STATISTICAL DATA MINING

HOMEWORK 3

NAME: SHREERAM G S

UB PERSON NUMBER: 50413349

- a) To begin with, we need to install package tidyverse. The tidyverse is a coherent system of packages for data manipulation, exploration and visualization.

```
> install.packages('tidyverse')
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/Sriram/Documents/R/win-library/4.1'
(as 'lib' is unspecified)
also installing the dependencies 'sys', 'ps', 'base64enc', 'fastmap', 'rappdirs', 'askpass', 'processx', 'htmltools', 'tinytex', 'jquerylib', 'assertthat', 'blob', 'DBI', 'gargle', 'uuid', 'ids', 'rematch2', 'mime', 'openssl', 'callr', 'fs', 'rmarkdown', 'selectr', 'dbplyr', 'dtplyr', 'googledrive', 'googlesheets4', 'httr', 'jsonlite', 'modelr', 'reprex', 'rstudioapi', 'rvest', 'xml2'


  There is a binary version available but the source version is later:
     binary source needs_compilation
uuid  0.1-4  1.0-2             TRUE

  Binaries will be installed
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/sys_3.4.zip'
Content type 'application/zip' length 59880 bytes (58 KB)
downloaded 58 KB
```

We use set.seed() function to initialize a pseudorandom number generator.
Next we use the replicate() function with 20 features and 1000 observations. The replicate() function is used for creating simulations, as it can repeat an expression a specific number of times.
The reduce function is used to generate a quantitative response vector. Reduce() reduces a vector, x , to a single value by recursively calling a function.

```
> require(tidyverse)
Loading required package: tidyverse
-- Attaching packages --------------------------------------------------------
v ggplot2 3.3.5     v purrr   0.3.4
v tibble  3.1.4     v dplyr   1.0.7
v tidyr   1.1.4     v stringr 1.4.0
v readr   2.0.2     v forcats 0.5.1
-- Conflicts -----------------------------------------------------------------
x tidyr::expand() masks Matrix::expand()
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
x tidyr::pack()   masks Matrix::pack()
x tidyr::unpack() masks Matrix::unpack()
> set.seed(1) #set seed(1) means to initialize a pseudorandom number generator
> df <- data.frame(replicate(20, rnorm(n = 1000)))
> #quantitaive response vector
> df %>%
+   reduce(function(y, x) y + ifelse(runif(1) < 0.5,
+                                    rnorm(1, mean = 5, sd = 1),
+                                    0)*x + rnorm(1000)) -> df$Y
> source("~/.active-rstudio-document")
```

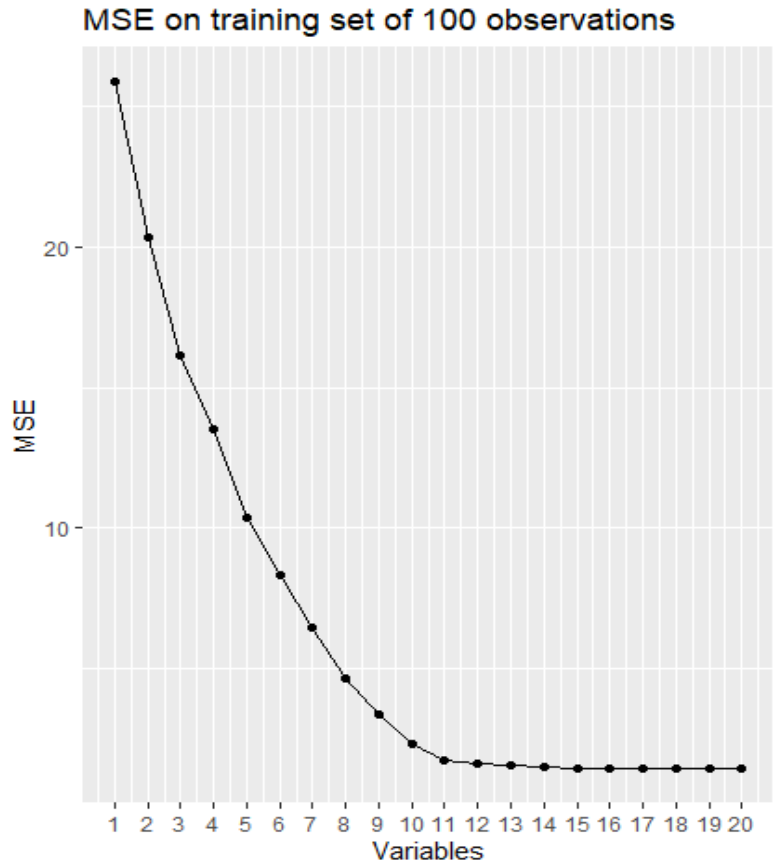
b) The caret package (Classification And REgression Training) contains functions to streamline the model training process for complex regression and classification problems.
The CreateDataPartition function is used to split the data, where p = 0.1. P is 0.1 because when splitting, " x.train <- df[set, -21] " and "y.train <- df[set, 21]", p = 0.1 means 0.1 of 1000 observation, hence x.tain and y.train has 100 observations. That means training set has 100 observations.
Where as in x.test and y.test for test data set, it is df[-set, -21], i.e., -set => 1000 -100 => 900 observations.

```
> require(caret)
>
> set <- createDataPartition(df$Y, p = 0.1, list = F)
>
> x.train <- df[set, -21]
> y.train <- df[set, 21]
> x.test <- df[-set, -21]
> y.test <- df[-set, 21]
>
```

