EXAM

1.a)

Code:

#Read the given CSV and assigns the data to variable data

data <- read.csv("BSE\_Sensex\_Index.csv", header = TRUE)

#prints out the dimensions and the head of the data frame

#dim(data)

#head(data)

data$Date <- as.Date(data$Date,format= "%m/%d/%Y")

diff\_Date = c()

for(i in 1:(length(data$Date)-1)){

diff <- difftime(data[i+1,1], data[i,1], units = "days")

diff\_Date <- append(diff\_Date,as.numeric(diff, units = "days"))

}

diff\_Date <- append(diff\_Date, 0)

#Gets the successive difference of the rows within a column. And for the last row in the column mean of the above 3 rows is added instead of null

diff\_Open <- diff(data$Open)

diff\_Open <- append(diff\_Open, mean(data[c(-5 : -2), 2]))

diff\_High <- diff(data$High)

diff\_High <- append(diff\_High, mean(data[c(-5 : -2), 3]))

diff\_Low <- diff(data$Low)

diff\_Low <- append(diff\_Low, mean(data[c(-5 : -2), 4]))

diff\_Close <- diff(data$Close)

diff\_Close <- append(diff\_Close, mean(data[c(-5 : -2), 5]))

diff\_Volume <- diff(data$Volume)

diff\_Volume <- append(diff\_Volume, mean(data[c(-5 : -2), 6]))

diff\_Adj.Close <- diff(data$Adj.Close)

diff\_Adj.Close <- append(diff\_Adj.Close, mean(data[c(-5 : -2), 7]))

data <- cbind(data, data.frame(diff\_Date, diff\_Open, diff\_High, diff\_Low, diff\_Close, diff\_Volume, diff\_Adj.Close))

#install the dpylr package

library(dplyr)

#Created a sample of 1000 observations with replacement

df1 <- sample\_n(data, 1000, replace = TRUE)

#Created a sample of 3000 observations with replacement

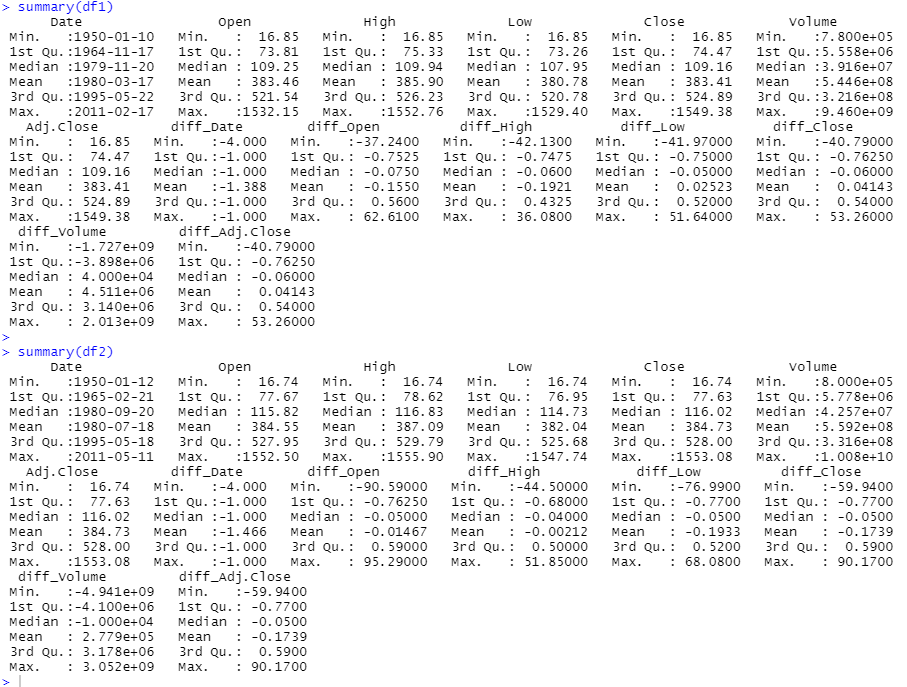
df2 <- sample\_n(data, 3000, replace = TRUE)

1.b)

