing_knn , test_data$Direction)
confusion_matrix_knn1 <- confusionMatrix(confusion_matrix_knn)

# Accuracy of KNN is : 79.81
round(confusion_matrix_knn1$overall[1]*100 , digits = 2)

## Accuracy
##    79.81
```

I get the accuracy of 79.81% and error rate of 20.19% from KNN model with K=1. This is even worse than LDA and logistic regression.

Part G)

MODEL	**ACCURACY**
Logistic Regression with all predictors	**56.11**
Logistic Regression with Lag2 only as predictor	**62.5**
LDA	**62.5**
KNN with K=1	**79.81**

So, out of all the models, LDA and Logistic Regression with Lag2 as predictors worked the best. As I saw in the scatterplots that the data follows normal distribution so it's anyway better to use LDA as it separates classes by drawing a linear decision boundary.

Part H)

Transforming and interacting with predictors:

a) Logistic Regression: Predictor today: Lag1
First I would like to take today (this week) and Lag1 (last week) as predictors:

```
logistic_reg1 <- glm(Direction~Lag1:Today, data = train_data, family =
binomial)
pred_model1 = predict(logistic_reg1, type="response")

pred_values1 = rep("Down", length(train_data$Direction))
pred_values1[pred_model1 > 0.5] = "Up"

#confusion matrix
confusion_matrix_interaction1 <- table(pred_values1, train_data$Direction)
confusion_matrix_interaction2 <- confusionMatrix(confusion_matrix)
accuracy_logistic_model <- 100 * confusion_matrix_interaction2$overall[1]
##Accuracy is : 55.43
round(accuracy_logistic_model, digits = 2)

## Accuracy
##    56.11
```

This gives accuracy of 56.11, which is worst than any model i have used on this dataset until now.

b) Logistic Regression: Predictor Lag1*Lag2*Lag3*Lag4*Lag5

```
train_lda1 <- lda(Direction~Lag1*Lag2*Lag3*Lag4*Lag5 , data = train_data)

## fitting model to test data ##
test_pred_lda1 <- predict(train_lda1, test_data)


## confusion matrix and computing error
confusion_matrix_lda_interaction1 <- table(test_pred_lda1$class ,
test_data$Direction)
confusion_matrix_lda_interaction2 <-
confusionMatrix(confusion_matrix_lda_interaction1)
# Accuracy is : 51.92
round(confusion_matrix_lda_interaction2$overall[1]*100 , digits = 2)

## Accuracy
##    51.92
```

c) KNN where K=5

```
set.seed(1)
testing_knn1 <- knn(x_train , x_test , train_data$Direction , k=5)
confusion_matrix_knn2 <- table(testing_knn1 , test_data$Direction)
confusion_matrix_knn3 <- confusionMatrix(confusion_matrix_knn2)

# Accuracy of KNN is : 88.46
round(confusion_matrix_knn3$overall[1]*100 , digits = 2)

## Accuracy
##    88.46
```

The accuracy is 88.46 i.e error rate is 11.54. This is the best model i have used on this dataset.

d) KNN where k=10

```
set.seed(1)
testing_knn2 <- knn(x_train , x_test , train_data$Direction , k=10)
confusion_matrix_knn4 <- table(testing_knn2 , test_data$Direction)
confusion_matrix_knn5 <- confusionMatrix(confusion_matrix_knn4)

# Accuracy of KNN is : 85.58
round(confusion_matrix_knn5$overall[1]*100 , digits = 2)

## Accuracy
##    85.58
```

The accuracy is 85.58 i.e error rate is 14.42. So, KNN with k=1 had accuracy of 79.81 . K = 5 had accuracy of 88.46 and k=10 has accuracy of 85.58. As i will keep increasing the k value the accuracy will keep on dropping now.