c) To perform best subset we use function regsubsets. The best subset selection by identifying the best model that contains a given number of predictors, where best is quantified using RSS.

```
> best.subset <- regsubsets(x = x.train, y = y.train, nvmax = 20)
>
> best.subset.summary <- summary(best.subset)
>
> data_frame(MSE = best.subset.summary$rss/900) %>%
+   mutate(id = row_number()) %>%
+   ggplot(aes(id, MSE)) +
+   ggtitle('MSE on training set of 100 observations')+
+   geom_line() + geom_point() +
+   xlab('Variables')  +
+   scale_x_continuous(breaks = 1:20)
```

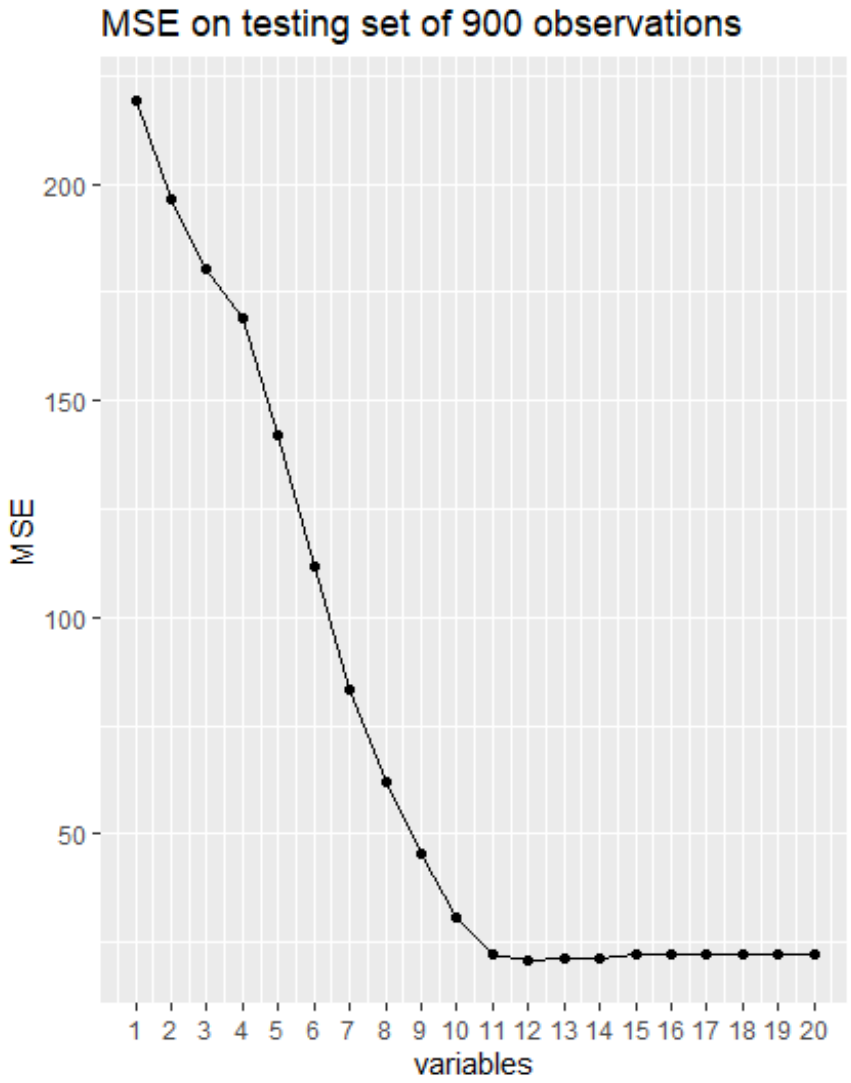
From this we get the plot between the mse values and the number of variables used.



d) First let us create test error variables of size 20 using rep() function. The rep() is a built-in R function that replicates the values in the provided vector.

```
> test.er = rep(NA,19)
> matrix.test <- model.matrix(Y ~ ., data = df[-set,])
> for (i in 1:20){
+   test.coefficients = coef(best.subset, id=i)
+   predictor = matrix.test[,names(test.coefficients)]%*%test.coefficients
+   test.er[i] = mean((y.test-predictor)^2)
+ }
>
>
> data_frame(MSE = test.er) %>%
+   mutate(id = row_number()) %>%
+   ggplot(aes(id, MSE)) +
+   geom_line() + geom_point() +
+   xlab('variables') +
+   ggtitle('MSE on testing set of 900 observations ') +
+   scale_x_continuous(breaks = 1:20)
>
```

The plots derived from this is:



e)Using which min() function with e0 variables used test set as parameter, we get the minimum MSE is achieved using at least 12 variables.

```
> which.min(test.er)
[1] 12
>
```

f) in reduce function above, we have done runif(1) < 0.5. that is the random numbers generated is less than 0.5 of the variables. Hence the co – effiecients of the numbers wher runif(1) > 0.5 is 0. Therfore 50 % of the

co-efficients will be 0. As there are 20 variables in total, 50% of this means 10. So 10 variables co efficients shall be 0.