Code:

summary(df1) → sample 1

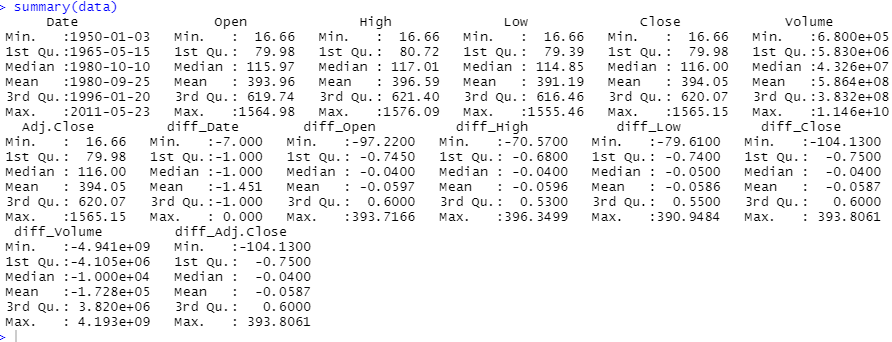
summary(df2) → sample 2



1.c)

Code:

summary(data) → for whole data



1.d)

Code:

boxplot(data$Open, data$Close, data$High, data$Low,

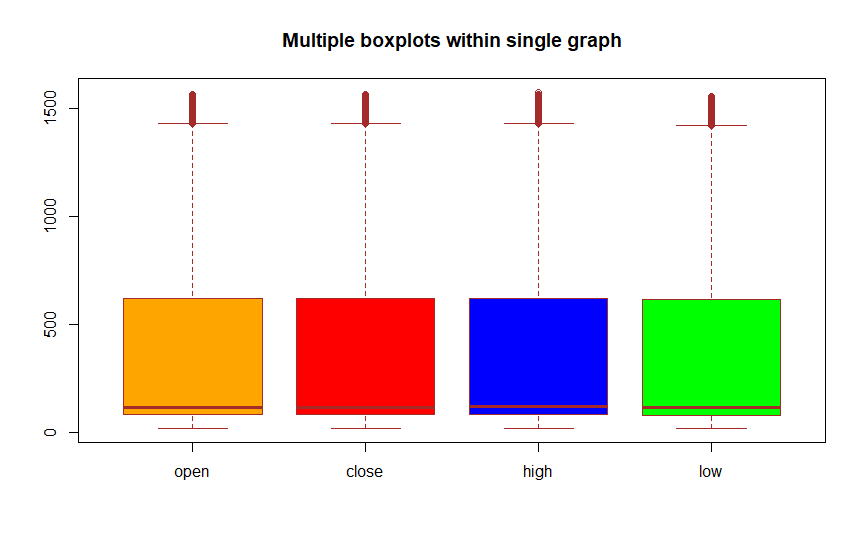
main = "Multiple boxplots within single graph",

names = c("open", "close", "high", "low"),

col = c("orange", "red", "blue", "green"),

border = "brown"

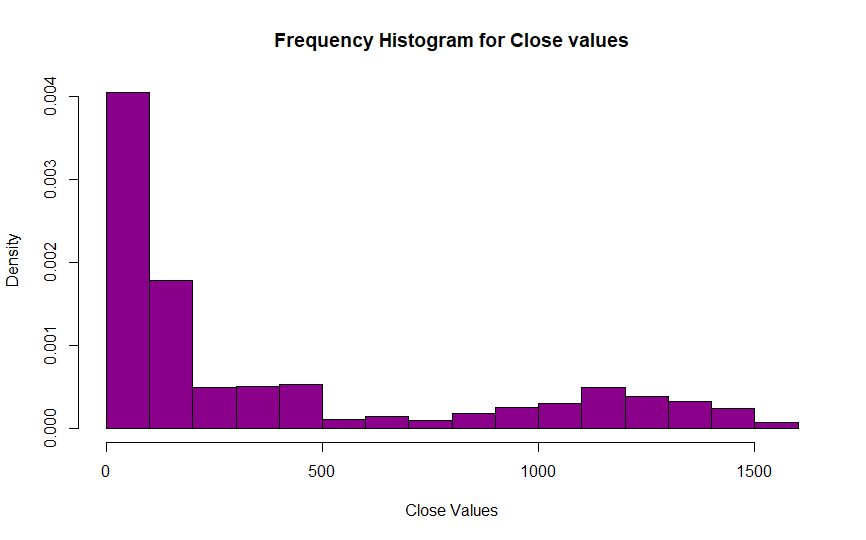
)

