Solutions

a) mydata<-read.csv("BSE\_Sensex\_Index.csv", header=FALSE)

sample\_1000 <- sample(seq(1, length(mydata[,1])), 1000, replace = T)

sample\_3000 <- sample(seq(1, length(mydata[,1])), 3000, replace = T)

b) mean(sample\_1000) 7771.474 mean(sample\_3000) 7858.797

max(sample\_1000) 15424 max(sample\_3000) 15445

var(sample\_1000) 20185607 var(sample\_3000) 20051208

quantile(sample\_1000,.25) 3894 quantile(sample\_3000,.25) 4048.75

c) mean(mydata$V2, na.rm = TRUE) 586421999

Mean for entire data is exponentially higher than samples of 1000 and 3000. Hence the mean of the data increased when we consider the entire sensex trends.

d) boxplot(open\_points, high\_ponts, low\_ponts, close\_ponts,

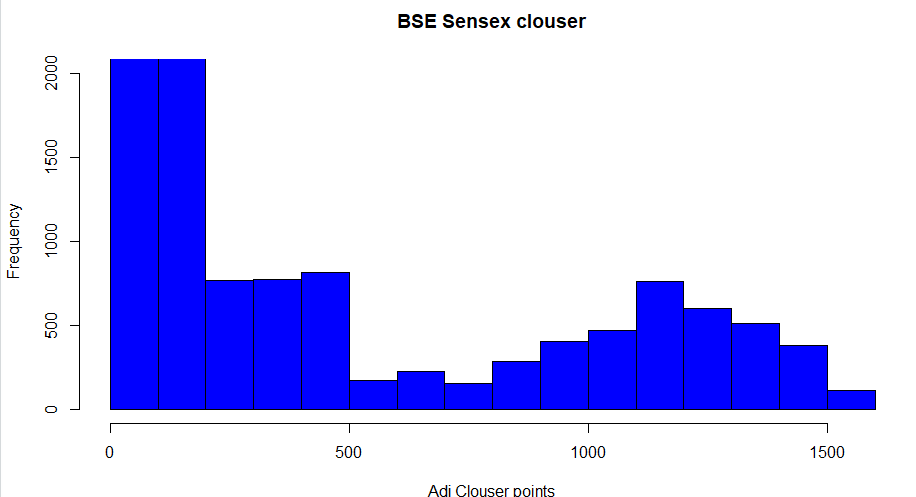
main = "BoxPlot for the values",

names = c("Open", "High", "Low", "Close"),

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

horizontal = TRUE, notch = TRUE)

e) hist(mydata$V7, ylim = c(0,2000), col="blue",main = "BSE Sensex clouser", xlab="Adj Clouser points")



2.

**Code :**

data = read.csv("apriori\_data.csv", header = TRUE);

View(data)

data$TID <- NULL

library(arules)

write.csv(data, "ItemList.csv", quote = FALSE, row.names = TRUE)

transactions = read.transactions("ItemList.csv", sep=',', rm.duplicates = TRUE)

inspect(transactions)

frequent\_itemsets <- apriori(transactions, parameter = list(sup = 0.03, conf = 0.5,target="frequent itemsets"))

inspect(sort(frequent\_itemsets)[1:15])

itemFrequencyPlot(transactions, topN = 5)

**Methods :**

Apriori Algorithm:

Given data set is as follows:

|  |  |
| --- | --- |
| id | Values |
| 101 | A,B,C,D,E |
| 102 | A,C,D |
| 103 | D,E |
| 104 | B,C,E |
| 105 | A,B,D,E |
| 106 | A,B |
| 107 | B,D,E |
| 108 | A,B,D |
| 109 | A,D |
| 110 | D,E |

MIN SUPPORT = 3

THRESHOLD – 30 %

Step – 1 :

|  |  |
| --- | --- |
| A | 6 |
| B | 6 |
| C | 3 |
| D | 8 |
| E | 6 |

As all the values are greater than or equal to the minimum support all are considered.