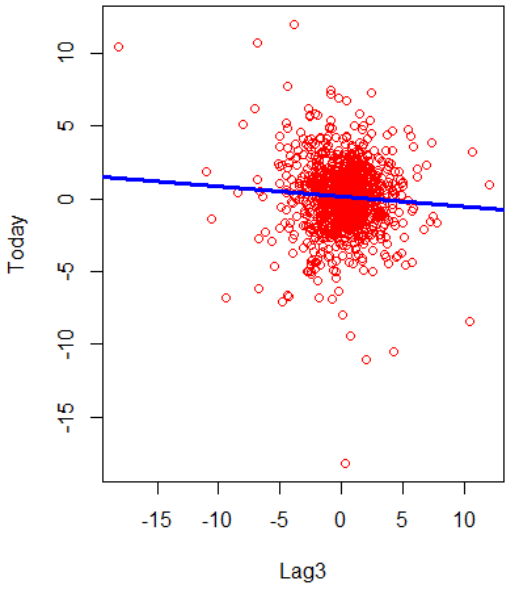
g)

2) a) Load the necessary packages (ISLR,MASS,Class). Using the summary function we can get the numerical summary of the weekly data.

```
> require(ISLR)
Loading required package: ISLR

Attaching package: 'ISLR'

The following object is masked _by_ '.GlobalEnv':

    College

> require(MASS)
Loading required package: MASS

Attaching package: 'MASS'

The following object is masked from 'package:dplyr':

    select

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

Next we shall use scatter plot and plot between today and lag 5 which we saw using the summary function to plot a graph.

```
#scatter plot of Today with respect of lag 3
plot(Today~Lag3, col="red", data=Weekly)
lm.function = lm(Today~Lag3, data=Weekly)
abline(lm.function, lwd= 3, col= "blue")
```



b) Using glm(generalized linear model) function, we can generate a logistic regression model. And after doing the model, upon reviewing the summary we can see that that lag2 has * in its summary. As it's p value < 0.05, we can say it is a good fit. The rest are clearly insignificant.

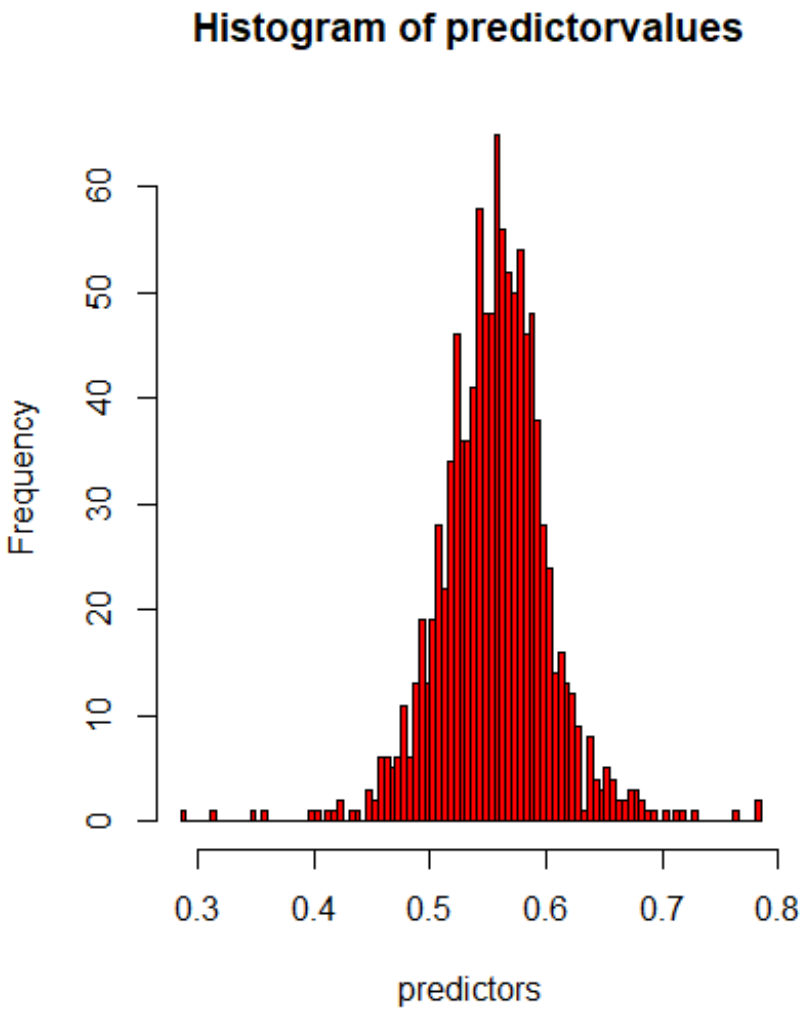
```
> #linear regression model
> reg.model = glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,family = "binomial", data=Weekly)
> summary(reg.model)