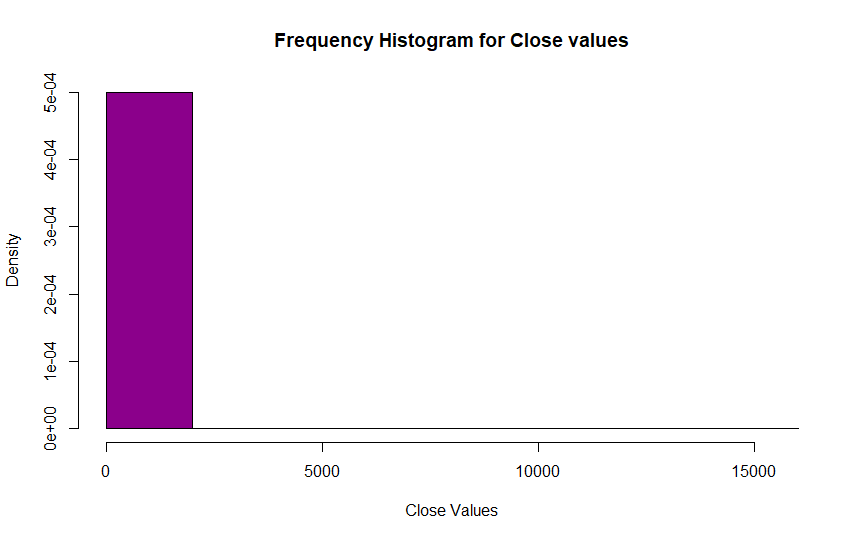


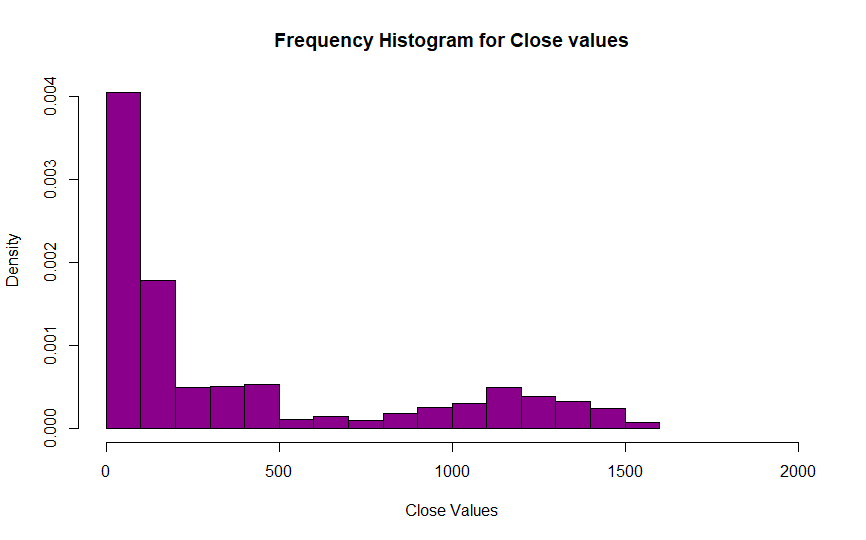
1.e)

Code:

hist(data$Close, main="Frequency Histogram for Close values", xlab="Close Values", xlim=c(0,16000),breaks = seq(from = 0,to = 16000,by = 2000), col="darkmagenta", freq=FALSE)







-------------------------------------------------------------------------------------------------------------------------------

2.a)

Code:

library(arules)

library(arulesViz)

library(RColorBrewer)

#The read.transactions() function read the file csv file and convert it to a transaction format

#Parameters: Transaction file: name of the csv file

#rm.duplicates : to make sure that we have no duplicate transaction entried

#format : basket (row 1: transaction ids, row 2: list of items)

#sep: separator between items, in this case commas

#cols : column number of transaction IDs

txn = read.transactions(file="ItemList.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

#getting rid of unnecessary quotes in transactions

txn@itemInfo$labels <- gsub("\"","",txn@itemInfo$labels)

#running the apriori algorithm

basket\_rules <- apriori(txn, parameter = list(sup = 0.3,target="rules"))

inspect(basket\_rules)