Step – 2:

|  |  |
| --- | --- |
| A,B | 4 |
| A,C | 2 |
| A,D | 5 |
| A,E | 2 |
| B,C | 2 |
| B,D | 4 |
| B,E | 4 |
| C,D | 2 |
| C,E | 2 |
| D,E | 5 |

As the min support = 3 we need to remove which are less than 3.

|  |  |
| --- | --- |
| A,B | 4 |
| A,D | 5 |
| B,D | 4 |
| B,E | 4 |
| D,E | 5 |

Step – 3 :

|  |  |
| --- | --- |
| A,B,D | 3 |
| B,D,E | 3 |

Step – 4 :

1. {A,B,D} – First b) {B,D,E} – Second

{A, B} => {D} { B,D} => {E}

Confidence = support {A,B,D} / support {A, B} = (3/ 4)\* 100 = 75% Confidence = support {B,D,E} / support {B,D} = (3/ 4)\* 100 = 75%

{A, D} => {B} {B,E} => {D}

Confidence = support {A,B,D} / support {A, D} = (3/ 5)\* 100 = 60% Confidence = support {B,D,E} / support {B, E} = (3/ 4)\* 100 = 75%

{B, D} => {A} {D,E} => {B}

Confidence = support {A,B,D } / support {B,D} = (3/ 4)\* 100 = 75% Confidence = support {B,D,E} / support {D,E} = (3/ 5)\* 100 = 60%

{A} => {B, D} {B} => {D,E}

Confidence = support {A, D, B} / support {A} = (3/ 6)\* 100 = 50% Confidence = support {B,D,E} / support {B} = (3/ 6)\* 100 = 50%