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
```

c) The predict() function is used to predict the values based on the input data from the regression model. The rep() replicates the predicted values and makes a list of predictors generated from the regression model. Using the table function we can view the confusion matrix. We can observe that most cases are goin UP, which means there is a large percentage of true positives. We find 557 true positives and 430 false positives. The true negatives value is 54/(54+430) i.e., 0.11157024793 which is very small. So 11% is true negatives. Similarly

By looking at the value of the mean, we can say that model predicts the weekly trend by 0.561 i.e, by nearly 56%.

```
> pred.value = predict(reg.model, type="response")
> predictor = rep("Down", length(pred.value))
> predictor[pred.value > 0.5] = "Up"
> table(predictor, Weekly$Direction)

predictor Down  Up
     Down   54  48
     Up    430 557
> mean(predictor ==  Weekly$Direction )
[1] 0.5610652
>
```

The graphical representation using histogram is:



Here we can see that chances of being is more 50%, hence we can keep criterion for as sample to going up is 0.5.

d)

```
> #train data set from 1990 - 2008
> train.set <- (Weekly$Year < 2009)
> weekly.train <- Weekly[train.set,]
> weekly.set <- Weekly[!train.set,]
> Direction_train <- weekly.train$Direction
> Direction_test <- weekly.set$Direction
>
> weekly.regression <- glm(Direction ~ Lag2,
+                          data = weekly.train,
+                          family = binomial)
> View(weekly.regression)
```

weekly.regression	list [30] (S3: glm, lm)	List of length 30
coefficients	double [2]	0.2033 0.0581
residuals	double [985]	-2.34 -2.28 1.83 1.95 1.67 -2.28 ...
fitted.values	double [985]	0.573 0.562 0.547 0.513 0.600 0.561 ...
effects	double [985]	-3.259 -2.024 0.940 0.967 0.902 -1.086 ...
R	double [2 x 2]	-15.57 0.00 -1.61 -34.84
rank	integer [1]	2
qr	list [5] (S3: qr)	List of length 5
family	list [12] (S3: family)	List of length 12
linear.predictors	double [985]	0.2946 0.2507 0.1876 0.0536 0.4074 0.2446 ...
deviance	double [1]	1350.543
aic	double [1]	1354.543
null.deviance	double [1]	1354.71
iter	integer [1]	4
weights	double [985]	0.245 0.246 0.248 0.250 0.240 0.246 ...
prior.weights	double [985]	1 1 1 1 1 1 ...
df.residual	integer [1]	983
df.null	integer [1]	984
y	double [985]	0 0 1 1 1 0 ...
converged	logical [1]	TRUE
boundary	logical [1]	FALSE
model	list [985 x 2] (S3: data.frame)	A data.frame with 985 rows and 2 columns
call	language	glm(formula = Direction ~ Lag2, family = binomial, data = weekly.train)
formula	formula	Direction ~ Lag2
terms	formula	Direction ~ Lag2
data	list [985 x 9] (S3: data.frame)	A data.frame with 985 rows and 9 columns
offset	NULL	Pairlist of length 0
control	list [3]	List of length 3
method	character [1]	'glm.fit'
contrasts	NULL	Pairlist of length 0
xlevels	list [0]	List of length 0

The confusion matrix found is:

```
> pred.value2 <- predict(weekly.regression, Weekly_test, type = "response")
> predictor2 = rep("Down", length(Direction_test))
> predictor2[pred.value2 > 0.5] <- "Up"
> table(predictor2, Direction_test)
          Direction_test
predictor2 Down Up
      Down    9  5
      Up     34 56
> mean(predictor2 == Direction_test)
[1] 0.625
>
```

From the mean we come to know that model predicted correctly 62.5% after splitting the data. 9 /(34+9) => 0.20930232558 is the true negatives value. Compared to previous model which was 11.1%, the value now has increased to 20.9%.

e) From this confusion matrix we can see that model predicted 62.5% correctly

```
> weekly.lda <- lda(Direction ~ Lag2, data = Weekly, subset = train)
> weekly.lda
Call:
lda(Direction ~ Lag2, data = Weekly, subset = train)

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
> pred.value3 <- predict(weekly.lda, Weekly_test)
> mean(pred.value3$class == Direction_test)
[1] 0.625
```

f) By using knn = 1, From this confusion matrix we can see that model predicted 50% correctly

```
> train.x <- as.matrix(Weekly$Lag2[train])
> test.x <- as.matrix(Weekly$Lag2[!train])
>
> set.seed(1)
> knn.predictors <- knn(train.x, test.x, Direction_train, k = 1)
> mean(knn.predictors == Direction_test)
[1] 0.5
```

g) We know that when knn = 1, the model accuracy is 0.5, so if we change the KNN value from 1 to 2, then we get the model acuracy as 50.9%, with true negatives of 19/(19+24) = 0.44186046511 i.e, 44%. Similarly, when we change KNN to a higher value say 7, then we get model accuracy as 53.84%. So clearly from this observation we cany that model accuracy increases with increase in KNN values.