#Alternative to inspect() is to convert rules to a dataframe and then use View()

df\_basket <- as(basket\_rules,"data.frame")

View(df\_basket)

library(arulesViz)

plot(basket\_rules)

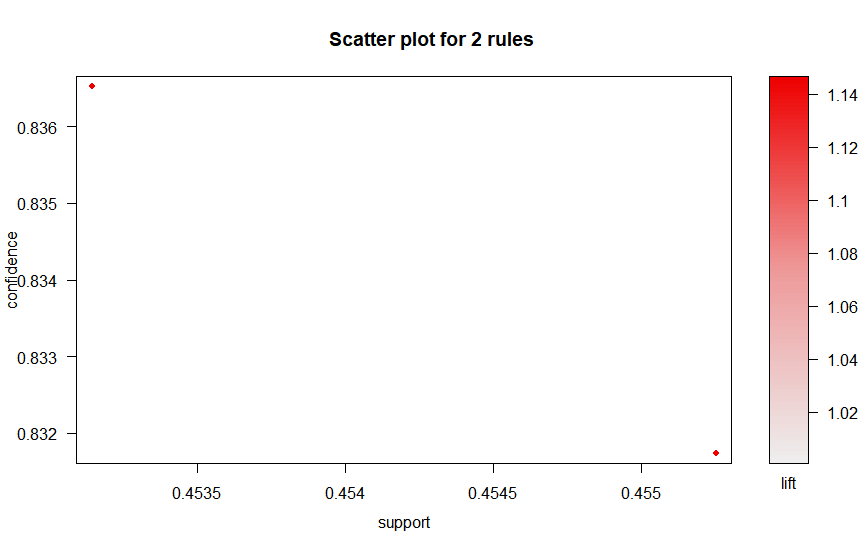
plot(basket\_rules, method = "grouped", control = list(k = 5))

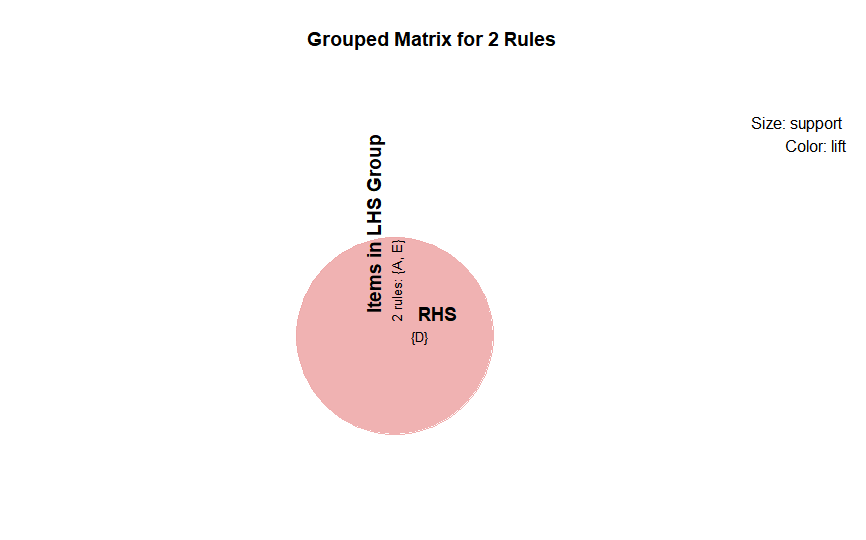
plot(basket\_rules, method="graph", control=list(type="items"))