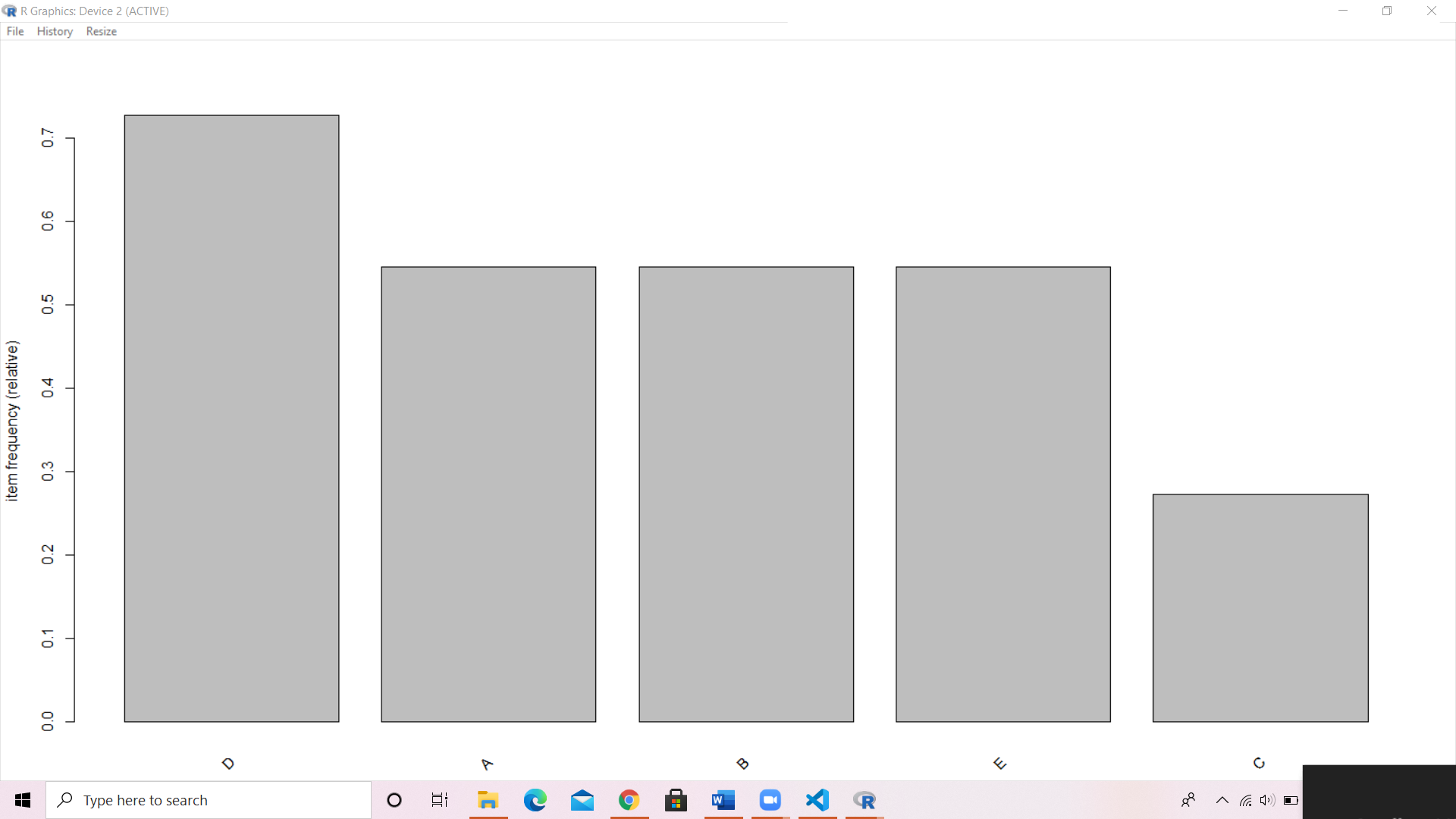
{B} => {A, D} {D} => {B,E}

Confidence = support {A, B, D} / support {B} = (3/ 6)\* 100 = 50% Confidence = support {B,D,E} / support {D} = (3/ 8)\* 100 = 37.5%

{D} => {A, B} {E} => {B,D}

Confidence = support {A,B,D} / support {D} = (3/ 8)\* 100 = 37.5% Confidence = support {B,D,E} / support {E} = (3/ 6)\* 100 = 50%

all the above association rules are strong if minimum confidence threshold is 30%.



3. lensdata = read.csv("lenses.data.csv", header = FALSE, col.names = c("index", "age", "spectacle\_prescription", "astigmatic", "tear\_production\_rate", "Class"))

lensdata$index <- NULL

library(rpart)

y<-as.factor(lensdata[,5])

x<-lensdata[,1:4]

model1<-rpart(y~.,x, parms = list(split = 'information'),control=rpart.control(minsplit=0,minbucket=0,cp=-1, maxcompete=0, maxsurrogate=0, usesurrogate=0, xval=0,maxdepth=5))

library(rpart.plot)