```
> train.x <- as.matrix(Weekly$Lag2[train])
> test.x <- as.matrix(Weekly$Lag2[!train])
>
> set.seed(1)
> knn.predictors <- knn(train.x, test.x, Direction_train, k = 1)
> mean(knn.predictors == Direction_test)
[1] 0.5
> train.x <- as.matrix(Weekly$Lag2[train])
> test.x <- as.matrix(Weekly$Lag2[!train])
>
> set.seed(1)
> knn.predictors2 <- knn(train.x, test.x, Direction_train, k = 2)
> table(knn.predictors2, Direction_test)
              Direction_test
knn.predictors2 Down Up
           Down   19 27
           Up     24 34
> mean(knn.predictors2 == Direction_test)
[1] 0.5096154
>
> train.x <- as.matrix(Weekly$Lag2[train])
> test.x <- as.matrix(Weekly$Lag2[!train])
>
> set.seed(1)
> knn.predictors.7 <- knn(train.x, test.x, Direction_train, k = 7)
> table(knn.predictors.7, Direction_test)
                Direction_test
knn.predictors.7 Down Up
            Down   15 20
            Up     28 41
> mean(knn.predictors.7 == Direction_test)
[1] 0.5384615
```

- Logistic regression with lag4 giving accuracy of 0.55 i.e., 55%

```
> #logistic with Lag4
> reg.model = glm(Direction~Lag4,family = "binomial", data=Weekly)
> summary(reg.model)

Call:
glm(formula = Direction ~ Lag4, family = "binomial", data = Weekly)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.416  -1.270   1.065   1.086   1.161

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22580    0.06113   3.694 0.000221 ***
Lag4        -0.01757    0.02592  -0.678 0.497930
---
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: 1495.7  on 1087  degrees of freedom
AIC: 1499.7

Number of Fisher Scoring iterations: 3

>
> pred.value = predict(reg.model, type="response")
> predictor = rep("Down", length(pred.value))
> predictor[pred.value > 0.5] = "Up"
> table(predictor, Weekly$Direction)

predictor Down  Up
       Up  484 605
> mean(predictor ==  Weekly$Direction )
[1] 0.5555556
>
```

- Logistic regression with lag2 giving accuracy of 56.1%

```
> reg.model = glm(Direction~Lag2,family = "binomial", data=Weekly)
> summary(reg.model)

Call:
glm(formula = Direction ~ Lag2, family = "binomial", data = Weekly)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.564  -1.267   1.008   1.086   1.386

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.21473    0.06123   3.507 0.000453 ***
Lag2         0.06279    0.02636   2.382 0.017230 *
---
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: 1490.4  on 1087  degrees of freedom
AIC: 1494.4

Number of Fisher Scoring iterations: 4

>
> pred.value = predict(reg.model, type="response")
> predictor = rep("Down", length(pred.value))
> predictor[pred.value > 0.5] = "Up"
> table(predictor, Weekly$Direction)

predictor Down  Up
     Down   33  26
     Up    451 579
> mean(predictor ==  Weekly$Direction )
[1] 0.5619835
```

- LDA with lag2 gives you accuracy of 0.62 i.e., 62.5%

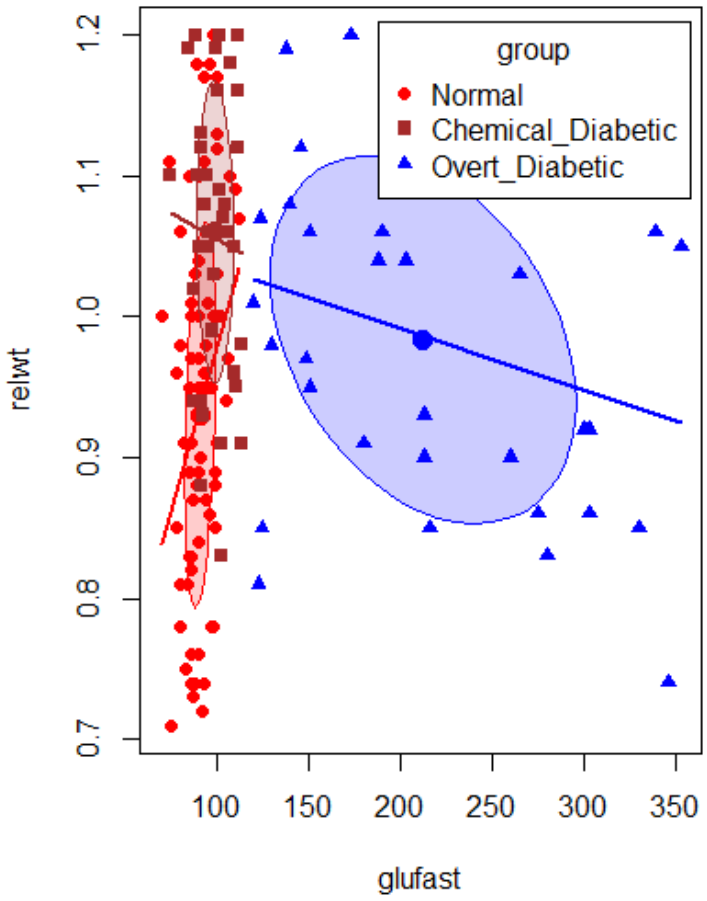
```
> train.x <- as.matrix(Weekly$Lag2[train])
> test.x <- as.matrix(Weekly$Lag2[!train])
>
> set.seed(1)
> knn.predictors <- knn(train.x, test.x, Direction_train, k = 1)
> mean(knn.predictors == Direction_test)
[1] 0.5
>
>
>
> eekly.lda <- lda(Direction ~ Lag3, data = Weekly, subset = train)
> weekly.lda
Call:
lda(Direction ~ Lag2, data = Weekly, subset = train)