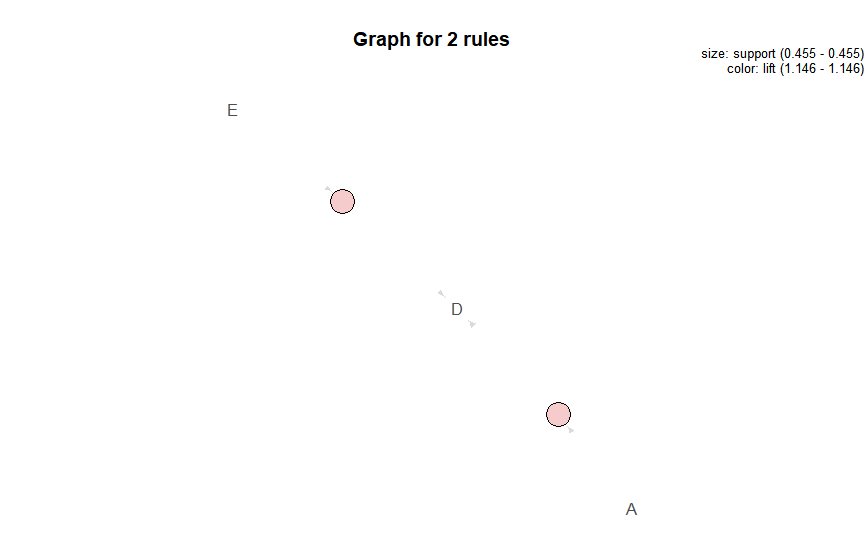
plot(basket\_rules, method="paracoord", control=list(alpha=.5, reorder=TRUE))

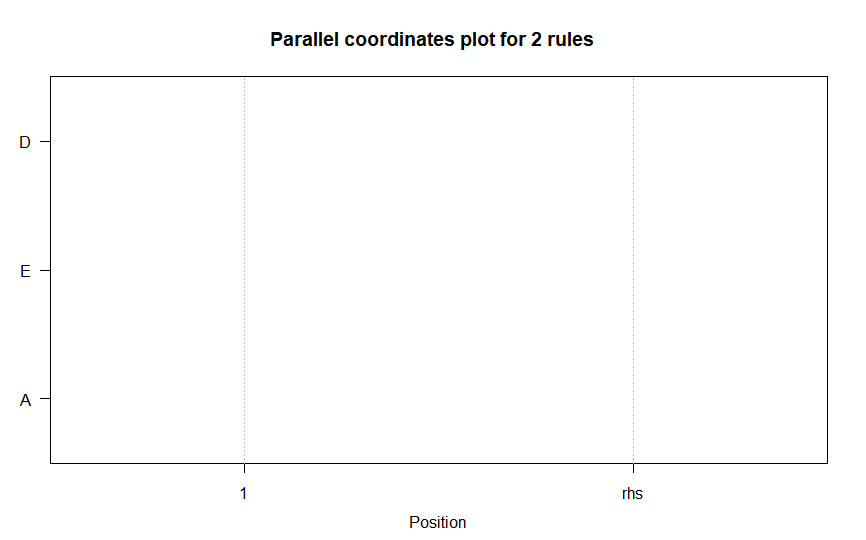
plot(basket\_rules,measure=c("support","lift"),shading="confidence",interactive=T,colour=green)

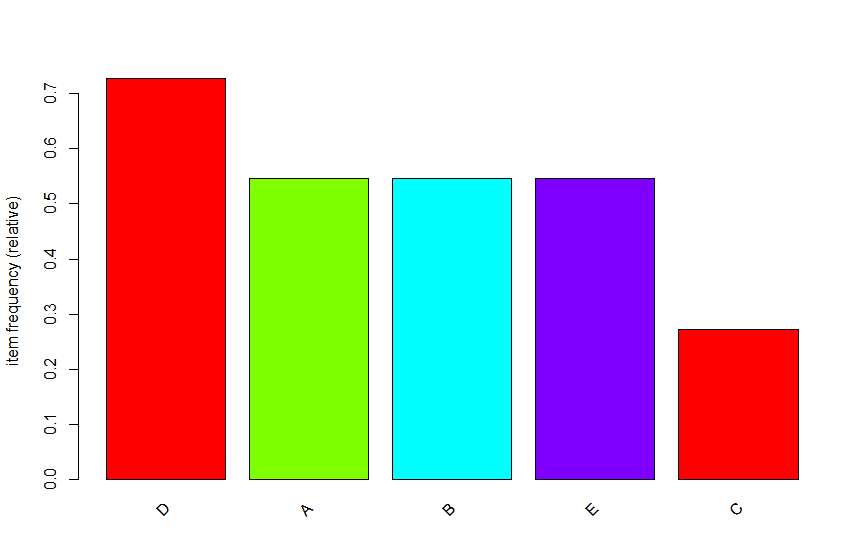
itemFrequencyPlot(txn, topN = 5, col = rainbow(4))











3)