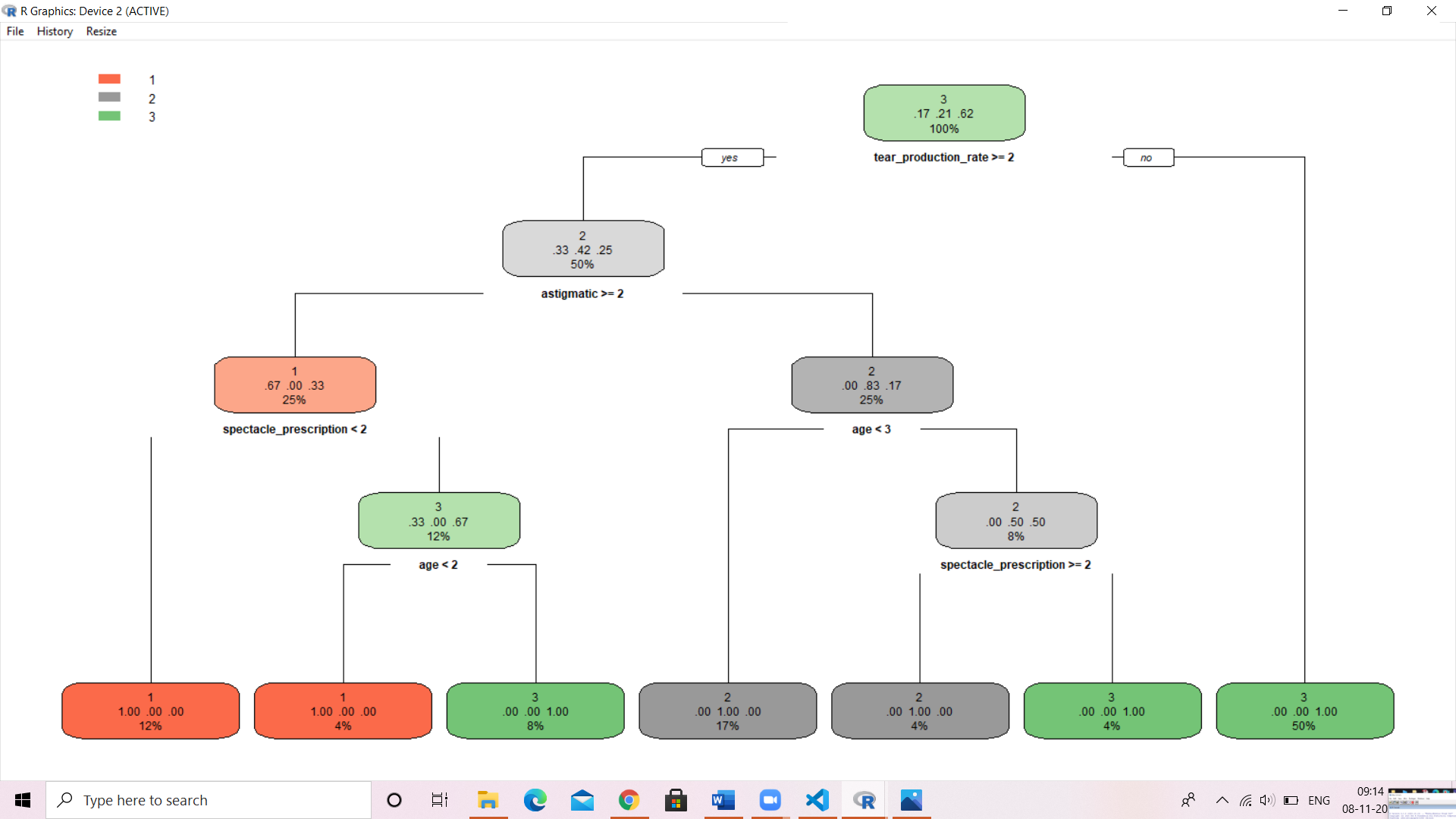
rpart.plot(model1)

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

gain = 1

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

error = 0



8. bse\_sensex <- read.csv("BSE\_Sensex\_Index.csv")

SGR\_Close <- c()

for (i in 1:15446){

SGR\_Close[i] <- (bse\_sensex$Close[i] - bse\_sensex$Close[i+1]) / bse\_sensex$Close[i+1]

}

SGR\_Close[15447] <- (SGR\_Close[15446] + SGR\_Close[15445] + SGR\_Close[15444]) / 3

SGR\_Close[15447]

x\_SGR\_Close <- c()

mean\_SGR <- mean(SGR\_Close)

mean\_SGR = 0.0003303709

sd\_SGR <- sd(SGR\_Close)

sd\_SGR = 0.009669

for (j in 1 : 15447){

x\_SGR\_Close[j] <- (SGR\_Close[j] - mean\_SGR) / sd\_SGR

}

x\_SGR\_Close

outliners\_dates <- c()

outliners\_count <- 0

otda <- 1

for(k in 1:15447){

+ if(x\_SGR\_Close[k] > 3){

+ outliners\_count <- outliners\_count+1

+ outliners\_dates[otda] <- bse\_sensex$Date[k]

+ otda <- otda+1

+ }

+ if(x\_SGR\_Close[k] < -3 ){

+ outliners\_count <- outliners\_count+1

+ outliners\_dates[otda] <- bse\_sensex$Date[k]

+ otda <- otda+1

}

outliners\_count -- 216

outliners\_dates