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
> pred.value3 <- predict(weekly.lda, Weekly_test)
> mean(pred.value3$class == Direction_test)
[1] 0.625
```

3) a) First we need to install “HE plot” and “candisc” packages for visualizing hypothesis tests in multivariate linear models.

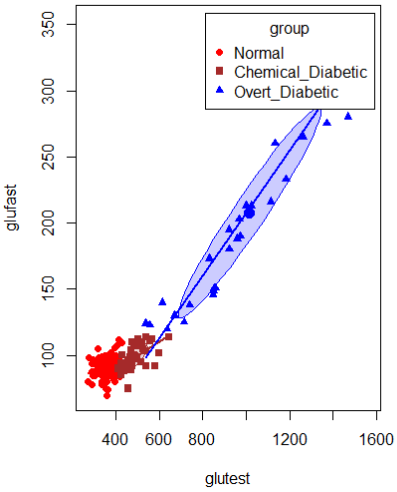
```
> library(heplots)
> library(candisc)
> data(Diabetes, package="heplots")
> str(Diabetes)
'data.frame':   145 obs. of  6 variables:
 $ relwt  : num  0.81 0.95 0.94 1.04 1 0.76 0.91 1.1 0.99 0.78 ...
 $ glufast: int  80 97 105 90 90 86 100 85 97 97 ...
 $ glutest: int  356 289 319 356 323 381 350 301 379 296 ...
 $ instest: int  124 117 143 199 240 157 221 186 142 131 ...
 $ sspg   : int  55 76 105 108 143 165 119 105 98 94 ...
 $ group  : Factor w/ 3 levels "Normal","Chemical_Diabetic",..: 1 1 1 1 1 1 1 1 1 1 ...
>
```

The variables present in the data frame are : relwt => relative weight, glufast, glutest, instest, sspg and group. Let us start plotting between the variables. Let us first begin with relwt and glufast w.r.t group.

```
scatterplot( relwt   ~ glufast | group, data=Diabetes,
             pch=c(16,15,17),
             col=c("red", "brown", "blue"),
             smooth=FALSE,
             grid=FALSE,
             legend=list(coords="topright"),
             lwd=2,
             ellipse=list(levels=0.5))
relwt
```



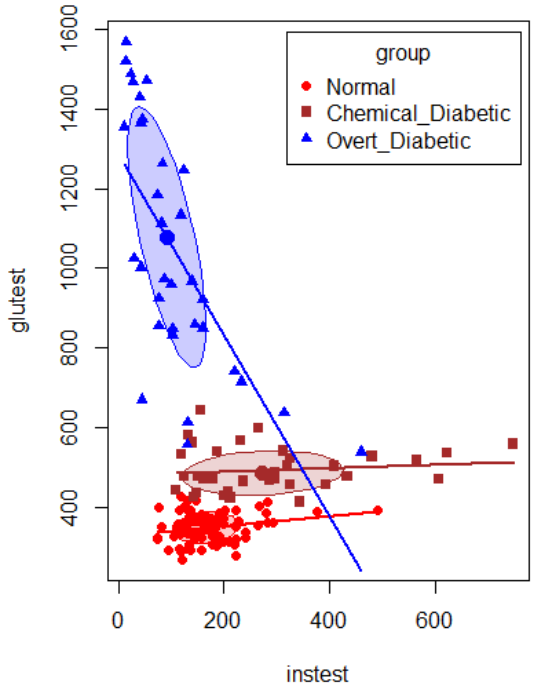
- Now we shall plot between glufast and glutest with respect to group.

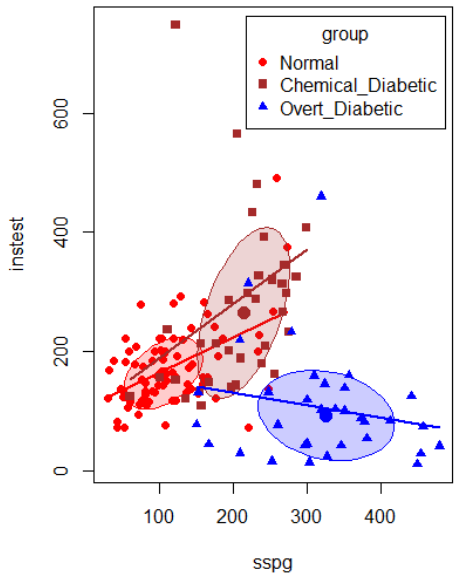



```
> scatterplot( glufast   ~ glutest | group, data=Diabetes,
+              pch=c(16,15,17),
+              col=c("red", "brown", "blue"),
+              smooth=FALSE,
+              grid=FALSE,
+              legend=list(coords="topright"),
+              lwd=2,
+              ellipse=list(levels=0.5))
> relwt
```

- Scatter plot between glutest and instest with respect to group.

```
scatterplot( glutest   ~instest   | group, data=Diabetes,
             pch=c(16,15,17),
             col=c("red", "brown", "blue"),
             smooth=FALSE,
             grid=FALSE,
             legend=list(coords="topright"),
             lwd=2,
             ellipse=list(levels=0.5))
relwt |
```





- Scatter plot between instest and ssng with respect to group.

```
> scatterplot( instest  ~ sspg | group, data=Diabetes,
+              pch=c(16,15,17),
+              col=c("red", "brown", "blue"),
+              smooth=FALSE,
+              grid=FALSE,
+              legend=list(coords="topright"),
+              lwd=2,
+              ellipse=list(levels=0.5))
>
```

Based on the above we can clearly say that is mixture of variance-covariance matrices. Thethe overt diabetic group the large variances when compared to other normal groups. With isntest and sspg there is direct progression from normal to overt as sspg value increases. In case of isntest and glufast, overt diabetic increases as glufast value increases. Same between glufast and glutest, where the progression of overt is depende on increase in value of glutest.

b) LDA

```
> Diabetes.lda <- lda(group ~ glufast + glutest + instest + sspg, data = Diabetes)
> Diabetes.lda
Call:
lda(group ~ glufast + glutest + instest + sspg, data = Diabetes)

Prior probabilities of groups:
           Normal Chemical_Diabetic    Overt_Diabetic
        0.5241379         0.2482759         0.2275862

Group means:
                   glufast   glutest  instest     sspg
Normal             91.18421  349.9737 172.6447 114.0000
Chemical_Diabetic  99.30556  493.9444 288.0000 208.9722
Overt_Diabetic    217.66667 1043.7576 106.0000 318.8788

Coefficients of linear discriminants:
                  LD1          LD2
glufast  0.0326456843  0.038927713
glutest -0.0119958220 -0.006559667
instest  0.0003148926 -0.006201852
sspg    -0.0056268696 -0.003231462

Proportion of trace:
   LD1    LD2
0.8958 0.1042
> prediction.value1 <- predict(Diabetes.lda, Diabetes)
> prediction.value1
$class
  [1] Normal            Normal            Normal            Normal            Normal            Normal
 [10] Normal            Normal            Normal            Normal            Normal            Normal
 [19] Normal            Normal            Normal            Normal            Normal            Normal
 [28] Normal            Normal            Normal            Normal            Normal            Normal
 [37] Normal            Normal            Normal            Normal            Normal            Normal
 [46] Normal            Normal            Normal            Normal            Normal            Normal
 [55] Normal            Normal            Normal            Normal            Normal            Normal
 [64] Normal            Chemical_Diabetic Normal            Normal            Normal            Chemical_Diabeti
 [73] Normal            Normal            Normal            Normal            Normal            Normal
 [82] Chemical_Diabetic Chemical_Diabetic Normal            Normal            Chemical_Diabetic Chemical_Diabeti
 [91] Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Normal
[100] Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Normal
[109] Chemical_Diabetic Normal            Chemical_Diabetic Chemical_Diabetic Overt_Diabetic    Overt_Diabetic
[118] Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic
[127] Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Chemical_Diabetic Overt_Diabetic
[136] Chemical_Diabetic Chemical_Diabetic Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic
[145] Overt_Diabetic
Levels: Normal Chemical_Diabetic Overt_Diabetic
```

QDA:

```
> Diabetes.qda <- qda(group ~ glufast + glutest + instest + sspg, data = Diabetes, subset = train)
> Diabetes.qda
Call:
qda(group ~ glufast + glutest + instest + sspg, data = Diabetes,
    subset = train)

Prior probabilities of groups:
           Normal Chemical_Diabetic    Overt_Diabetic
        0.5241379         0.2482759         0.2275862