Code:

install.packages("caret")

library(caret)

library(rpart.plot)

lens = read.csv("lenses.data.csv", header = FALSE, col.names = c("1", "2", "3", "4", "5", "Label"))

str(lens)

summary(lens)

x = lens[,1:5]

y = as.factor(lens$Label)

model = rpart(y~.,x,control=rpart.control(minsplit=0,minbucket=0,cp=-1, maxcompete=0, maxsurrogate=0, usesurrogate=0, xval=0,maxdepth=5))

plot(model)

text(model)

rpart.plot(model)

#Information Gain

sum(y==predict(model,x,type="class"))/length(y)

#mis classification error

1-sum(y==predict(model,x,type="class"))/length(y)

model1 = rpart(y~.,x,control=rpart.control(minsplit=0,minbucket=0,cp=-1, maxcompete=0, maxsurrogate=0, usesurrogate=0, xval=0,maxdepth=7))

plot(model1)

text(model1)

rpart.plot(model1)

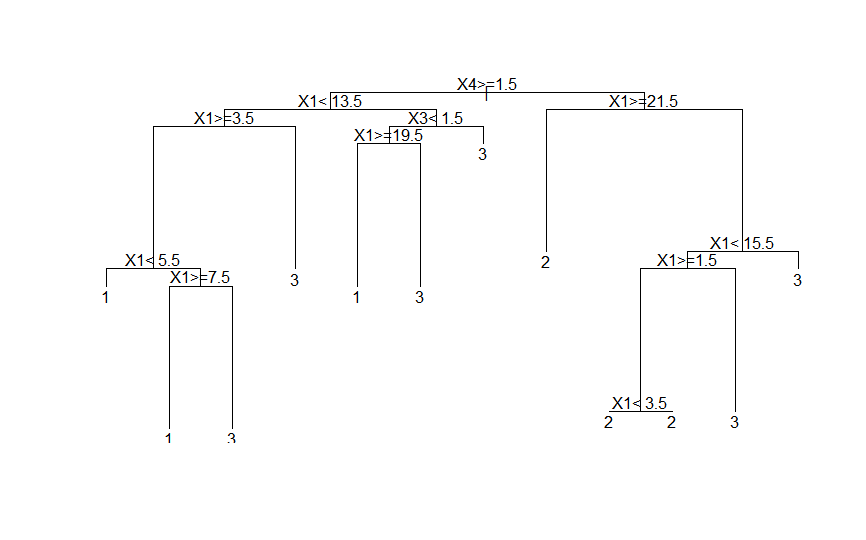
#Information Gain

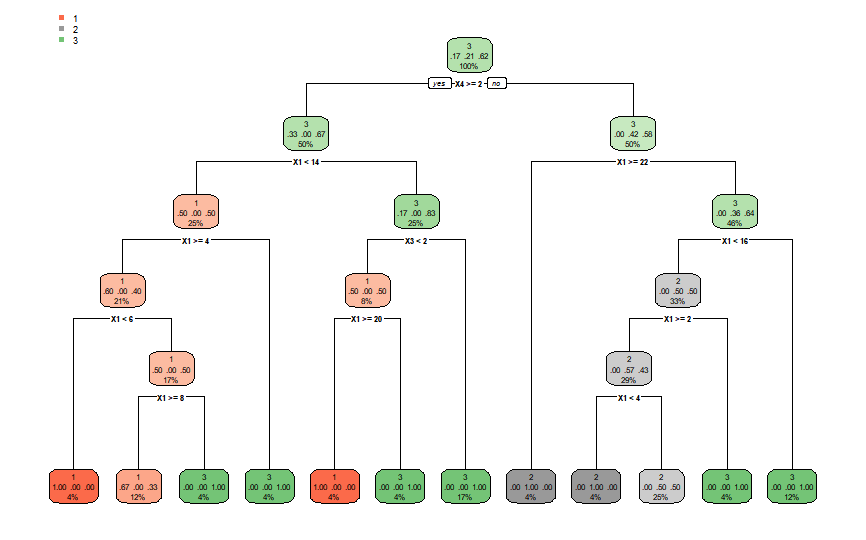
sum(y==predict(model1,x,type="class"))/length(y)

#mis classification error

1-sum(y==predict(model1,x,type="class"))/length(y)

#Result: If tree depth is increased, the mis classification error has decreased and information gain increased.





