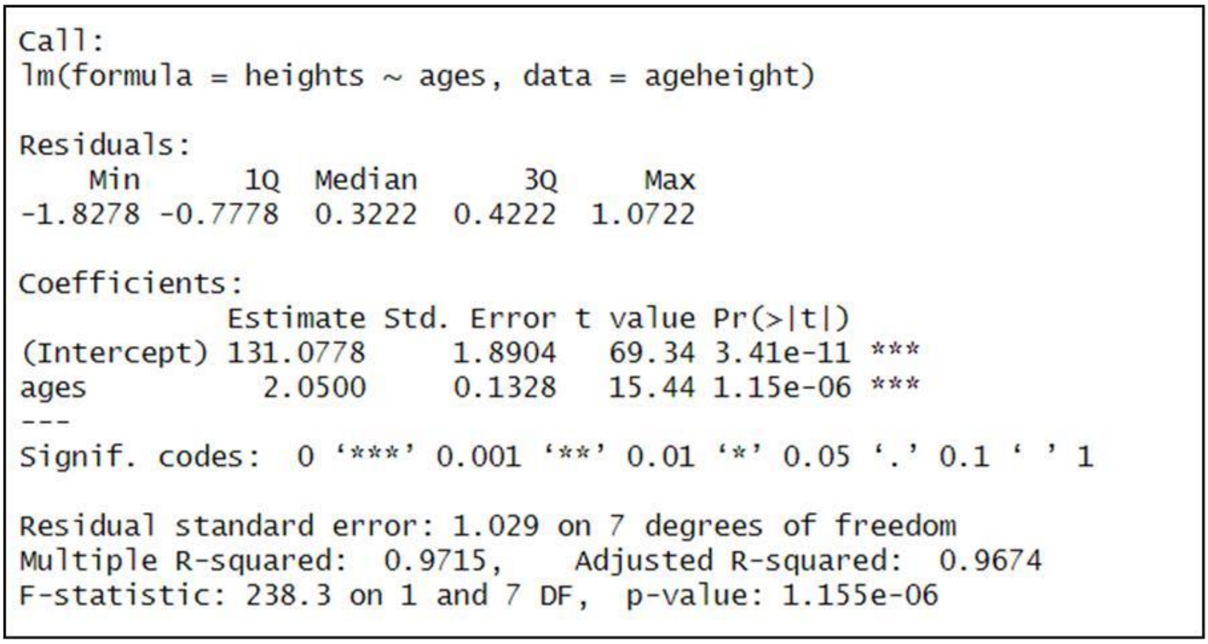
• Q1)Implementation of Linear Regression in R

```
install.packages("xlsx")

library("xlsx")

ageheight <- read.xlsx("C:\\Harshal\\Documents\\MCA\\ADBMS\\Data\\ageandheight.xls",
sheetName = "linear regression")

result <- lm(heights~ages, data=ageheight)

summary(result)
```

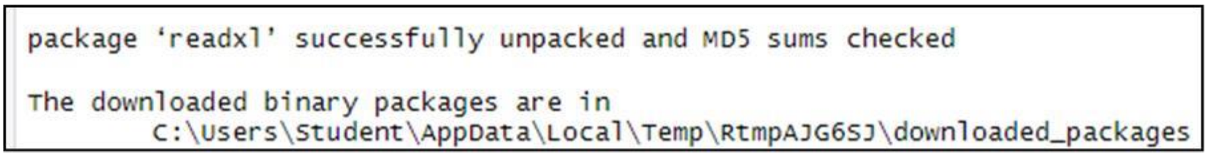


```
Call:
lm(formula = heights ~ ages, data = ageheight)

Residuals:
    Min      1Q  Median      3Q     Max
-1.8278 -0.7778  0.3222  0.4222  1.0722

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 131.0778     1.8904   69.34 3.41e-11 ***
ages          2.0500     0.1328   15.44 1.15e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.029 on 7 degrees of freedom
Multiple R-squared:  0.9715,    Adjusted R-squared:  0.9674
F-statistic: 238.3 on 1 and 7 DF,  p-value: 1.155e-06
```

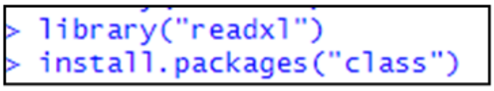
Q2. Implementation and analysis of Classification algorithms: Naive Bayesian, K-Nearest Neighbor, ID3, C4.5

Implementation and Analysis of Classification Algorithm : Naïve Bayesian Analysis

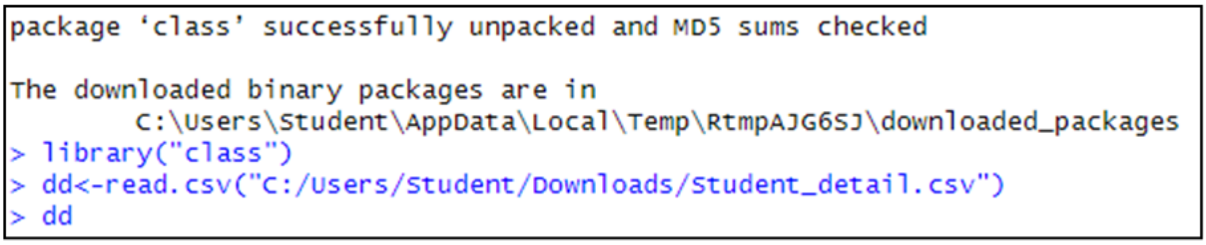
```
install.packages("readxl")
library("readxl") install.packages("class")
library("class")
dd<-read.csv("C:/Users/Student/Downloads/Student_detail.csv") dd
dd1<-read_excel("C:/Users/Student/Downloads/Output.xls") dd1
dd2<-read.table("C:/Users/Student/Downloads/Student_detail.csv",header=TRUE,sep=",")
dd2
```



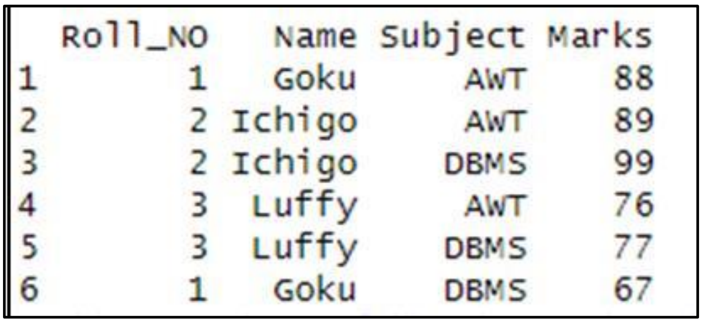
```
package 'readxl' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Student\AppData\Local\Temp\RtmpAJG6SJ\downloaded_packages
```



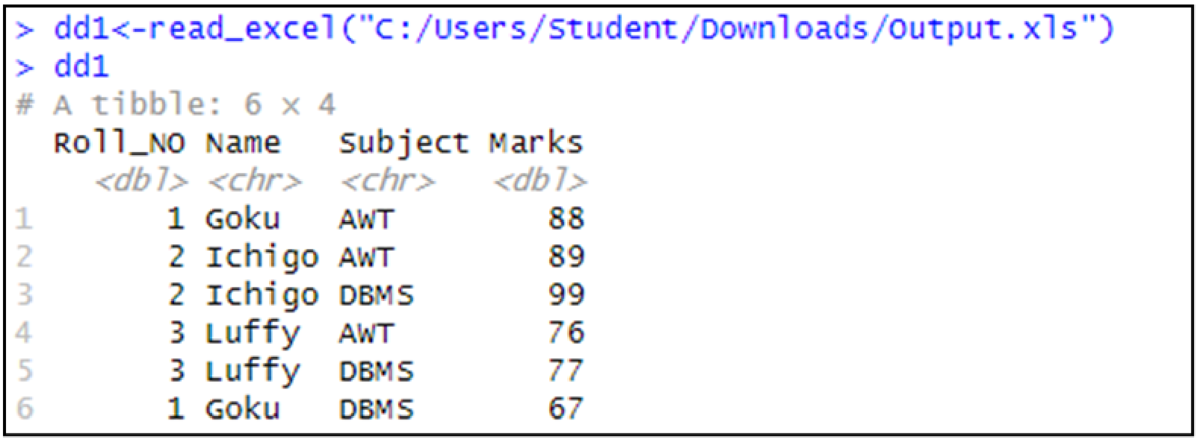
```
> library("readxl")
> install.packages("class")
```



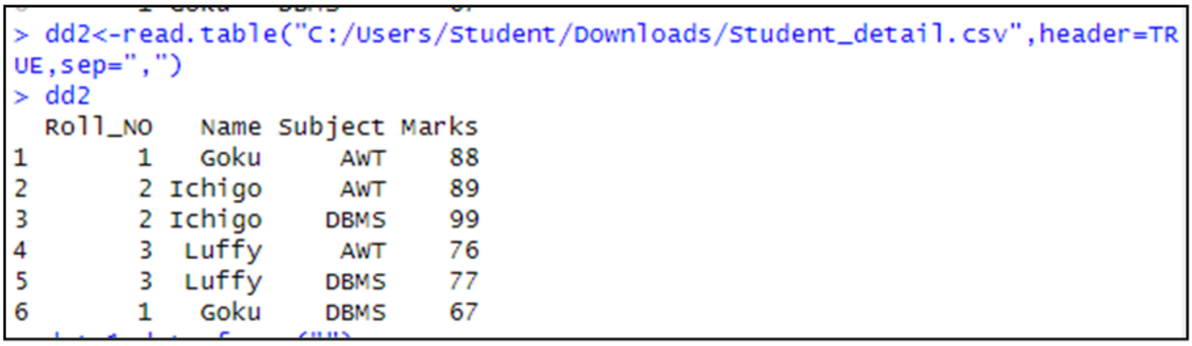
```
package 'class' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Student\AppData\Local\Temp\RtmpAJG6SJ\downloaded_packages
> library("class")
> dd<-read.csv("C:/Users/Student/Downloads/Student_detail.csv")
> dd
```



```
  Roll_NO   Name Subject Marks
1       1   Goku     AWT    88
2       2 Ichigo     AWT    89
3       2 Ichigo    DBMS    99
4       3  Luffy     AWT    76
5       3  Luffy    DBMS    77
6       1   Goku    DBMS    67
```



```
> dd1<-read_excel("C:/Users/Student/Downloads/output.xls")
> dd1
# A tibble: 6 × 4
  Roll_NO Name   Subject Marks
    <dbl> <chr>  <chr>   <dbl>
1       1 Goku   AWT        88
2       2 Ichigo AWT        89
3       2 Ichigo DBMS       99
4       3 Luffy  AWT        76
5       3 Luffy  DBMS       77
6       1 Goku   DBMS       67
```



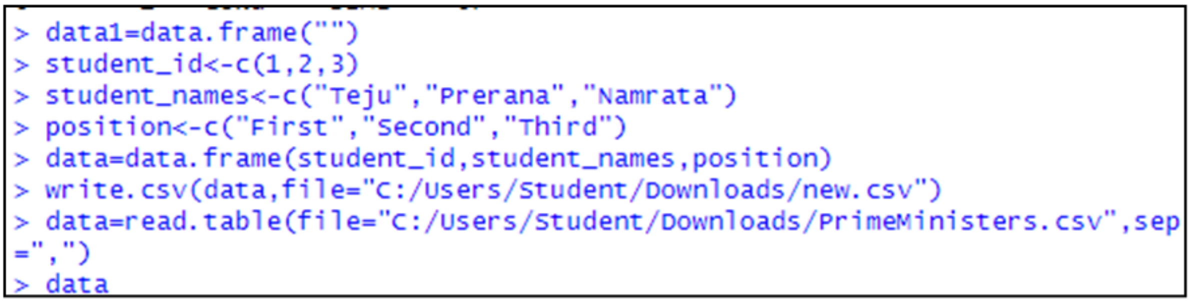
```
> dd2<-read.table("C:/Users/Student/Downloads/Student_detail.csv",header=TR
UE,sep=",")
> dd2
  Roll_NO   Name Subject Marks
1       1   Goku     AWT    88
2       2 Ichigo     AWT    89
3       2 Ichigo    DBMS    99
4       3  Luffy     AWT    76
5       3  Luffy    DBMS    77
6       1   Goku    DBMS    67
```

data1=data.frame("")

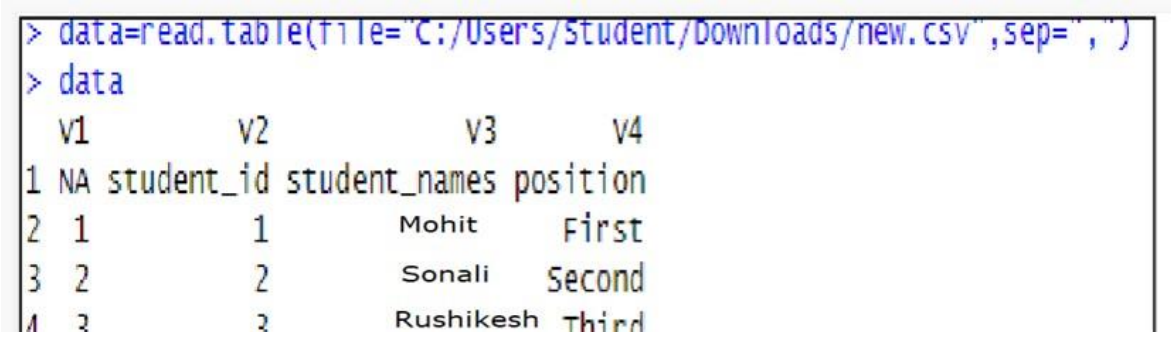
student_id<-c(1,2,3)
student_names<-c("Teju","Prerana","Namrata") position<-c("First","Second","Third")
data=data.frame(student_id,student_names,position)

write.csv(data,file="C:/Users/Student/Downloads/new.csv")

data=read.table(file="C:/Users/Student/Downloads/new.csv",sep=",") data



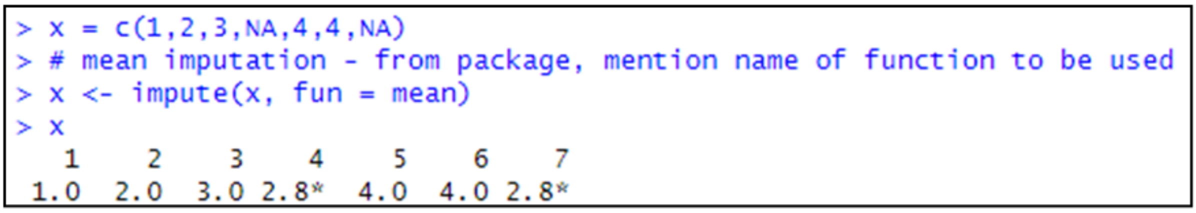
```
> data1=data.frame("")
> student_id<-c(1,2,3)
> student_names<-c("Teju","Prerana","Namrata")
> position<-c("First","Second","Third")
> data=data.frame(student_id,student_names,position)
> write.csv(data,file="C:/Users/Student/Downloads/new.csv")
> data=read.table(file="C:/Users/Student/Downloads/PrimeMinisters.csv",sep
=",")
> data
```



```
> data=read.table(file="C:/Users/Student/Downloads/new.csv",sep=",")
> data
  V1         V2            V3       V4
1 NA student_id student_names position
2  1          1         Mohit    First
3  2          2        Sonali   Second
4  3          3     Rushikesh    Third
```

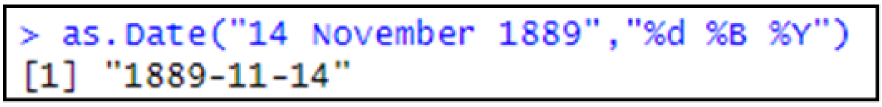
install.packages("Hmisc") library(Hmisc)

x = c(1,2,3,NA,4,4,NA)
# mean imputation - from package, mention name of function to be used
x <- impute(x, fun = mean) x



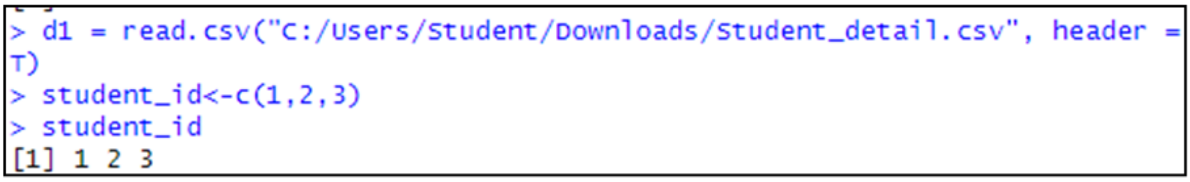
```
> x = c(1,2,3,NA,4,4,NA)
> # mean imputation - from package, mention name of function to be used
> x <- impute(x, fun = mean)
> x
   1    2    3    4    5    6    7 
 1.0  2.0  3.0 2.8*  4.0  4.0 2.8* 
```

as.Date("14 November 1889","%d %B %Y")

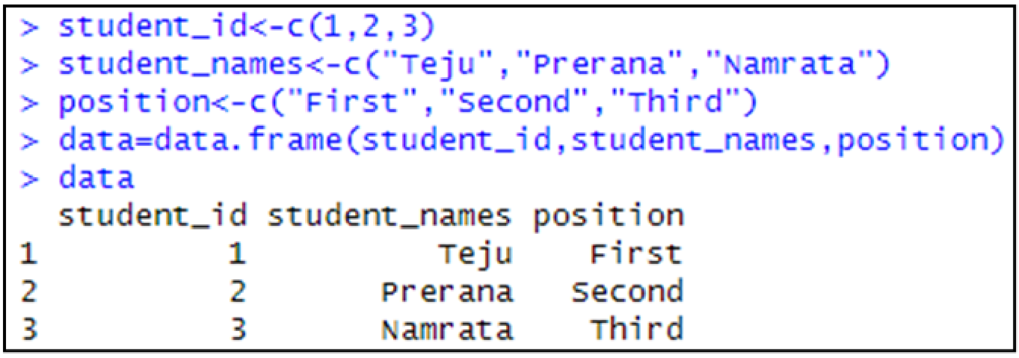


```
> as.Date("14 November 1889","%d %B %Y")
[1] "1889-11-14"
```

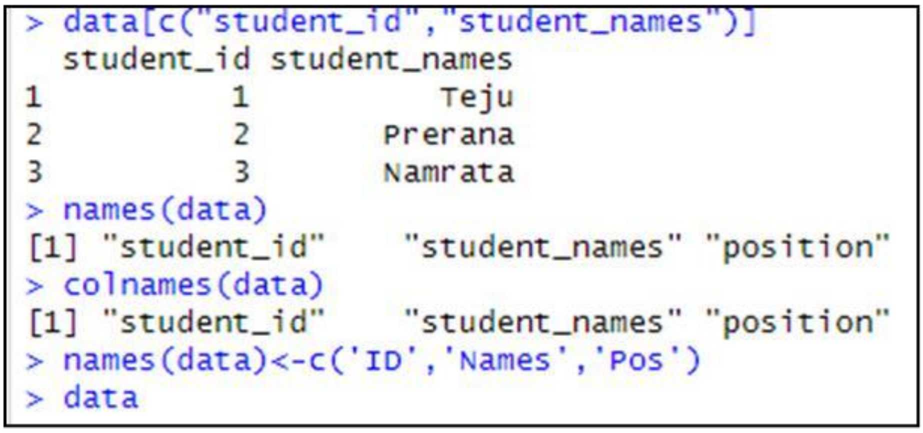
d1 = read.csv("C:/Users/Student/Downloads/Student_detail.csv", header = T) student_id<-c(1,2,3)
student_names<-c("Teju","Prerana","Namrata") position<-c("First","Second","Third")
data=data.frame(student_id,student_names,position) data
data[c("student_id","student_names")]
names(data) colnames(data)
names(data)<-c('ID','Names','Pos')
data nrow(data) ncol(data)



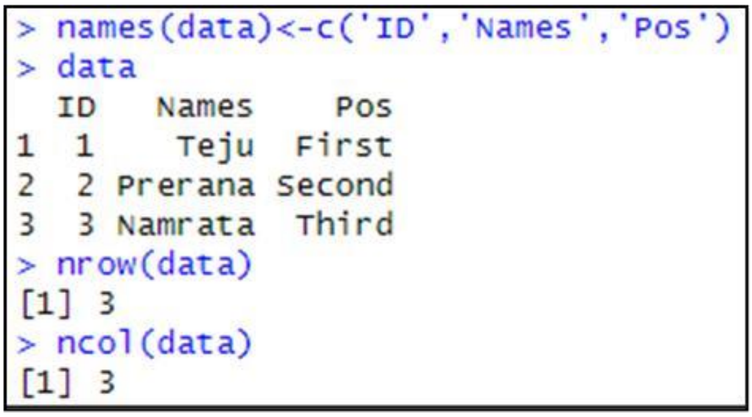
```
> d1 = read.csv("C:/Users/Student/Downloads/Student_detail.csv", header =
T)
> student_id<-c(1,2,3)
> student_id
[1] 1 2 3
```



```
> student_id<-c(1,2,3)
> student_names<-c("Teju","Prerana","Namrata")
> position<-c("First","Second","Third")
> data=data.frame(student_id,student_names,position)
> data
  student_id student_names position
1          1          Teju    First
2          2       Prerana   Second
3          3       Namrata    Third
```

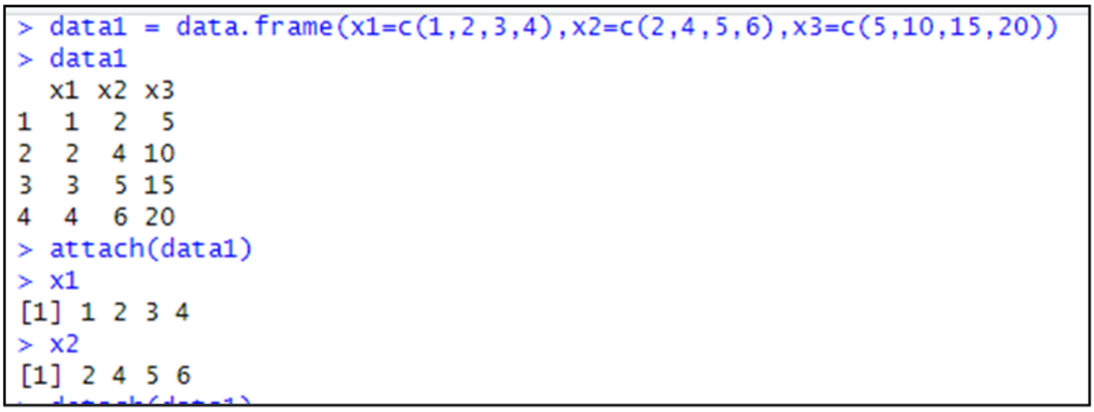


```
> data[c("student_id","student_names")]
  student_id student_names
1          1          Teju
2          2       Prerana
3          3       Namrata
> names(data)
[1] "student_id"    "student_names" "position"
> colnames(data)
[1] "student_id"    "student_names" "position"
> names(data)<-c('ID','Names','Pos')
> data
```



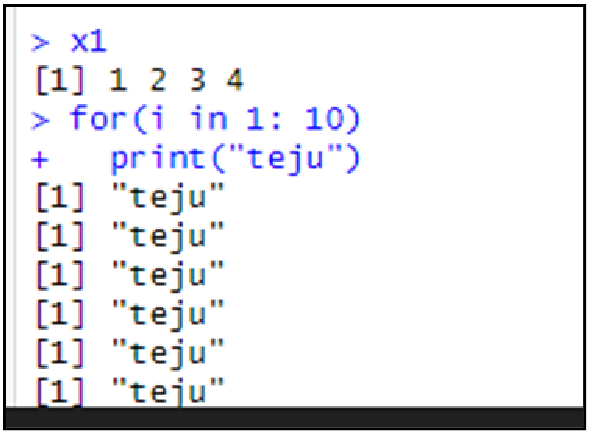
```
> names(data)<-c('ID','Names','Pos')
> data
  ID   Names    Pos
1  1    Teju  First
2  2 Prerana Second
3  3 Namrata  Third
> nrow(data)
[1] 3
> ncol(data)
[1] 3
```

data1 = data.frame(x1=c(1,2,3,4),x2=c(2,4,5,6),x3=c(5,10,15,20))
data1 attach(data1) x1
x2

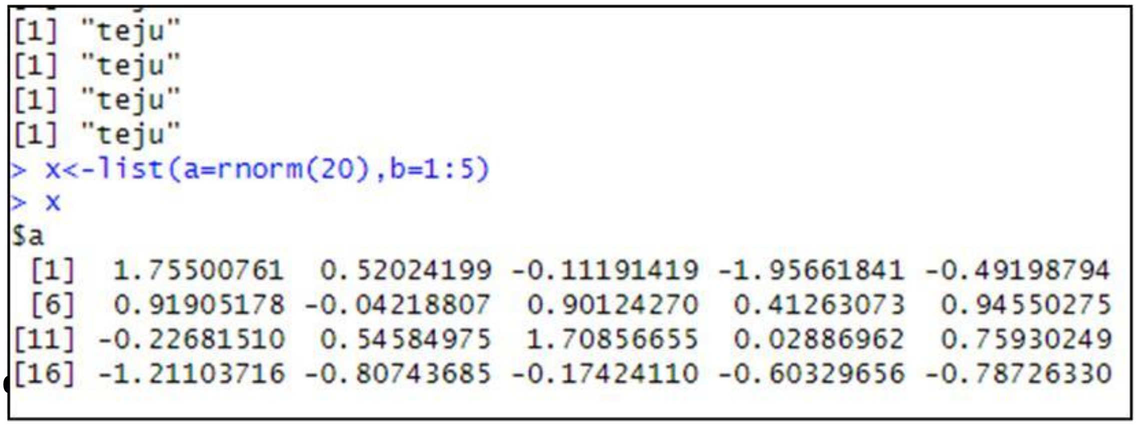


```
> data1 = data.frame(x1=c(1,2,3,4),x2=c(2,4,5,6),x3=c(5,10,15,20))
> data1
  x1 x2 x3
1  1  2  5
2  2  4 10
3  3  5 15
4  4  6 20
> attach(data1)
> x1
[1] 1 2 3 4
> x2
[1] 2 4 5 6
> detach(data1)
```

attach(data1) x1 for(i in 1: 10) print("teju") x<- list(a=rnorm(20),b=1:5)
x
b<-lapply(x,sum)



```
> x1
[1] 1 2 3 4
> for(i in 1: 10)
+   print("teju")
[1] "teju"
[1] "teju"
[1] "teju"
[1] "teju"
[1] "teju"
[1] "teju"
```

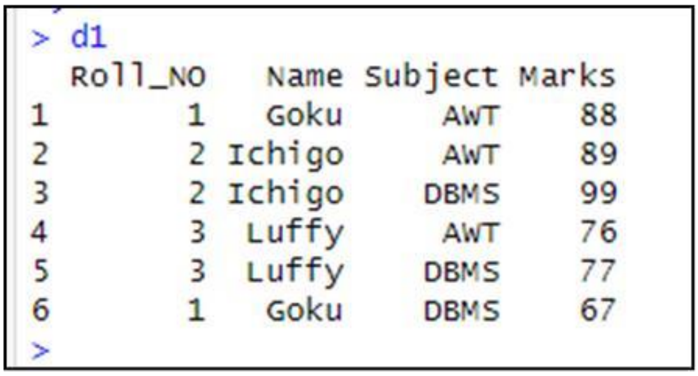


```
[1] "teju"
[1] "teju"
[1] "teju"
[1] "teju"
> x<-list(a=rnorm(20),b=1:5)
> x
$a
 [1]  1.75500761  0.52024199 -0.11191419 -1.95661841 -0.49198794
 [6]  0.91905178 -0.04218807  0.90124270  0.41263073  0.94550275
[11] -0.22681510  0.54584975  1.70856655  0.02886962  0.75930249
[16] -1.21103716 -0.80743685 -0.17424110 -0.60329656 -0.78726330
```

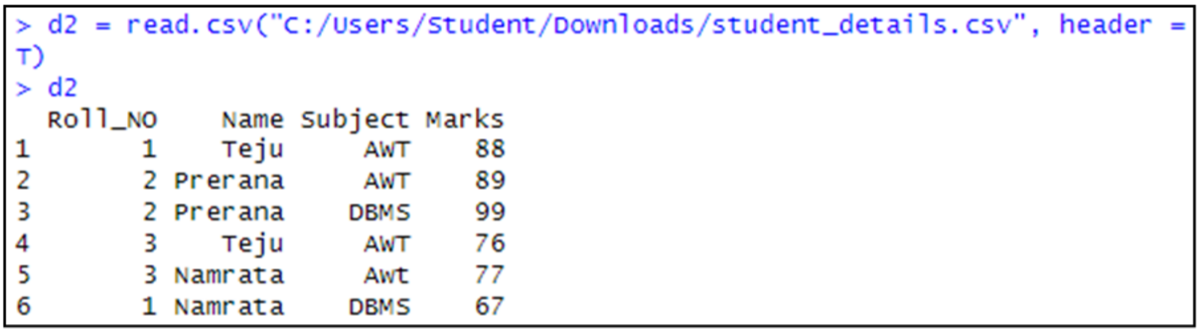
T)
b d1

# d2 data frame
d2 = read.csv("C:/Users/Student/Downloads/student_details.csv", header = T) d2

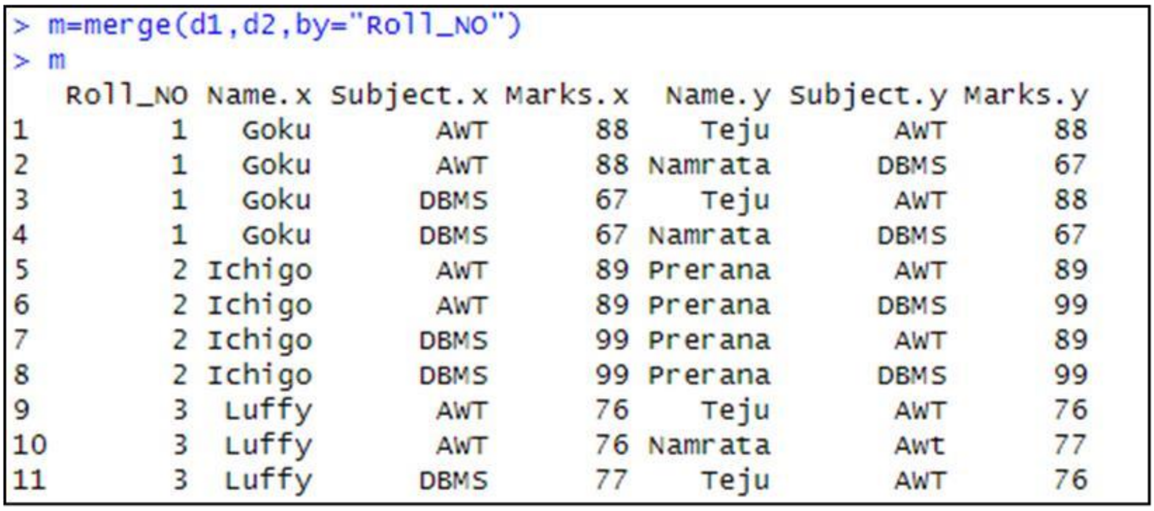
m=merge(d1,d2,by="Roll_NO")
m



```
> d1
  Roll_NO   Name Subject Marks
1       1   Goku     AWT    88
2       2 Ichigo     AWT    89
3       2 Ichigo    DBMS    99
4       3  Luffy     AWT    76
5       3  Luffy    DBMS    77
6       1   Goku    DBMS    67
>
```



```
> d2 = read.csv("C:/Users/Student/Downloads/student_details.csv", header =
T)
> d2
  Roll_NO    Name Subject Marks
1       1    Teju     AWT    88
2       2 Prerana     AWT    89
3       2 Prerana    DBMS    99
4       3    Teju     AWT    76
5       3 Namrata     Awt    77
6       1 Namrata    DBMS    67
```



```
> m=merge(d1,d2,by="Roll_NO")
> m
   Roll_NO Name.x Subject.x Marks.x  Name.y Subject.y Marks.y
1        1   Goku       AWT      88    Teju       AWT      88
2        1   Goku       AWT      88 Namrata      DBMS      67
3        1   Goku      DBMS      67    Teju       AWT      88
4        1   Goku      DBMS      67 Namrata      DBMS      67
5        2 Ichigo       AWT      89 Prerana       AWT      89
6        2 Ichigo       AWT      89 Prerana      DBMS      99
7        2 Ichigo      DBMS      99 Prerana       AWT      89
8        2 Ichigo      DBMS      99 Prerana      DBMS      99
9        3  Luffy       AWT      76    Teju       AWT      76
10       3  Luffy       AWT      76 Namrata       Awt      77
11       3  Luffy      DBMS      77    Teju       AWT      76
```

Naive Bayesian

#install package e1071, holds the Naive Bayes classifier install.package("e1071")

#loading the library
library(e1071)

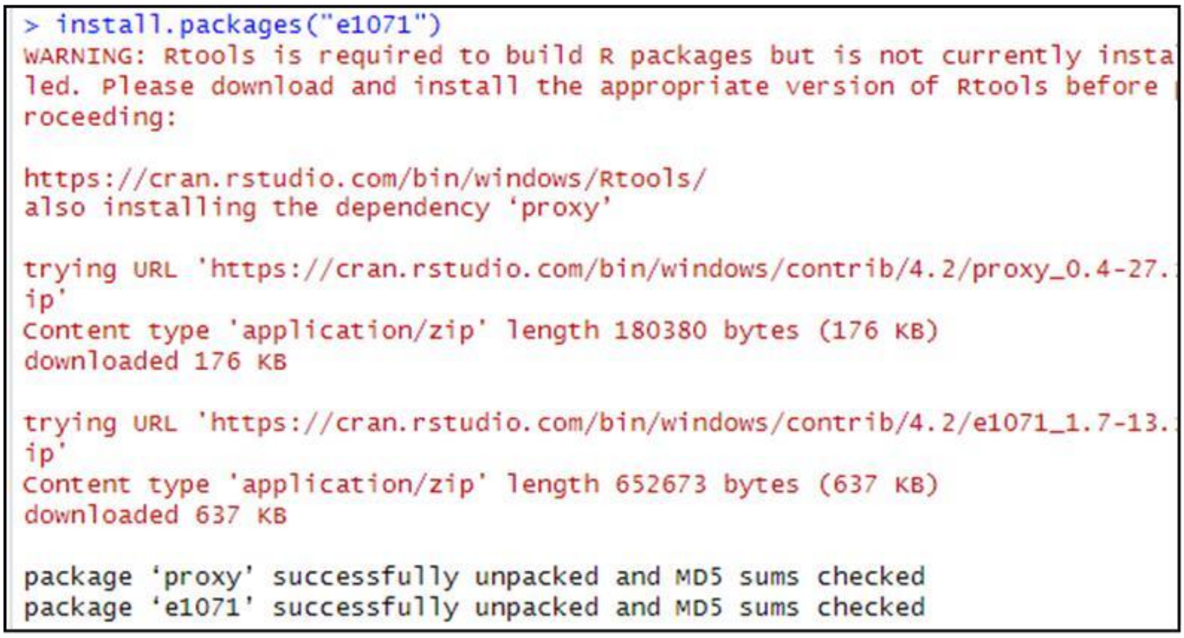
#loading the data
bc=read.csv(file.choose(),header=T)

#splitting the data into train and test data train=bc[1:450,]
test=bc[451:569,]

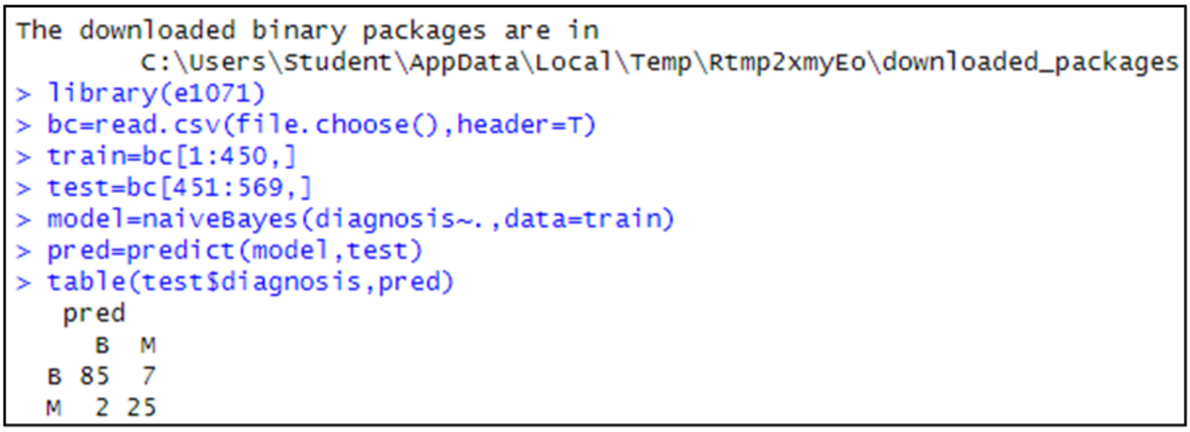
#creating the model
model=naiveBayes(diagnosis~.,data=train)

#prediction using the model pred=predict(model,test)

#confusion matrix to test the accuracy table(test$diagnosis,pred)



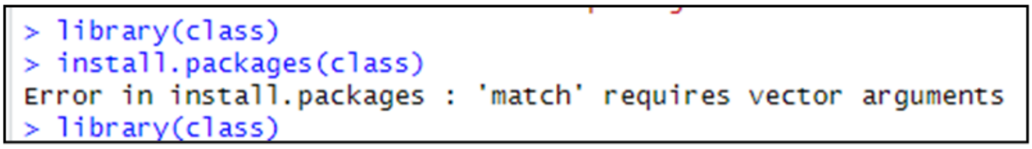
```
> install.packages("e1071")
WARNING: Rtools is required to build R packages but is not currently insta
led. Please download and install the appropriate version of Rtools before 
roceeding:

https://cran.rstudio.com/bin/windows/Rtools/
also installing the dependency ‘proxy’

trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.2/proxy_0.4-27.
ip'
Content type 'application/zip' length 180380 bytes (176 KB)
downloaded 176 KB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.2/e1071_1.7-13.
ip'
Content type 'application/zip' length 652673 bytes (637 KB)
downloaded 637 KB

package ‘proxy’ successfully unpacked and MD5 sums checked
package ‘e1071’ successfully unpacked and MD5 sums checked
```



```
The downloaded binary packages are in
        C:\Users\Student\AppData\Local\Temp\Rtmp2xmyEo\downloaded_packages
> library(e1071)
> bc=read.csv(file.choose(),header=T)
> train=bc[1:450,]
> test=bc[451:569,]
> model=naiveBayes(diagnosis~.,data=train)
> pred=predict(model,test)
> table(test$diagnosis,pred)
   pred
     B  M
  B 85  7
  M  2 25
```

K Nearest Neighbour

#installing adding the class package for knn
install.packages(class) library(class)



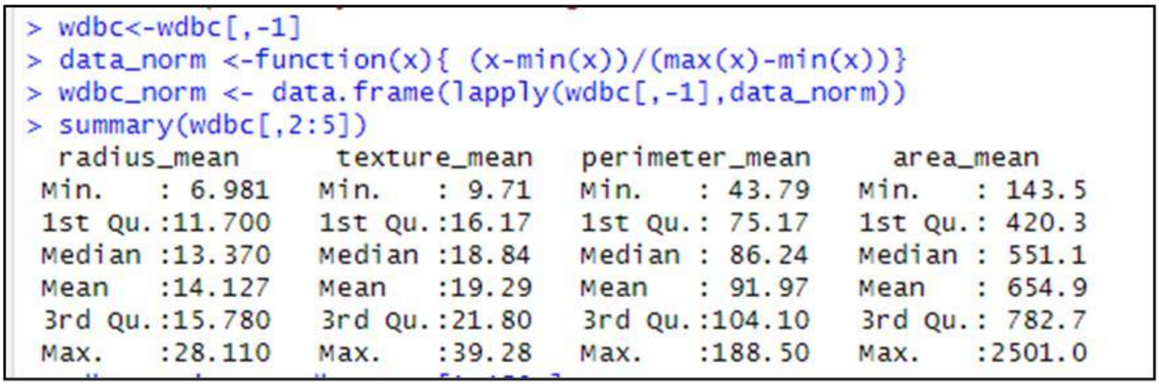
```
> library(class)
> install.packages(class)
Error in install.packages : 'match' requires vector arguments
> library(class)
```

#adding the dataset
wdbc <- read.csv(file.choose(),header=T)

```
> wdbc <- read.csv(file.choose(),header=T
```

#Data Cleaning
Removing the id column which is unnecessary wdbc<-wdbc[,-1]

#Normalize the data data_norm <-function(x){ (x-min(x))/(max(x)-min(x))} wdbc_norm <- data.frame(lapply(wdbc[,-1],data_norm)) summary(wdbc[,2:5])

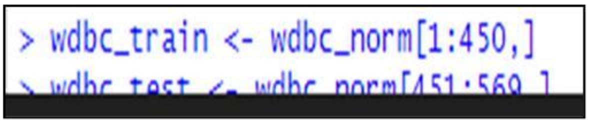


```
> wdbc<-wdbc[,-1]
> data_norm <-function(x){ (x-min(x))/(max(x)-min(x))}
> wdbc_norm <- data.frame(lapply(wdbc[,-1],data_norm))
> summary(wdbc[,2:5])
  radius_mean      texture_mean   perimeter_mean     area_mean
 Min.   : 6.981   Min.   : 9.71   Min.   : 43.79   Min.   : 143.5
 1st Qu.:11.700   1st Qu.:16.17   1st Qu.: 75.17   1st Qu.: 420.3
 Median :13.370   Median :18.84   Median : 86.24   Median : 551.1
 Mean   :14.127   Mean   :19.29   Mean   : 91.97   Mean   : 654.9
 3rd Qu.:15.780   3rd Qu.:21.80   3rd Qu.:104.10   3rd Qu.: 782.7
 Max.   :28.110   Max.   :39.28   Max.   :188.50   Max.   :2501.0
```

summary(wdbc_norm[,1:4])

#Creating Training and Testing dataset wdbc_train <- wdbc_norm[1:450,]

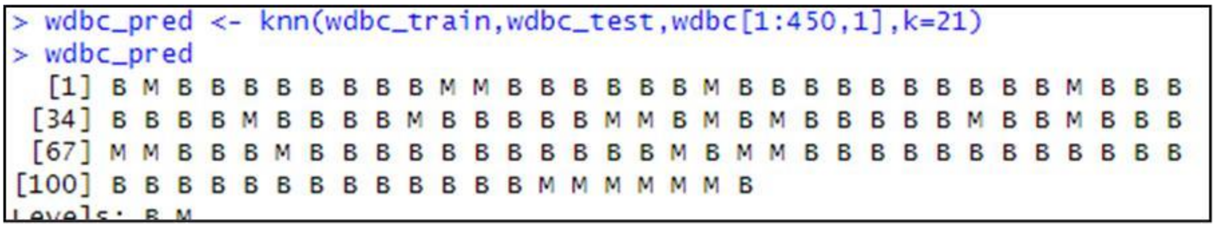
wdbc_test <- wdbc_norm[451:569,]



```
> wdbc_train <- wdbc_norm[1:450,]
> wdbc_test <- wdbc_norm[451:569,]
```

#Applying KNN model

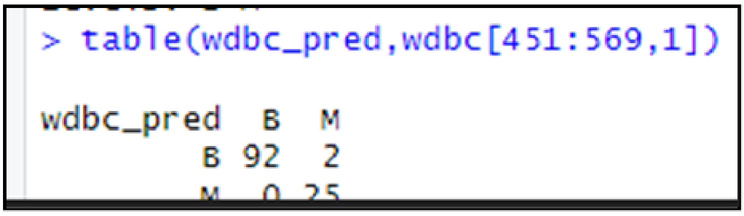
wdbc_pred <- knn(wdbc_train,wdbc_test,wdbc[1:450,1],k=21) wdbc_pred



```
> wdbc_pred <- knn(wdbc_train,wdbc_test,wdbc[1:450,1],k=21)
> wdbc_pred
  [1] B M B B B B B B B B M M B B B B B B M B B B B B B B B B B M B B B
 [34] B B B B M B B B B M B B B B B M M B M B M B B B B B M B B M B B B
 [67] M M B B B M B B B B B B B B B B B M B M M B B B B B B B B B B B B
[100] B B B B B B B B B B B B B M M M M M M B
Levels: B M
```

#confusion matrix or frequency table

Table function in R table(), performs categorical tabulation of data with the variable and its frequency. table(wdbc_pred,wdbc[451:569,1])

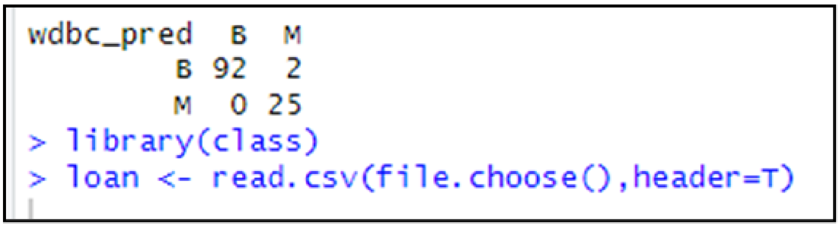


```
> table(wdbc_pred,wdbc[451:569,1])

wdbc_pred  B  M
        B 92  2
        M  0 25
```

#Import Loan dataset

library(class)

loan <- read.csv(file.choose(),header=T) str(loan)



```
wdbc_pred  B  M
        B 92  2
        M  0 25
> library(class)
> loan <- read.csv(file.choose(),header=T)
```

```
#Data Cleaning  loan
<-
loan[c('existing_credits','checking_balance','months_loan_duration','credit_history','purpos
e','
amount','savings_balance','employment_length','personal_status','other_debtors','property'
,'ag e','housing','dependents','foreign_worker','job')]
```

```
#Creating Training and Testing dataset
nrow(loan)  [1] 858 loan_train <-
loan_norm[1:297,]
loan_test<-loan_norm[298:424,]
```

```
#Creating Training and Testing dataset
> nrow(loan)
[1] 858
> loan_train <- loan_norm[1:297,]
> loan_test<-loan_norm[298:424,]
```

```
#Applying KNN model
loan_pred <- knn(loan_train,loan_test,loan[1:297,1],k=21)
> summary(loan_pred)
1 2
69 58
```

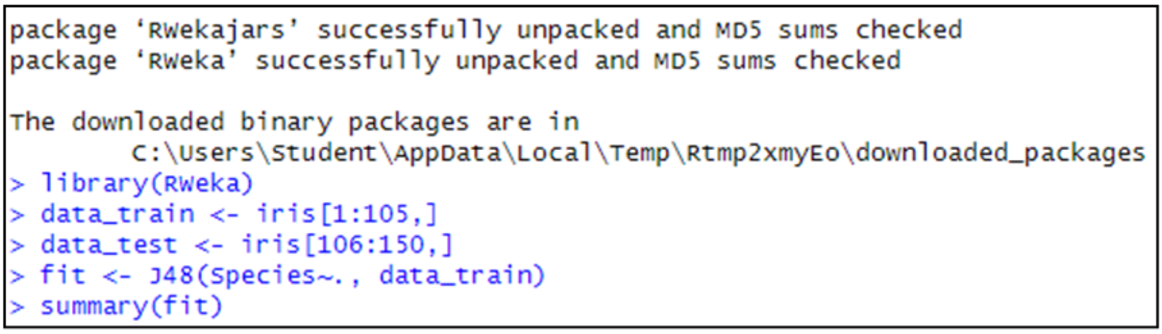
```
#Confusion Matrix or frequency table
> table(loan_pred,loan[298:424,1])

loan_pred 1 2
1 43 26
2 33 25
> summary(loan_pred)
1 2
69 58
```

C4.5

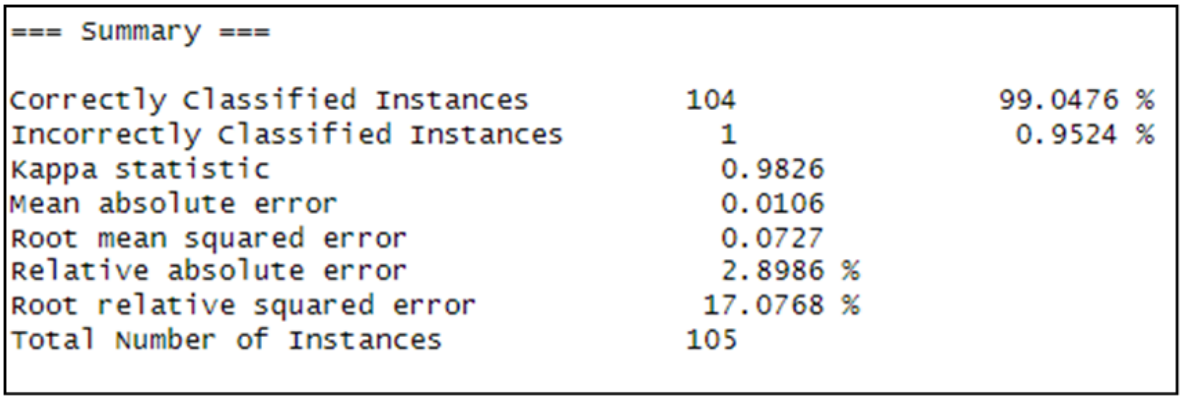
The C4.5 algorithm is an extension of the ID3 algorithm and constructs a decision tree to maximize information gain (difference in entropy).

.

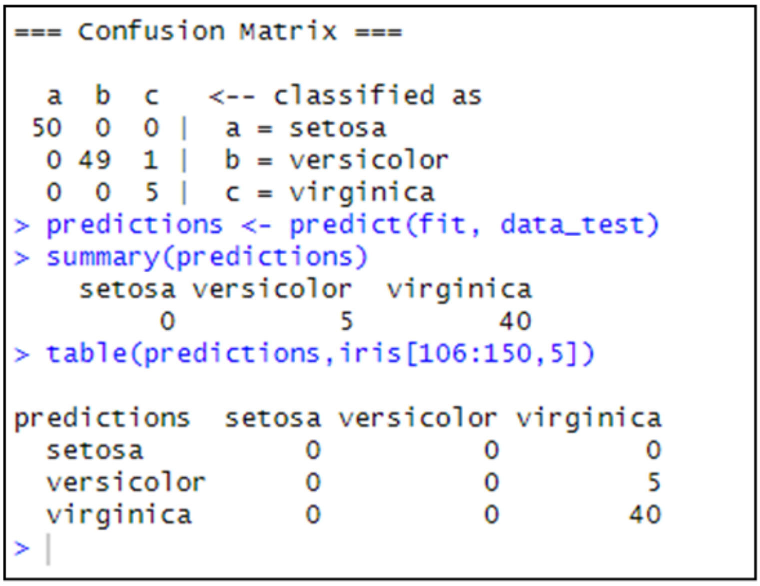
```
#install the package RWeka
install.packages("RWeka")

library(RWeka)  data_train
<- iris[1:105,]
data_test <- iris[106:150,]

fit <- J48(Species~., data_train)  summary(fit)
predictions <- predict(fit, data_test)
summary(predictions)

table(predictions,iris[106:150,5])
```



```
package ‘RWekajars’ successfully unpacked and MD5 sums checked
package ‘RWeka’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Student\AppData\Local\Temp\Rtmp2xmyEo\downloaded_packages
> library(RWeka)
> data_train <- iris[1:105,]
> data_test <- iris[106:150,]
> fit <- J48(Species~., data_train)
> summary(fit)
```



```
=== Summary ===

Correctly Classified Instances         104               99.0476 %
Incorrectly Classified Instances         1                0.9524 %
Kappa statistic                          0.9826
Mean absolute error                      0.0106
Root mean squared error                  0.0727
Relative absolute error                  2.8986 %
Root relative squared error             17.0768 %
Total Number of Instances              105
```



```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 50  0  0 |  a = setosa
  0 49  1 |  b = versicolor
  0  0  5 |  c = virginica
> predictions <- predict(fit, data_test)
> summary(predictions)
    setosa versicolor  virginica
         0          5         40
> table(predictions,iris[106:150,5])

predictions  setosa versicolor virginica
  setosa          0          0         0
  versicolor      0          0         5
  virginica       0          0        40
> 
```

Q3. Implementation and analysis of Apriori Algorithm using Market Basket Analysis.

# Apriori Algorithm install.packages("ddply")
#install.packages(arules)
#install.packages(arules)

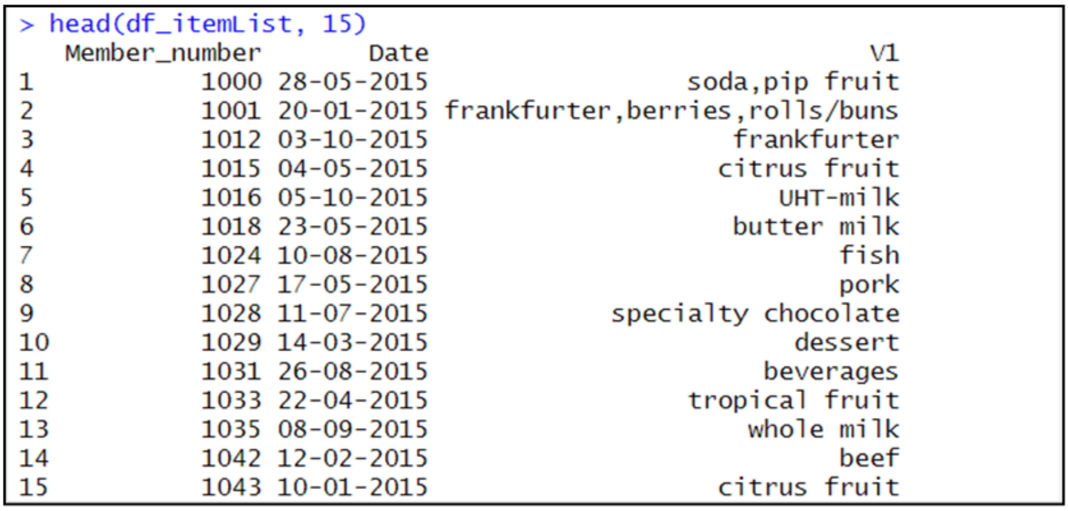
# Read the data
df_groceries <- read.csv("Groceries_dataset.csv")

# Data cleaning and manipulations using R

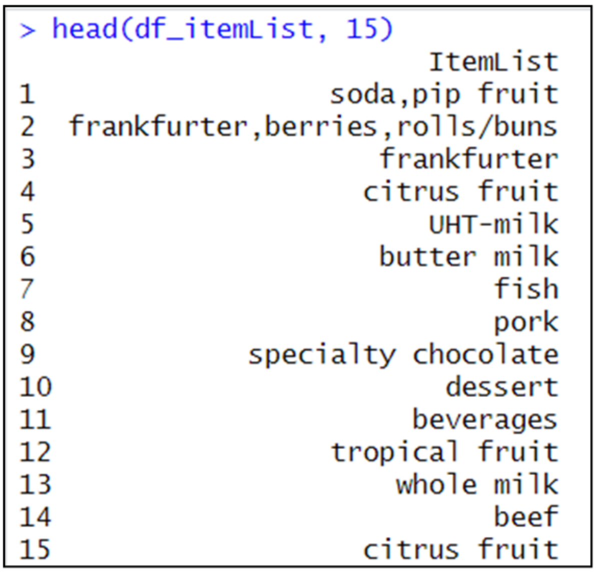
#First make sure that the Member numbers are of numeric data type and then #sort the dataframe based on the Member_number.

df_sorted <- df_groceries[order(df_groceries$Member_number),]
df_sorted$Member_number <- as.numeric(df_sorted$Member_number)
#install.packages("plyr")
library(plyr) format df_itemList <-
ddply(df_groceries,c("Member_number","Date"),

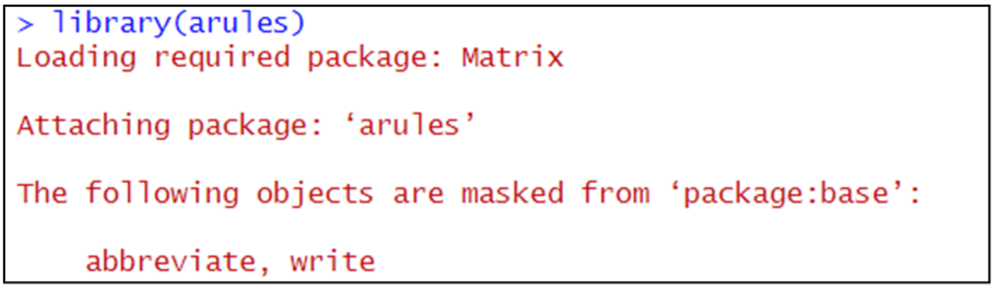
```
function(df1)paste(df1$itemDescription,collapse = ","))
head(df_itemList, 15)

df_itemList$Member_number <- NULL  # drop (delete) columns  df_itemList$Date
<- NULL

#Rename column headers for ease of use colnames(df_itemList)
<- c("ItemList")

head(df_itemList, 15)

#Write dataframe to a csv file using write.csv()
write.csv(df_itemList,"Grocery_ItemList1.csv", row.names = TRUE)


#Find the association rules #install.packages("arules")
library(arules)

txn = read.transactions(file="Grocery_ItemList1.csv", rm.duplicates= TRUE,
format="basket",sep=",",cols=1);
txn

basket_rules <- apriori(txn,parameter = list(sup = 0.01, conf = 0.01));

print(basket_rules)

inspect(basket_rules)

#install.packages("arulesViz")
library(arulesViz)
plot(basket_rules)

#Graph to display top 5 items
itemFrequencyPlot(txn, topN = 5)
```



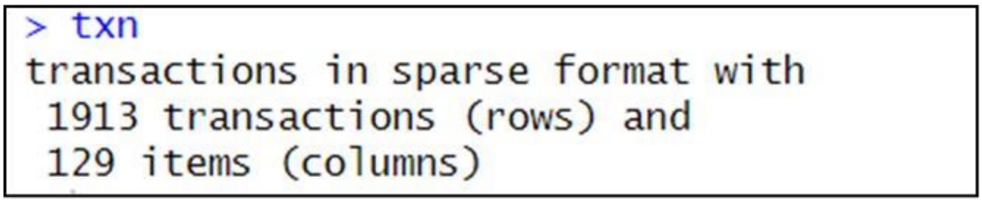
```
> head(df_itemList, 15)
   Member_number       Date                           V1
1           1000 28-05-2015               soda,pip fruit
2           1001 20-01-2015 frankfurter,berries,rolls/buns
3           1012 03-10-2015                  frankfurter
4           1015 04-05-2015                 citrus fruit
5           1016 05-10-2015                     UHT-milk
6           1018 23-05-2015                  butter milk
7           1024 10-08-2015                         fish
8           1027 17-05-2015                         pork
9           1028 11-07-2015          specialty chocolate
10          1029 14-03-2015                      dessert
11          1031 26-08-2015                    beverages
12          1033 22-04-2015               tropical fruit
13          1035 08-09-2015                   whole milk
14          1042 12-02-2015                         beef
15          1043 10-01-2015                 citrus fruit
```



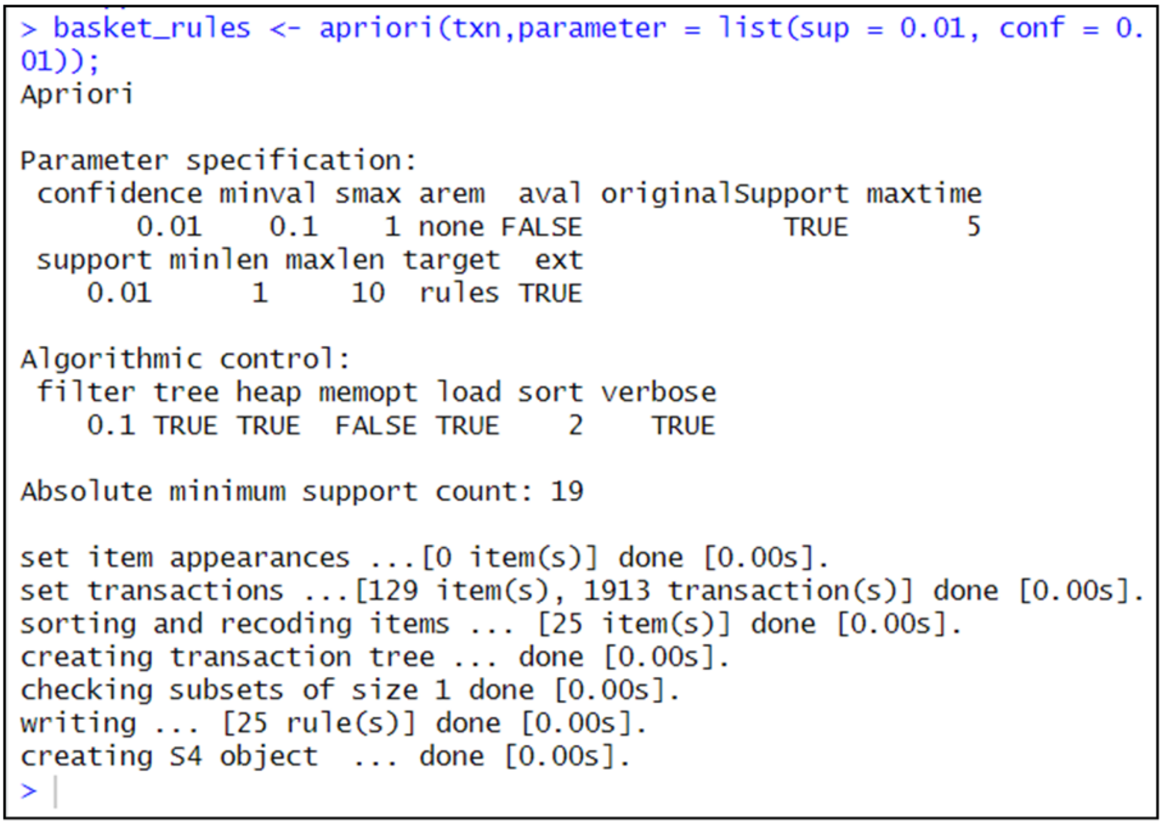
```
> head(df_itemList, 15)
                         ItemList
1                  soda,pip fruit
2  frankfurter,berries,rolls/buns
3                     frankfurter
4                    citrus fruit
5                        UHT-milk
6                     butter milk
7                            fish
8                            pork
9             specialty chocolate
10                        dessert
11                      beverages
12                 tropical fruit
13                     whole milk
14                           beef
15                   citrus fruit
```



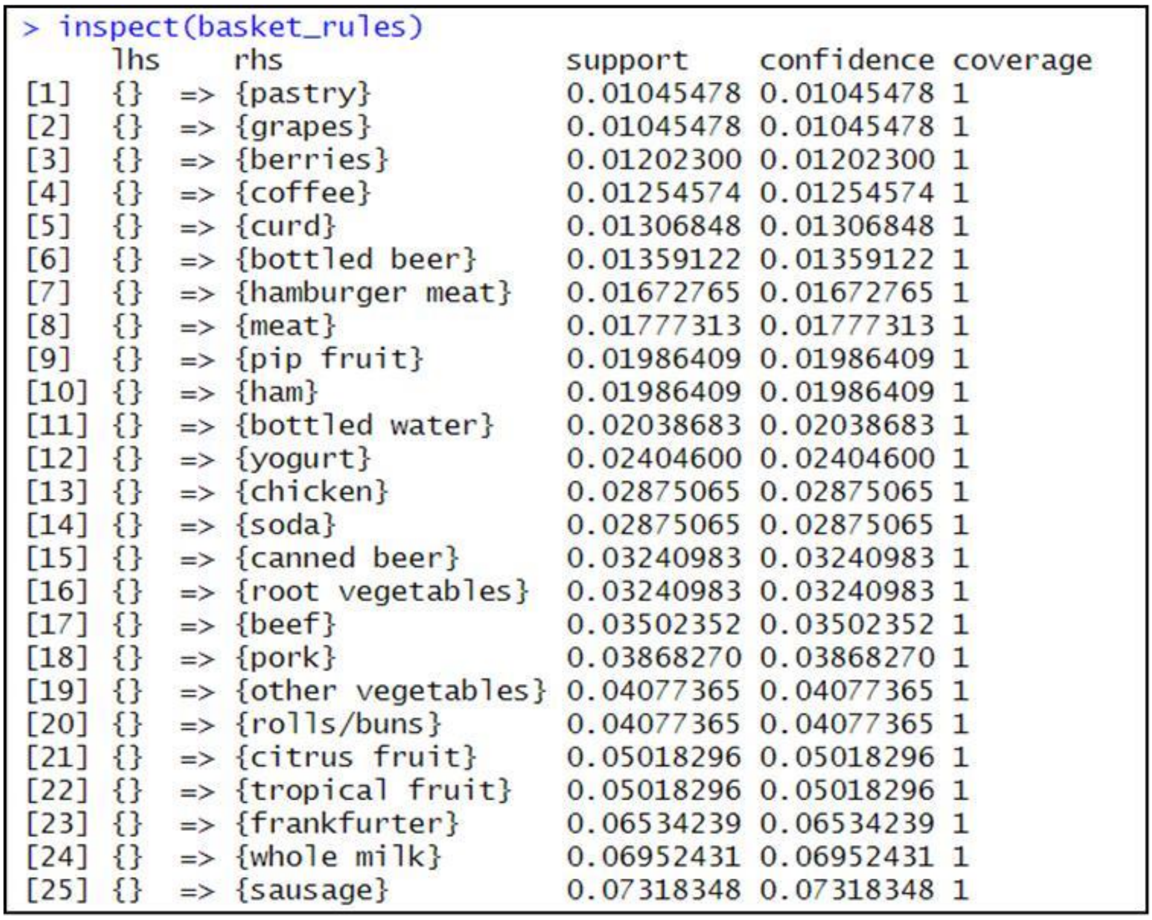
```
> library(arules)
Loading required package: Matrix

Attaching package: ‘arules’

The following objects are masked from ‘package:base’:

    abbreviate, write
```



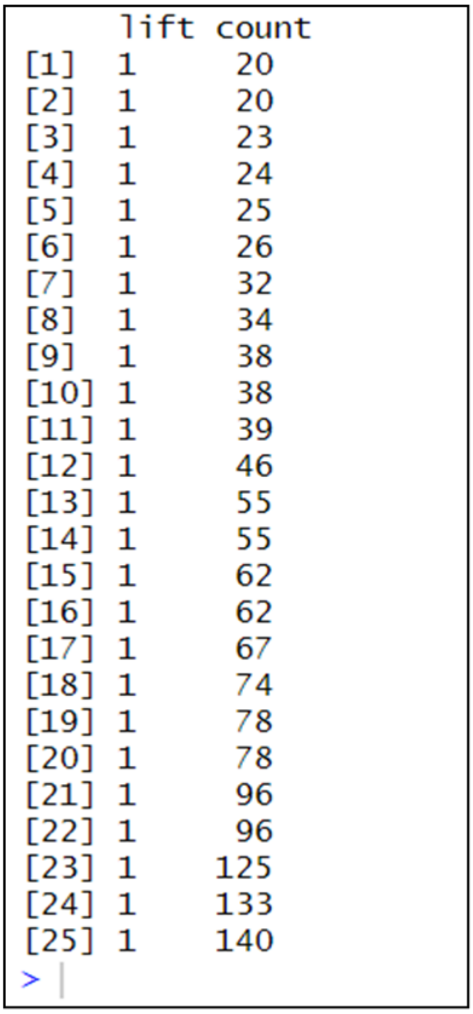
```
> txn
transactions in sparse format with
 1913 transactions (rows) and
 129 items (columns)
```



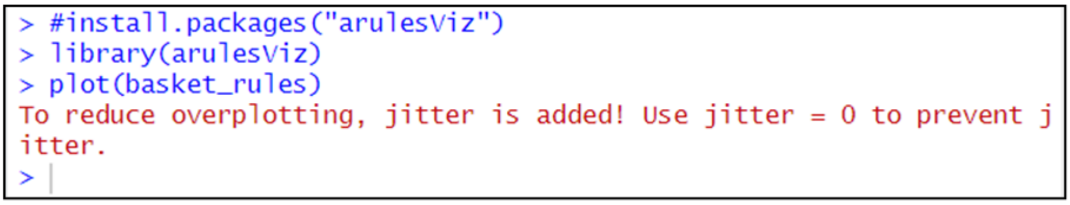
```
> basket_rules <- apriori(txn,parameter = list(sup = 0.01, conf = 0.
01));
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime
       0.01    0.1    1 none FALSE            TRUE       5
 support minlen maxlen target  ext
    0.01      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 19

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[129 item(s), 1913 transaction(s)] done [0.00s].
sorting and recoding items ... [25 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 done [0.00s].
writing ... [25 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> 
```



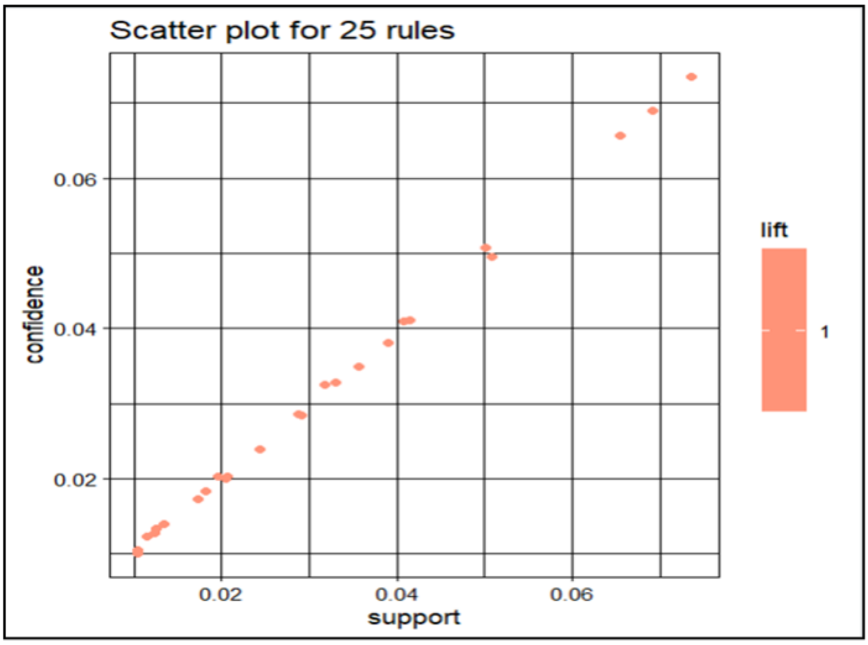
```
> inspect(basket_rules)
     lhs    rhs                support    confidence coverage
[1]  {}  => {pastry}           0.01045478 0.01045478 1
[2]  {}  => {grapes}           0.01045478 0.01045478 1
[3]  {}  => {berries}          0.01202300 0.01202300 1
[4]  {}  => {coffee}           0.01254574 0.01254574 1
[5]  {}  => {curd}             0.01306848 0.01306848 1
[6]  {}  => {bottled beer}     0.01359122 0.01359122 1
[7]  {}  => {hamburger meat}   0.01672765 0.01672765 1
[8]  {}  => {meat}             0.01777313 0.01777313 1
[9]  {}  => {pip fruit}        0.01986409 0.01986409 1
[10] {}  => {ham}              0.01986409 0.01986409 1
[11] {}  => {bottled water}    0.02038683 0.02038683 1
[12] {}  => {yogurt}           0.02404600 0.02404600 1
[13] {}  => {chicken}          0.02875065 0.02875065 1
[14] {}  => {soda}             0.02875065 0.02875065 1
[15] {}  => {canned beer}      0.03240983 0.03240983 1
[16] {}  => {root vegetables}  0.03240983 0.03240983 1
[17] {}  => {beef}             0.03502352 0.03502352 1
[18] {}  => {pork}             0.03868270 0.03868270 1
[19] {}  => {other vegetables} 0.04077365 0.04077365 1
[20] {}  => {rolls/buns}       0.04077365 0.04077365 1
[21] {}  => {citrus fruit}     0.05018296 0.05018296 1
[22] {}  => {tropical fruit}   0.05018296 0.05018296 1
[23] {}  => {frankfurter}      0.06534239 0.06534239 1
[24] {}  => {whole milk}       0.06952431 0.06952431 1
[25] {}  => {sausage}          0.07318348 0.07318348 1
```

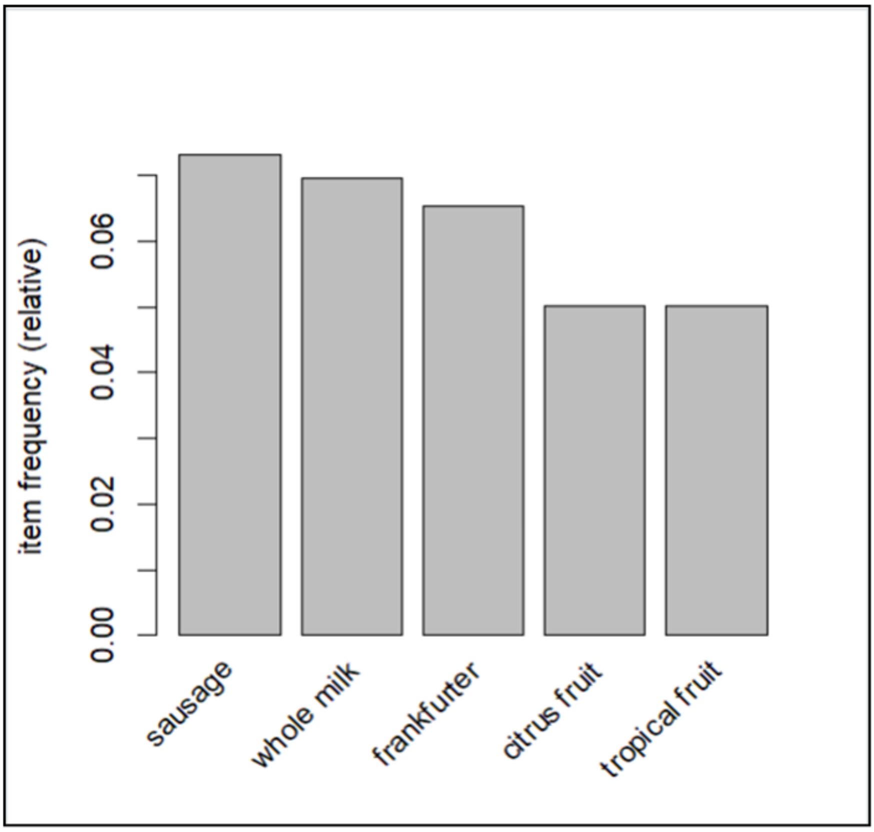


```
      lift count
[1]   1     20
[2]   1     20
[3]   1     23
[4]   1     24
[5]   1     25
[6]   1     26
[7]   1     32
[8]   1     34
[9]   1     38
[10]  1     38
[11]  1     39
[12]  1     46
[13]  1     55
[14]  1     55
[15]  1     62
[16]  1     62
[17]  1     67
[18]  1     74
[19]  1     78
[20]  1     78
[21]  1     96
[22]  1     96
[23]  1    125
[24]  1    133
[25]  1    140
> 
```



```
> #install.packages("arulesViz")
> library(arulesViz)
> plot(basket_rules)
To reduce overplotting, jitter is added! Use jitter = 0 to prevent j
itter.
> 
```

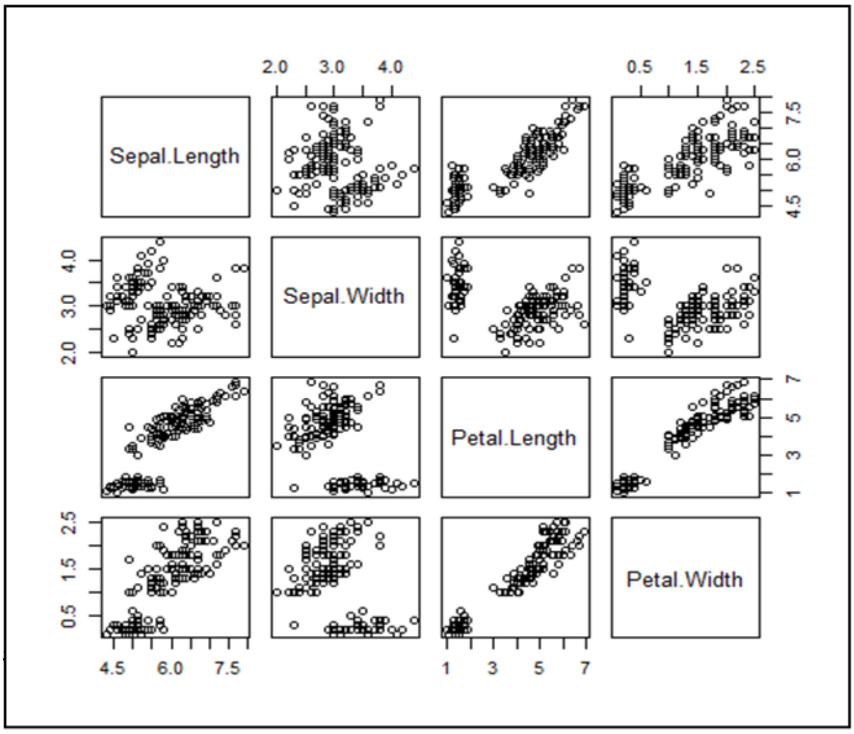


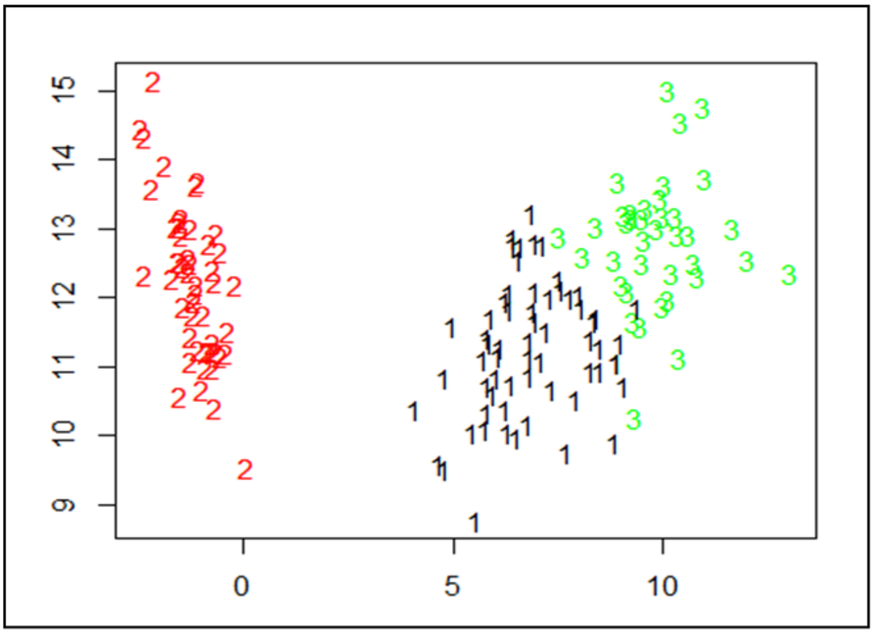


Q4. Implementation and analysis of clustering algorithms: K-Means and Agglomerative

Implementation and analysis of clustering algorithms: K-Means and Agglomerative

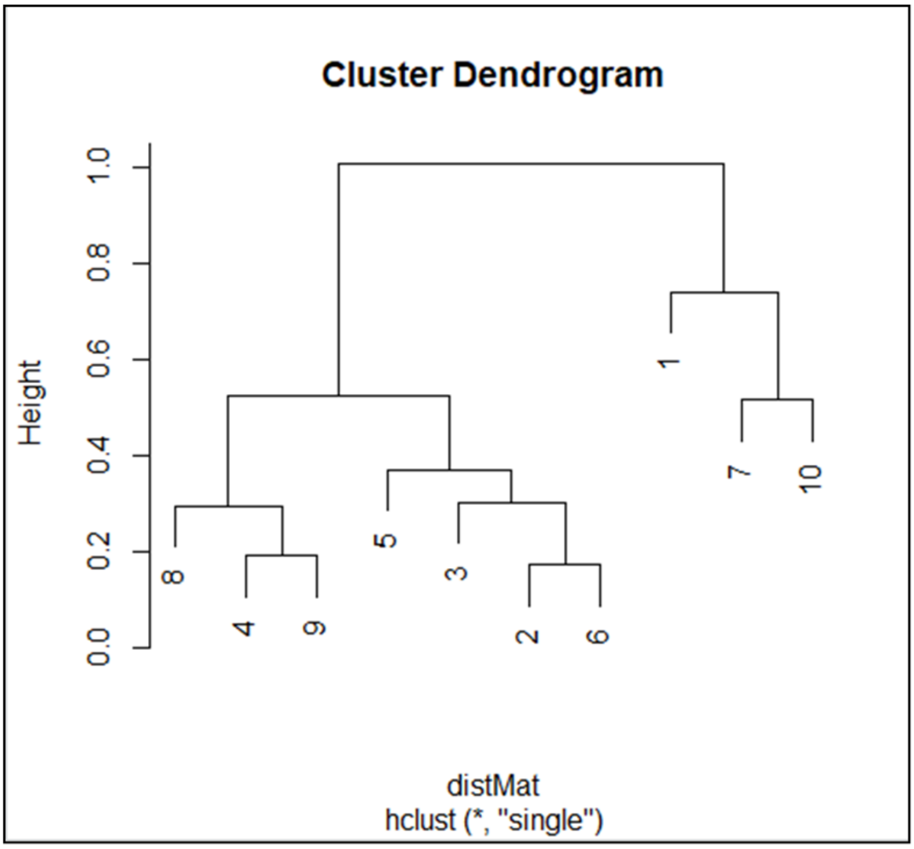
```
#Kmeans clustering

rm(list = ls()) # Free up memory gc()
# Garbage Collection

library(cluster)
library(fpc) # Flexible Procedures for Clustering

data(iris) summary(iris)
data.points <- iris[, -5] # Remove the class variable
head(data.points) plot(data.points)

# K-means clustering set.seed(789)
clust <- kmeans(data.points, centers=3,iter.max = 10) clust

table(iris$Species, clust$cluster)

plotcluster(data.points, clust$cluster)
```

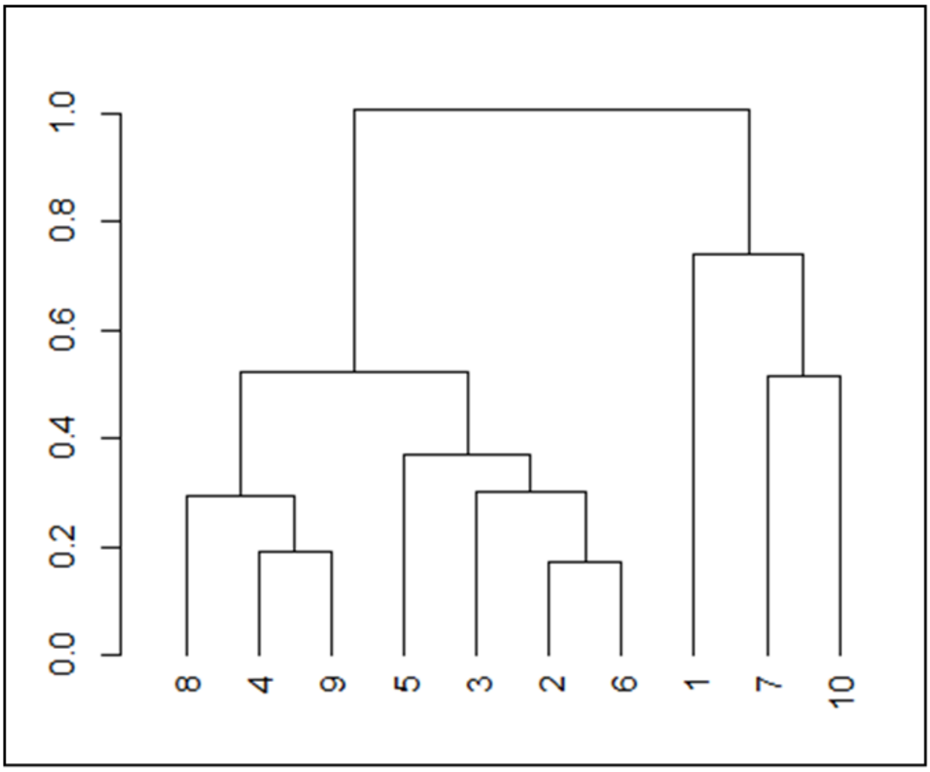




```
rm(list = ls()) # Free up memory gc()
# Garbage Collection

library(cluster)
#library(fpc) # Flexible Procedures for Clustering

data.points1 <- read.csv("seeds_dataset1.csv", header = TRUE)
#plot(data.points1)


distMat <- dist(data.points1,method = "euclidean")
Clust1 <- hclust(distMat,method="single")
Clust1
```

```
plot(Clust1)  #plot the clusters dend
<- as.dendrogram(Clust1

plot(dend)
```


Cluster Dendrogram
Height
distMat
hclust (*, "single")


0.0
0.2
0.4
0.6
0.8
1.0
8
4
9
5
3
2
6
1
7
10