Group means:
                    glufast   glutest  instest     sspg
Normal              91.18421  349.9737 172.6447 114.0000
Chemical_Diabetic   99.30556  493.9444 288.0000 208.9722
Overt_Diabetic     217.66667 1043.7576 106.0000 318.8788
> prediction.value2 <- predict(Diabetes.qda, Diabetes)
> prediction.value2
$class
  [1] Normal            Normal            Normal            Normal            Normal            Norma
 [10] Normal            Normal            Normal            Normal            Normal            Norma
 [19] Normal            Normal            Normal            Normal            Normal            Norma
 [28] Normal            Normal            Normal            Normal            Normal            Norma
 [37] Normal            Normal            Normal            Normal            Normal            Norma
 [46] Normal            Normal            Normal            Normal            Normal            Norma
 [55] Normal            Normal            Normal            Normal            Chemical_Diabetic Norma
 [64] Normal            Chemical_Diabetic Chemical_Diabetic Normal            Chemical_Diabetic Chemi
 [73] Normal            Normal            Normal            Normal            Normal            Norma
 [82] Chemical_Diabetic Chemical_Diabetic Normal            Chemical_Diabetic Chemical_Diabetic Chemi
 [91] Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemi
[100] Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemical_Diabetic Chemi
[109] Chemical_Diabetic Normal            Chemical_Diabetic Chemical_Diabetic Overt_Diabetic    Overt
[118] Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt
[127] Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Chemical_Diabetic Overt
[136] Chemical_Diabetic Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt_Diabetic    Overt
[145] Overt_Diabetic
Levels: Normal Chemical_Diabetic Overt_Diabetic

$posterior
           Normal Chemical_Diabetic Overt_Diabetic
1    9.992880e-01      6.354057e-04   7.658728e-05
2    9.999935e-01      3.369464e-06   3.106260e-06
3    9.999723e-01      1.165538e-05   1.607240e-05
4    9.989164e-01      1.002878e-03   8.071543e-05
5    9.991812e-01      6.362778e-04   1.825507e-04
6    9.549876e-01      4.405885e-02   9.535187e-04
7    9.995643e-01      3.485271e-04   8.714778e-05
8    9.998422e-01      1.013367e-04   5.641779e-05
9    9.979975e-01      1.951694e-03   5.077131e-05
```

LDA is used when a linear boundary is required between classifiers and QDA is used to find a non-linear boundary between classifiers.

The mean from predictors of lda with of posterior and x are as follows:

```
> mean(prediction.value1$posterior)
[1] 0.3333333
> mean(prediction.value1$x)
[1] 3.990911e-17
>
```

Hence from this we can infer that the Linear regression model predicts posterior with 0.3 accuracy.

The mean from predictors of QDA with of posterior and x are as follows:

```
> mean(prediction.value2$posterior)
[1] 0.3333333
>
```

. But here we see that there is no x variable in the predictors of QDA.

cBy adding the columns of individuals we get :

```
> column5=c(0.98)
> column6=c(122)
> column7=c(544)
> column8=c(186)
> column9=c(184)
>
> Diabetes.qda
Call:
qda(group ~ (glufast + glutest + instest + sspg + relwt), data = Diabetes,
    subset = train)

Prior probabilities of groups:
          Normal Chemical_Diabetic    Overt_Diabetic
       0.5241379         0.2482759         0.2275862

Group means:
                    glufast   glutest  instest     sspg     relwt
Normal             91.18421  349.9737 172.6447 114.0000 0.9372368
Chemical_Diabetic  99.30556  493.9444 288.0000 208.9722 1.0558333
Overt_Diabetic    217.66667 1043.7576 106.0000 318.8788 0.9839394
> individual <- c(0.98,122,544,186,184)
> Data = Diabetes[ ,which(names(Diabetes) %in% c("column5","column6","column7","column8","column9","column10"))]
> Diabetes
    relwt glufast glutest instest sspg            group
1    0.81      80     356     124   55           Normal
2    0.95      97     289     117   76           Normal
3    0.94     105     319     143  105           Normal
4    1.04      90     356     199  108           Normal
5    1.00      90     323     240  143           Normal
6    0.76      86     381     157  165           Normal
7    0.91     100     350     221  119           Normal
8    1.10      85     301     186  105           Normal
9    0.99      97     379     142   98           Normal
10   0.78      97     296     131   94           Normal
11   0.90      91     353     221   53           Normal
12   0.73      87     306     178   66           Normal
13   0.96      78     290     136  142           Normal
14   0.84      90     371     200   93           Normal
15   0.74      86     312     208   68           Normal
16   0.98      80     393     202  102           Normal
17   1.10      90     364     152   76           Normal
18   0.85      99     359     185   37           Normal
19   0.83      85     296     116   60           Normal
```

```
> pairs(Diabetes[1:5])
> cols <- character(nrow(Data))
> cols[]<-"black"
> cols[Data$v10 == 3] <- "blue"
> cols[Data$v10 == 2] <- "red"
> pairs(Diabetes[1:5],col=cols)
>
>
> data.frame =data.frame(column5,column6,column7,column8,column9)
>
> data.frame
  column5 column6 column7 column8 column9
1    0.98     122     544     186     184
```

So now that it is done we compare with lda function mentioned above and predict it with respect to the given data frame. Similarly it is done for QDA. Upon doing so we get the new individual classified into overt group w.r.t lda and normal group w.r.t to QDA