4)

Code:

liver = read.csv("Liver\_data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sgpt", "sgot", "gammagt", "drinks","selector"))

liver$selector <- as.factor(liver$selector)

#The cut function is often quite useful for making discrete variables from continuous or numerical ones, sometimes in combination with the function quantile.

#we can group some of the data together.

liver$drinks <- cut(liver$drinks, breaks = c(0,5,10,15,20), labels = c('C1', 'C2', "C3", 'C4'), right = FALSE)

#for omiting null variable

liver <- na.omit(liver)

#subset is for taking a small subset of the given data upon a condition

#contains all the 7 columns

train = subset(liver, liver$selector == 1)

test = subset(liver, liver$selector == 2)

#contains 5 columns. Removed selector and drinks.

#Selector is used for spliting test and train data. drinks is the categorical output which we have to find

x\_train <- subset(train, select = -c(selector, drinks))

x\_test <- subset(test, select = -c(selector, drinks))

library(class)

#contains only drinks

y\_train = train[,6, drop = TRUE]

y\_test = test[,6, drop = TRUE]

#k = 1

fit1 = knn(x\_train,x\_train,y\_train,k=1)

1-sum(y\_train==fit1)/length(y\_train)

#Misclassification error: 0

1-sum(y\_test==fit1)/length(y\_test)

# Misclassification error : 0.515

#K = 2

fit2 = knn(x\_train,x\_train,y\_train,k=2)

1-sum(y\_train==fit2)/length(y\_train)

#Misclassification error: 0.1468531

1-sum(y\_test==fit2)/length(y\_test)

#Misclassification error: 0.46

#k = 3

fit3 = knn(x\_train,x\_train,y\_train,k=3)

1-sum(y\_train==fit3)/length(y\_train)

#Misclassification error: 0.1818182

1-sum(y\_test==fit3)/length(y\_test)

#Misclassification error: 0.47

5)

Code:

liver = read.csv("Liver\_data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sgpt", "sgot", "gammagt", "drinks","selector"))

liver$selector <- as.factor(liver$selector)

liver$drinks <- cut(liver$drinks, breaks = c(0, 5,10,15,20), labels = c('C1', 'C2', "C3", 'C4'), right = FALSE)

liver <- na.omit(liver)

train = subset(liver, liver$selector == 1)

test = subset(liver, liver$selector == 2)

x\_train <- subset(train, select = -c(selector, drinks))

x\_test <- subset(test, select = -c(selector, drinks))

library(class)

y\_train = train[,6, drop = TRUE]

y\_test = test[,6, drop = TRUE]

library(e1071)

#SVM

#For training

fit = svm(x\_train, y\_train)

1-sum(y\_train==predict(fit,x\_train))/length(y\_train)

#Misclassification error: 0.2027

#For test data

fit = svm(x\_test, y\_test)

1-sum(y\_test==predict(fit,x\_test))/length(y\_test)

#Misclassification error: 0.265

#Result: The error for test data is less in svm than knn

6)

Code:

data = read.csv("Liver\_data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sgpt", "sgot", "gammagt", "drinks","selector"))

x = data[,1:2]

plot(x,pch=19,xlab=expression(x[1]),ylab=expression(x[2]))

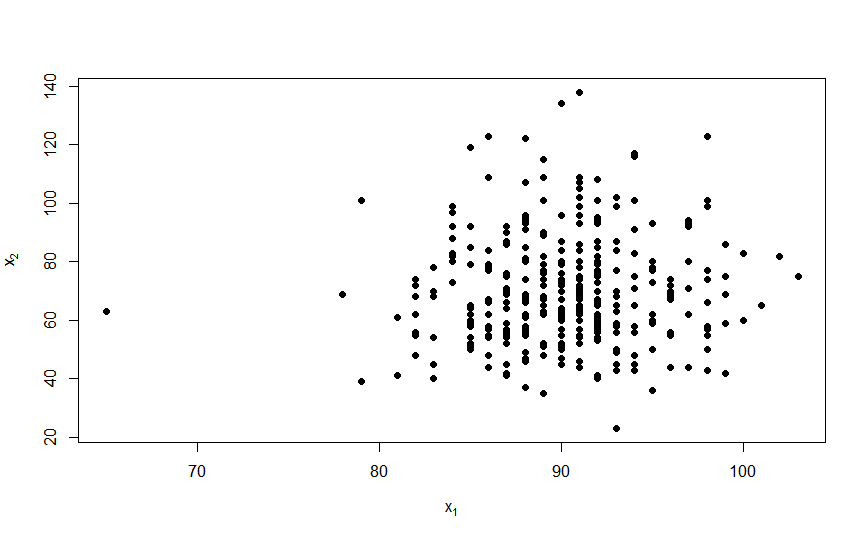
fit<-kmeans(x, 4)

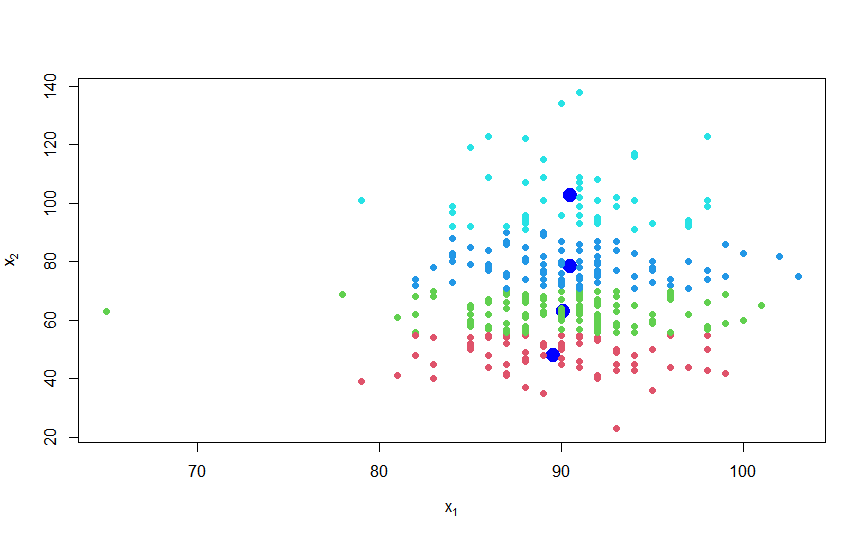
points(fit$centers,pch=19,col="blue",cex=2)

library(class)

knnfit = knn(fit$centers,x,as.factor(c(-2,-1,1,2)))

points(x,col=1+1\*as.numeric(knnfit),pch=19)





7)

Code:

data = read.csv("Liver\_data.csv", header = FALSE, col.names = c("mcv", "alkphos", "sgpt", "sgot", "gammagt", "drinks","selector"))

x = data[,1:5]

y = data[,6]

fit = kmeans(x,4)

library(class)

knnfit = knn(fit$centers,x,as.factor(c(-2,-1,1,2)))

error = 1-sum(knnfit == y)/length(y)

print(error)

#Result: Misclassification Error Rate = 0.9304348

8)

Code:

sensexdata = read.csv("BSE\_Sensex\_Index.csv", header = TRUE)

#In R programming, the mutate function is used to create a new variable from a data set.

#In order to use the function, we need to install the dplyr package, which is an add-on to R that

#includes a host of cool functions for selecting, filtering, grouping, and arranging data.

#getting growth\_rate column

library(dplyr)

new\_sensexdata = mutate(sensexdata, growth\_rate = lead((lag(Close) - Close) / Close))

#replacing last row in growth\_rate column with the mean of the above there rows in the column

lastcol = nrow(new\_sensexdata)

new\_sensexdata$growth\_rate[lastcol] = mean(new\_sensexdata$growth\_rate[c((lastcol-3) : (lastcol-1))])

#calculating z-scores

growth\_rate\_mean <- mean(new\_sensexdata$growth\_rate, na.rm=TRUE)

growth\_rate\_sd <- sd(new\_sensexdata$growth\_rate,na.rm=TRUE)

z<-(new\_sensexdata[,8] - growth\_rate\_mean) / growth\_rate\_sd

sort(z)

new\_sensexdata$zscores <- z

#Dates of the outliers

dates <- subset(new\_sensexdata[,1], new\_sensexdata[,"zscores"] >= 3.0 | new\_sensexdata[,"zscores"] <= -3.0)

View(dates)

Dates:

