

Deepak.H

(192210627)

Day-1

1. The intervals and corresponding frequencies are as follows. age frequency

1-5. 200

5-15 450

15-20 300

20-50 1500

50-80 700

80-110 44

Compute an approximate median value for the data

The screenshot shows the RStudio interface with the following details:

- Code Editor:** An R script named "age_frequency.R" is open. The code defines a vector "age" with values 5, 15, 20, 50, 80, 110 and a frequency vector "f" with values 200, 450, 300, 1500, 700, 44. It then calculates the median of both vectors.
- Environment View:** Shows the "Global Environment" pane with "age" and "f" defined. "age" is a numeric vector [1:6] with values 5, 15, 20, 50, 80, 110. "f" is a numeric vector [1:6] with values 200, 450, 300, 1500, 700, 44.
- Console:** Displays the R session output, including the source command, the definition of "age" and "f", and the resulting medians for each.
- File Explorer:** Shows a folder structure under "Home" containing various files like "1.cpp", "2d_dynamic_array.cpp", and several "26_09_23_" files.

2. Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 16, 19, 20, 20, 20, 21, 22, 22, 25, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

(a) What is the mean of the data? What is the median?

(b) What is the mode of the data? Comment on the data's modality (i.e., bimodal, trimodal, etc.).

(c) What is the midrange of the data?

(d) Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?

```

age_frequency_1.R
#mean,median,mode,quartile
age<-c(13,13,15,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
mean(age)
median(age)
mode<-names(table(age))[table(age)==max(table(age))]
mode_age
range(age)
quantile(age,.25)
quantile(age,.75)

```

Console output:

```

R 4.4.1 - /~/
> #mean,median,mode,quartile
> age<-c(13,13,15,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
> mean(age)
[1] 29.96296
> median(age)
[1] 25
> mode_age
[1] "25" "35"
> range(age)
[1] 13 70
> quantile(age,.25)
25%
[1] 20.5
> quantile(age,.75)
75%
[1] 35

```

3.Data Preprocessing: Reduction and Transformation Use the two methods below to normalize the following group of data: 200, 300, 400, 600, 1000 (a) min-max normalization by setting min = 0 and max = 1 (b) z-score normalization

```

R 4.4.1 - /~/
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> # Original data
> data <- c(200, 300, 400, 600, 1000)

> # Min-Max Normalization
> min_val <- 0
> max_val <- 1
> data_min <- min(data)
> data_max <- max(data)
> min_max_normalized <- (data - data_min) / (data_max - data_min) * (max_val - min_val) + min_val

> # Z-Score Normalization
> data_mean <- mean(data)
> data_sd <- sd(data)
> z_score_normalized <- (data - data_mean) / data_sd

> # Print results
print("Original data:")
[1] "Original data:"
print(data)
[1] 200 300 400 600 1000
print("Min-Max Normalized data:")
[1] "Min-Max Normalized data:"
> print(min_max_normalized)
[1] 0.000 0.125 0.250 0.500 1.000
> print("Z-Score Normalized data:")
[1] "Z-Score Normalized data:"
> print(z_score_normalized)
[1] -0.9486833 -0.6324555 -0.3162278 0.3162278 1.5811388
>

```

4.Data:11,13,13,15,15,16,19,20,20,21,21,22,23,24,30,40,45,45,45,71,72,73,75

a) Smoothing by bin mean

b) Smoothing by bin median

c) Smoothing by bin boundaries

The screenshot shows the RStudio interface with the following details:

- File Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project Bar:** Project: (None).
- Source Editor:** Contains R code for calculating mean, median, and mode across bins of age data.
- Environment View:** Shows a list of objects including `min_max_smooth` (List of 5), `age` (num [1:27]), `bin_indices` (Factor w/ 5 levels "(0,9,23,8]"), `bins` (5), `data` (num [1:24]), `mean_smooth` (num [1:5(1d)] 17.8 27 43.8 NA 72.8), `median_smooth` (num [1:5(1d)] 19.5 27 45 NA 72.5), and `mode_age` (chr [1:2] "25" "35").
- Files View:** Shows a file tree with various R files and executables from previous assignments.
- Console:** Displays the R session history with commands and their results.

```

#192211089
data <- c(11,13,13,15,15,16,19,20,20,21,21,22,23,24,30,40,45,45,45,71,72,73,75)
bins <- 5
bin_indices <- cut(data, bins)
mean_smooth <- tapply(data, bin_indices, mean)
median_smooth <- tapply(data, bin_indices, median)
min_max_smooth <- tapply(data, bin_indices, function(x) c(min(x), max(x)))
print(min_max_smooth)
11
3:1 (Top Level) R Script

Console Terminal Background Jobs
R 4.4.1 - ~/ ...
> #192211089
> data <- c(11,13,13,15,15,16,19,20,20,21,21,22,23,24,30,40,45,45,45,71,72,73,75)
> print(mean_smooth)
[10.9,23.8] (23.8,36.6] (36.6,49.4] (49.4,62.2] (62.2,75.1]
17.8571 27.00000 43.75000 NA 72.75000
> print(min_max_smooth)
$'(10.9,23.8]'
[1] 11 23
$'(23.8,36.6]'
[1] 24 30
$'(36.6,49.4]'
[1] 40 45

```

5. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

- Calculate the mean, median, and standard deviation of age and %fat.
- Draw the boxplots for age and %fat.
- Draw a scatter plot and a q-q plot based on these two variables.

The screenshot shows the RStudio interface with the following details:

- File Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project Bar:** Project: (None).
- Source Editor:** Contains R code for calculating summary statistics and creating boxplots for age and fat.
- Environment View:** Shows a list of objects including `min_max_smooth` (List of 5), `age` (num [1:18]), `bin_indices` (Factor w/ 5 levels "(0,9,23,8]"), `bins` (5), `data` (num [1:24]), `fat` (num [1:18]), `mean_smooth` (num [1:5(1d)] 9.5 26.5 7.8 17.8 31.4 25.9 27.4 27.2 31.2 34...), and `median_smooth` (num [1:5(1d)] 19.5 27 45 NA 72.5).
- Plots View:** Displays a side-by-side boxplot comparing age (left) and fat (right) for two groups (1 and 2). The y-axis ranges from 10 to 60.
- Console:** Displays the R session history with commands and their results.

```

#192211089
age<-c(23,27,27,39,41,47,49,50,52,54,54,56,57,58,60,61)
fat<-c(9.5,26.5,7.8,17.8,31.4,25.9,27.4,27.2,31.2,34...
mean(age)
[1] 46.44444
sd(age)
[1] 30.7
mean(fat)
[1] 28.78333
boxplot(age,fat)
>

```

6. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

(i) Use min-max normalization to transform the value 35 for age onto the range [0.0, 1.0].

(ii) Use z-score normalization to transform the value 35 for age, where the standard deviation of age is 12.94 years.

(iii) Use normalization by decimal scaling to transform the value 35 for age. Perform the above functions using R – tool

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Displays an R script named "Untitled1.R" containing the following code:

```
#192211089
v<-c(23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
min<-0
max<-1
min_max<-(max(v)-min(v))/(max(v)-min(v))
print(min_max)
m<-mean(v)
s<-12.94
z_score<-(35-m)/s
print(z_score)
decimal_scaling<-1
j<-max(m)<1
decimal_scaling<-m/10^j
print(decimal_scaling)
35
```
- Environment View:** Shows the following variables and their values:

values	
decimal_scaling	35
j	FALSE
m	35
max	1
min	0
min_max	0.315789473684211
s	12.94
v	num [1:18] 23 23 27 27 39 41 47 49 50 52 ...
z_score	-0.88442383651039
- Console View:** Displays the R session output:

```
R 4.4.1 -- f
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> source("~/active-rstudio-document")
[1] 0.3157895
[1] -0.8844238
[1] 35
> #192211089
> v<-c(23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
> print(min_max)
[1] 0.3157895
>
> print(z_score)
[1] -0.8844238
> print(decimal_scaling)
[1] 35
```

7. The following values are the number of pencils available in the different boxes. Create a vector and find out the mean, median and mode values of set of pencils in the given data.

Box1	Box2	Box3	Box4	Box5	Box6	Box7	Box8	Box9	Box 10
9	25	23	12	11	6	7	8	9	10

The screenshot shows the RStudio interface. In the top-left, there's a script editor with three tabs: Untitled1*, Untitled2*, and Untitled3*. Untitled3* contains the following R code:

```

1 #192211089
2 pencils<-c(9,25,23,12,11,6,7,8,9,10)
3 mean(pencils)
4 median(pencils)
5 mode.names(table(pencils))[table(pencils)==max(table(pencils))]
6 mode
7

```

In the top-right, the Environment viewer shows:

	mode	"9"
pencils	num [1:10]	9 25 23 12 11 6 7 8 9 10

At the bottom, the Console tab displays the R session history:

```

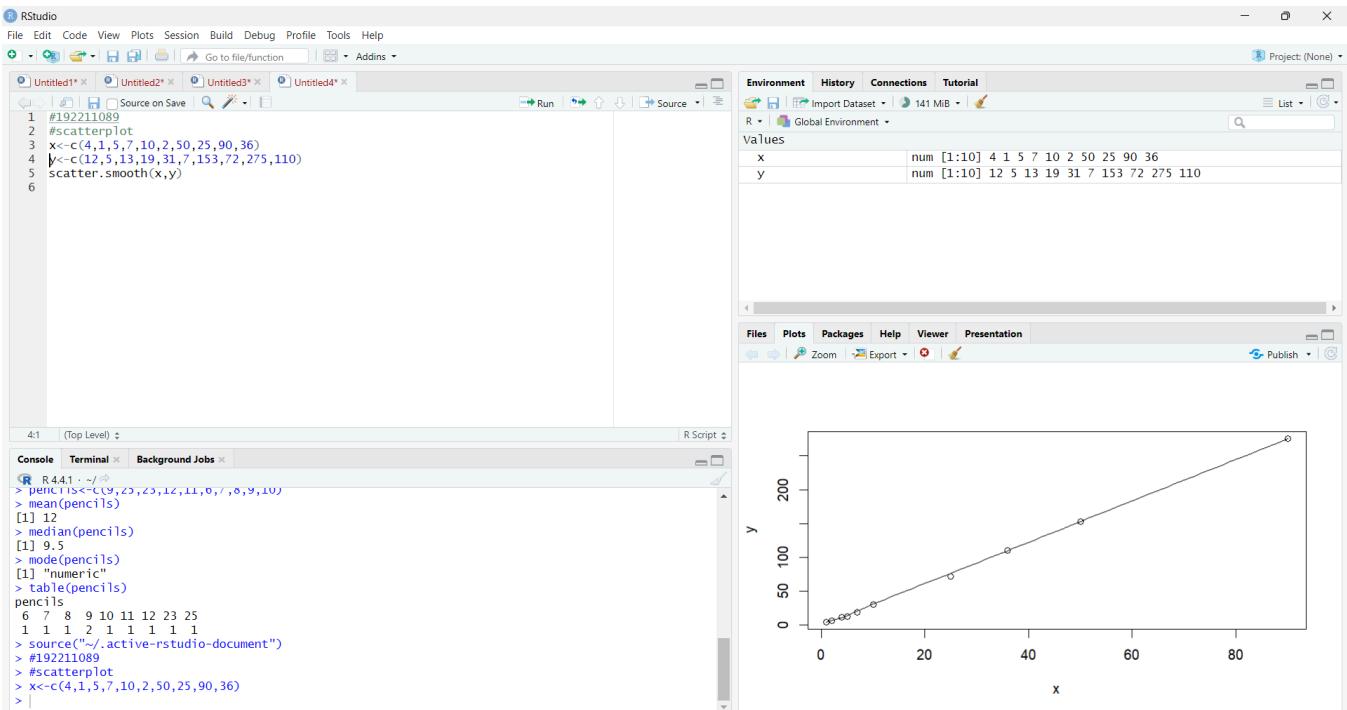
R 4.4.1 · ~/r
> source("~/active-rstudio-document")
> source("~/active-rstudio-document")
> #192211089
> pencils<-c(9,25,23,12,11,6,7,8,9,10)
> mean(pencils)
[1] 12
> median(pencils)
[1] 9.5
> mode(pencils)
[1] "numeric"
> table(pencils)
pencils
 6 7 8 9 10 11 12 13 23 25
1 1 1 2 1 1 1 1 1 1
>

```

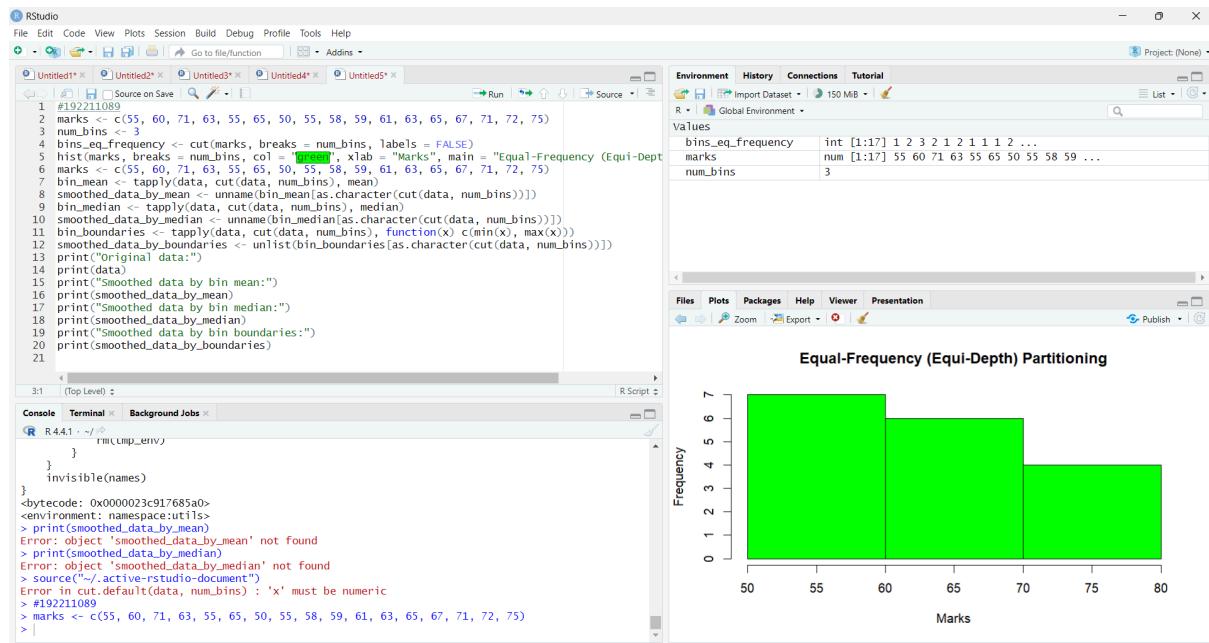
8. The following table would be plotted as (x,y) points, with the first column being the x values as several mobile phones sold and the second column being the y values as money. To use the scatter plot for how many mobile phones are sold.

x:4 1 5 7 10 2 50 25 90 36

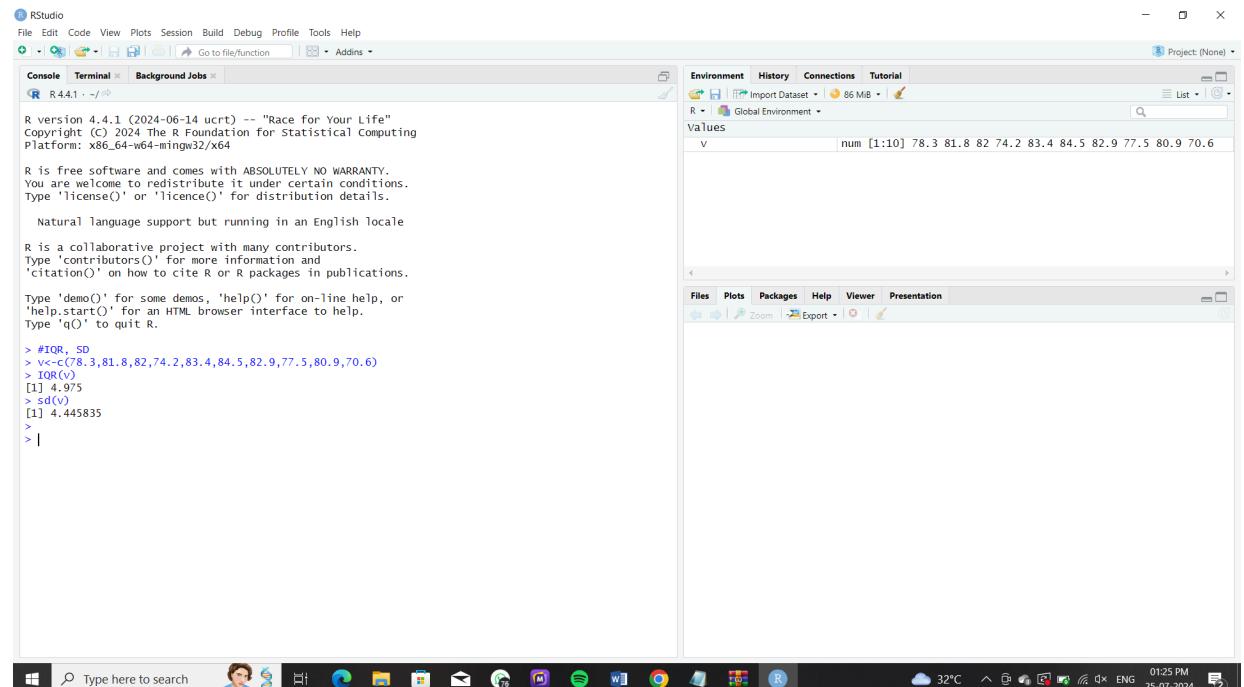
y:12 5 13 19 31 7 153 72 275 110



9. Implementing the R script using marks scored by a student in his model exam has been sorted as follows: 55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65, 67, 71, 72, 75. They are partitioned into three bins using each of the following methods. Plot the data points using the histogram.a) equal-frequency (equi-depth) partitioning (b) equal-width partitioning



10. Suppose that the speed car is mentioned in different driving style. Regular 78.3 81.8 82 74.2 83.4 84.5 82.9 77.5 80.9 70.6 Speed Calculate the Inter quantile and standard deviation of the given data.



11. Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?

The screenshot shows the RStudio interface. In the top-left pane, there are several tabs: 'untitled3*', 'Untitled4*', 'Untitled1*', 'Untitled5*', 'Untitled6*', 'Untitled7*', 'Untitled8*', 'Untitled9*', and 'untitled10'. The 'untitled10' tab contains the following R code:`1 #192211089
2 #Q1, Q2
3 age<-c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
4 quantile(age,.25)
5 quantile(age,.75)`

In the top-right pane, the 'Environment' tab is selected, showing a table with 'Values' and 'age' as a 'num [1:27]' vector containing the values: 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

In the bottom-left pane, the 'Console' tab is active, showing the same R code and its execution results:`R 4.4.1 : ~/ ◁
> source("./active-rstudio-document")
> #192211089
> #Q1, Q2
> age<-c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
> quantile(age,.25)
25%
20.5
> quantile(age,.75)
75%
35
> |`

DAY - 2

1.Covariance and correlation Children of three ages are asked to indicate their preference

for three photographs of adults. Do the data suggest that there is a significant relationship

between age and photograph preference? What is wrong with this study?
Photograph: Age

of child A B C 5-6 years: 18 22 20 7-8 years: 2 28 40 9-10 years: 20 10 40

1. Use cov() to calculate the sample covariance between B and C.

2. Use another call to cov() to calculate the sample covariance matrix for the preferences.

3. Use cor() to calculate the sample correlation between B and C.

4. Use another call to cor() to calculate the sample correlation matrix for the

The screenshot shows the RStudio interface with several panes:

- Code Editor:** Displays R code related to age preferences and covariance matrices.
- Environment:** Shows the global environment with objects like cov_matrix, preferences, and correlation values.
- Console:** Shows the R session history with commands and their outputs.
- Output:** Shows the results of the correlation matrix calculation.

```
1 #192211089
2 # Data
3 age_5_6 <- c(18, 22, 20)
4 age_7_8 <- c(2, 28, 40)
5 age_9_10 <- c(20, 10, 40)
6 # combine data into a matrix
7 preferences <- rbind(age_5_6, age_7_8, age_9_10)
8 colnames(preferences) <- c("A", "B", "C")
9 # Calculate the sample covariance between B and C
10 cov_BC <- cov(preferences[, "B"], preferences[, "C"])
11 cov_BC
12 # Calculate the sample covariance matrix for the preferences
13 cov_matrix <- cov(preferences)
14 Cov_matrix
15 # Calculate the sample correlation between B and C
16 cor_BC <- cor(preferences[, "B"], preferences[, "C"])
17 cor_BC
18 # Calculate the sample correlation matrix for the preferences
19 cor_matrix <- cor(preferences)
20 cor_matrix
```

4:1 (Top Level) R Script

R 4.4.1 - ./

```
> source("~/active-RsL010-document.R")
Error in eval(ei, envir) : object 'Cov_matrix' not found
> #192211089
> # Data
> age_5_6 <- c(18, 22, 20)
> cov(preferences)
A   B   C
A 97.33333 -74 -46.66667
B -74.00000  84 -20.00000
C -46.66667 -20 133.33333
> cor(preferences)
```

A	B	C
1.0000000	-0.8183918	-0.4096440
-0.8183918	1.0000000	-0.1889822
-0.4096440	-0.1889822	1.0000000

2. Imagine that you have selected data from the All Electronics data warehouse for analysis.

The data set will be huge! The following data are a list of All Electronics prices for commonly

sold items (rounded to the nearest dollar). The numbers have been sorted: 1, 1, 5, 5, 5, 5, 5,

8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20,

20, 20, 20, 20, 20, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30 the dataset using an equal-frequency partitioning method with bin equal to 3 (ii) apply data smoothing using bin

means and bin boundary. (iii) Plot Histogram for the above frequency division

The figure shows the RStudio interface with the following details:

- Code Editor:** The left pane displays R code for data manipulation and histogram generation. The code includes creating a data frame, setting bin sizes, calculating mean values for each bin, and plotting three different types of histograms.
- Console:** The bottom-left pane shows the R command-line interface output, including the execution of the R code and the results of the correlation matrix and preference scores.
- Environment View:** The top-right pane shows the global environment with various objects and their types and values.
- Plots:** Three histograms are displayed in the bottom-right pane:
 - Original Data Histogram:** A histogram of the original data with bins from 0 to 30.
 - Data Smoothed by Bin Means:** A histogram where the data has been smoothed by bin means, showing a much smoother distribution.
 - Data Smoothed by Bin Boundaries:** A histogram where the data has been smoothed by bin boundaries, showing a distribution with sharp peaks at integer price points.

3. Two Maths teachers are comparing how their Year 9 classes performed in the end of

year exams. Their results are as follows: Class A: 76, 35, 47, 64, 95, 66, 89, 36,

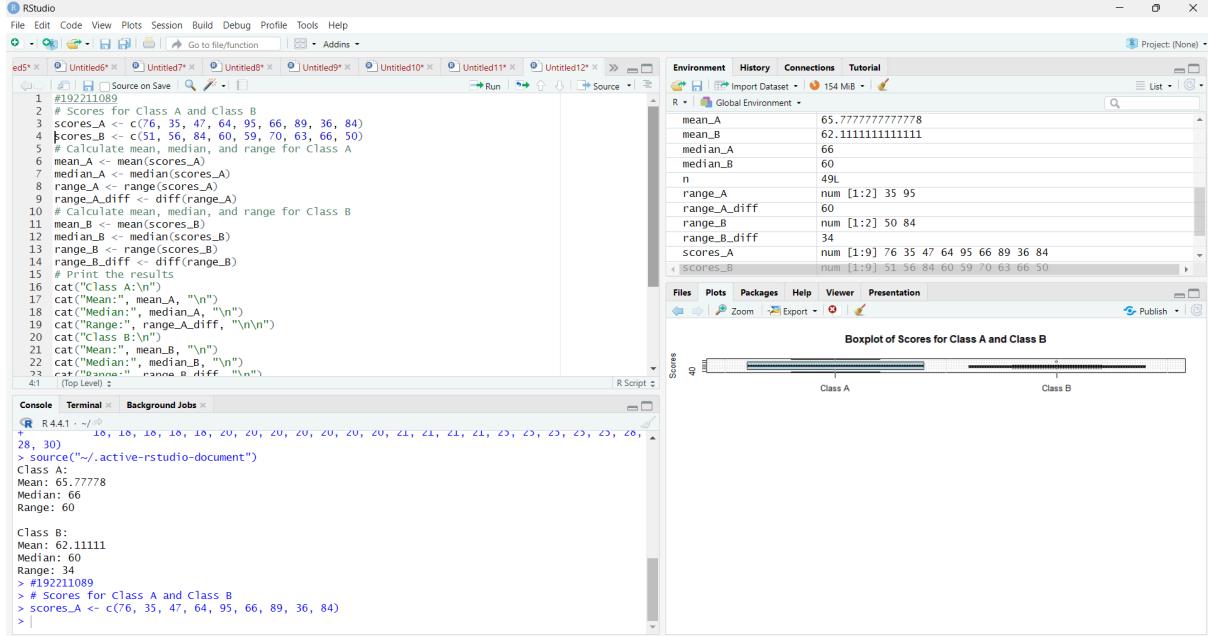
84,76,35,47,64,95,66,89,36,84 Class B: 51, 56, 84, 60, 59, 70, 63, 66,

5051,56,84,60,59,70,63,66,50 (i) Find which class had scored higher mean, median and

range. (ii) Plot above in boxplot and give the inferences Class B: 51, 56, 84, 60, 59, 70, 63,

66, 5051, 56, 84, 60, 59, 70, 63, 66, 50

66, 5051, 56, 84, 60, 59, 70, 63, 66, 50



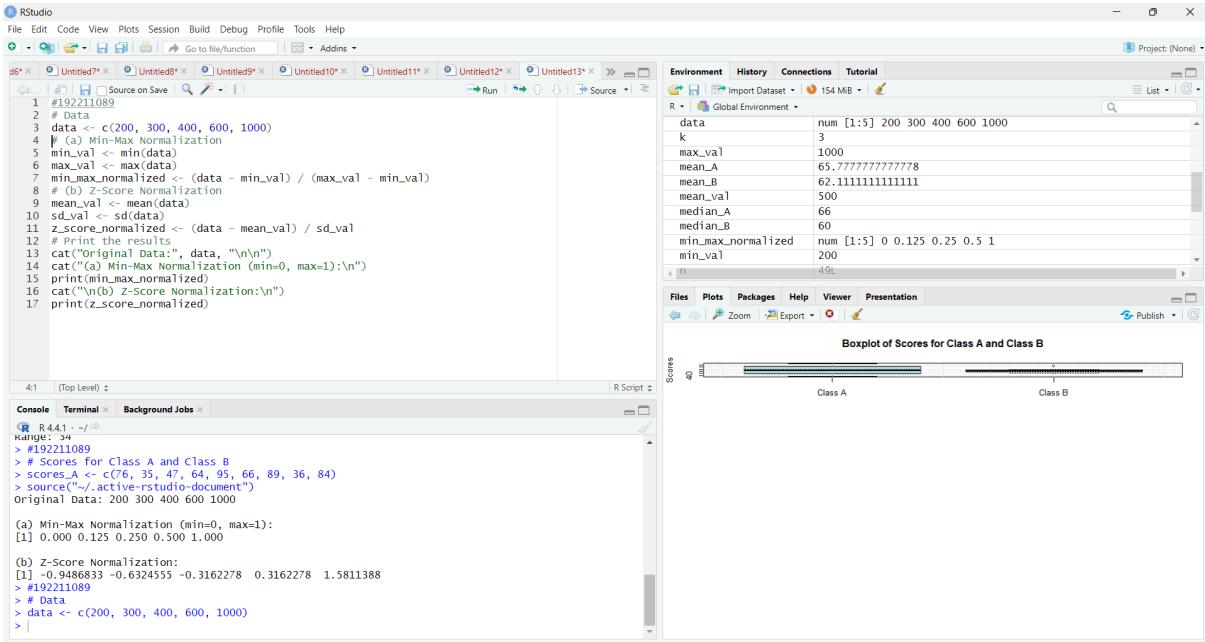
4. Let us consider one example to make the calculation method clear. Assume that the

minimum and maximum values for the feature F are \$50,000 and \$100,000 correspondingly.

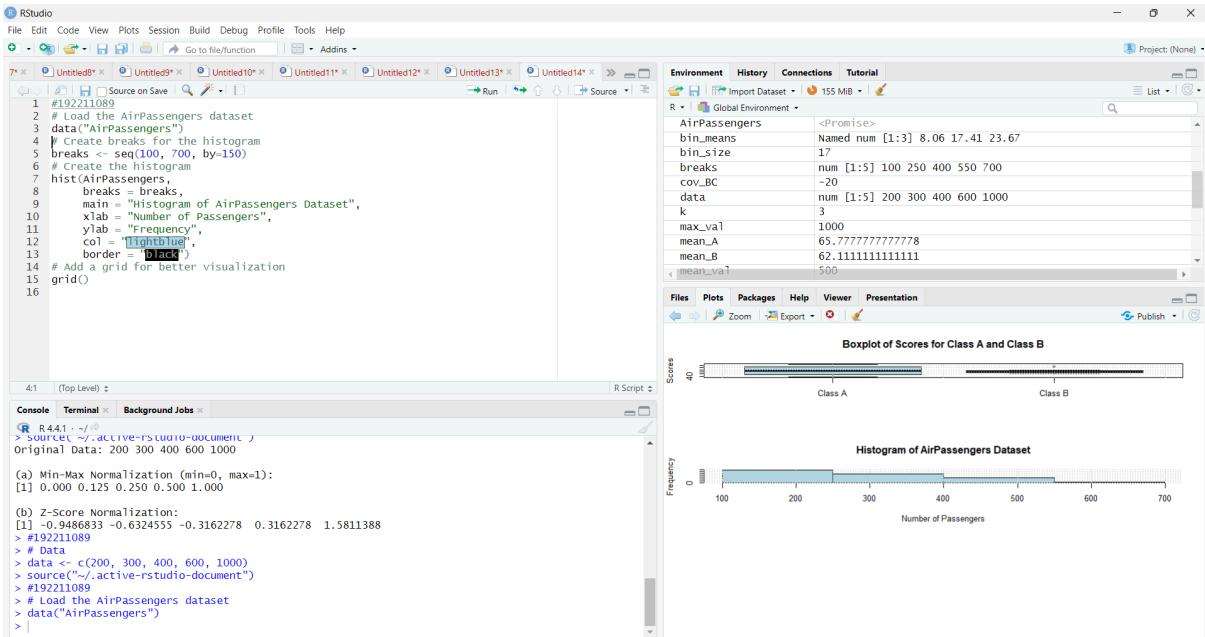
It needs to range F from 0 to 1. By min-max normalization, v = \$80, b) Use

the two methods below to normalize the following group of data: 200, 300, 400, 600, 1000

(a) min-max normalization by setting min = 0 and max = 1 (b) z-score normalization

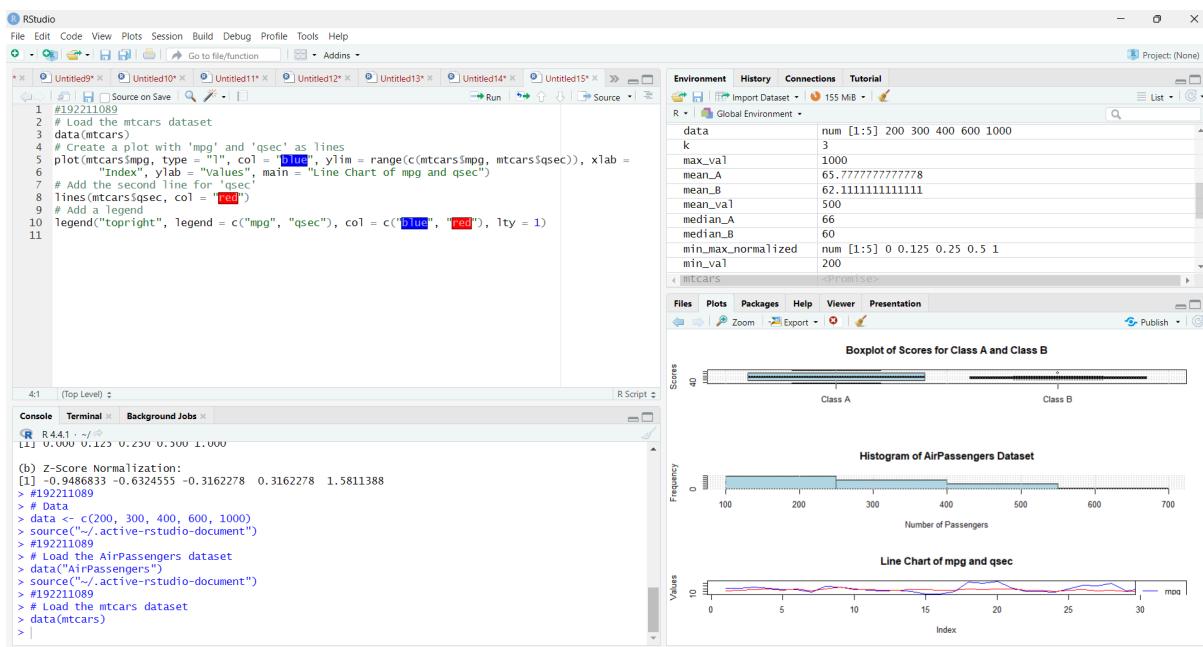


5. Make a histogram for the “AirPassengers” dataset, starting at 100 on the x-axis, and from values 200 to 700, make the bins 150 wide

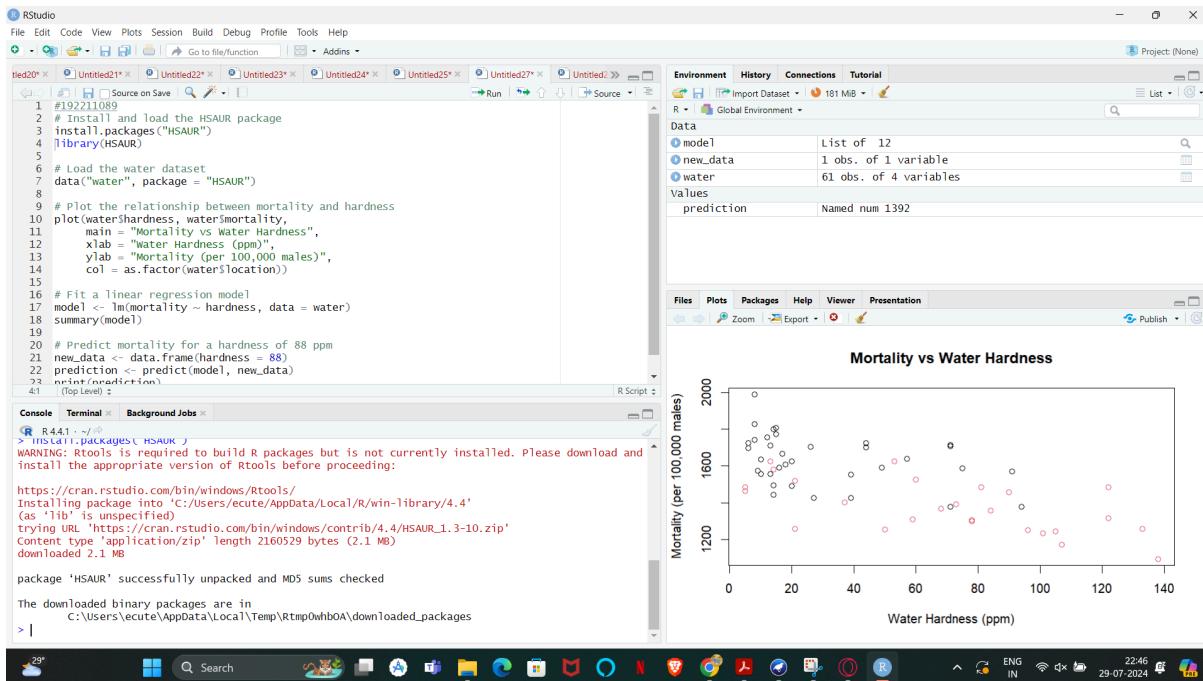


6. Obtain Multiple Lines in a Line Chart using a single Plot Function in Use attributes “mpg” and “sec” of the dataset “mtcars”

Code:

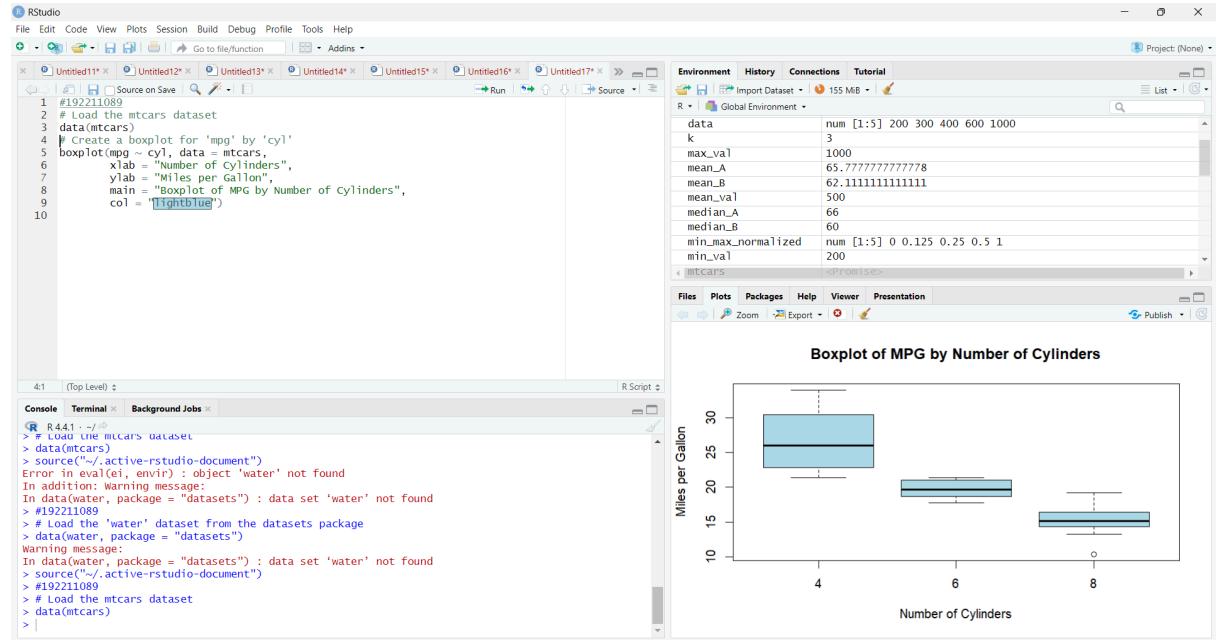


7. Download the Dataset "water" From the R dataset Link. Find out whether there is a linear relation between attributes "mortality" and "hardness" by plot function.Fit



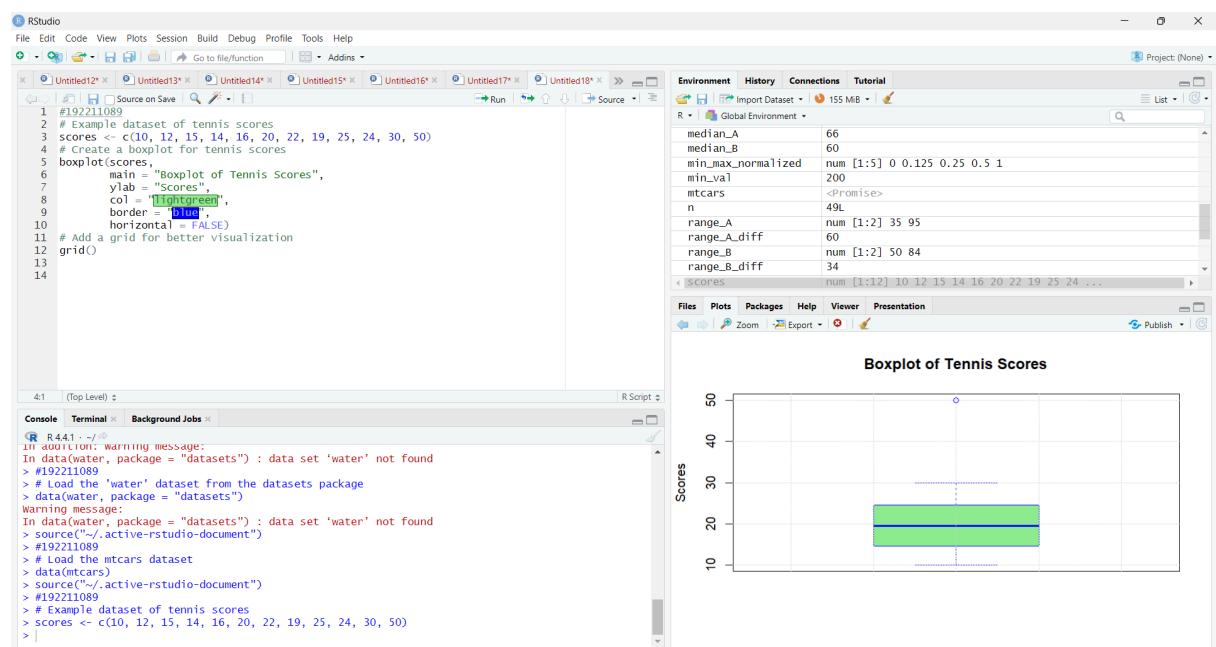
8.Create a Boxplot graph for the relation between "mpg"(miles per gallon) and "cyl"(number of Cylinders) for the dataset "mtcars" available in R Environment

Code:



9. Assume the Tennis coach wants to determine if any of his team players are scoring outliers. To visualize the distribution of points scored by his players, then how can he decide to develop the box plot? Give a suitable example using the Boxplot visualization technique.

Code:



10. Implement using R language in which age group of people are affected by blood pressure based on the diabetes dataset show it using scatterplot and bar chart (that is BloodPressure vs Age using dataset “diabetes.csv”)

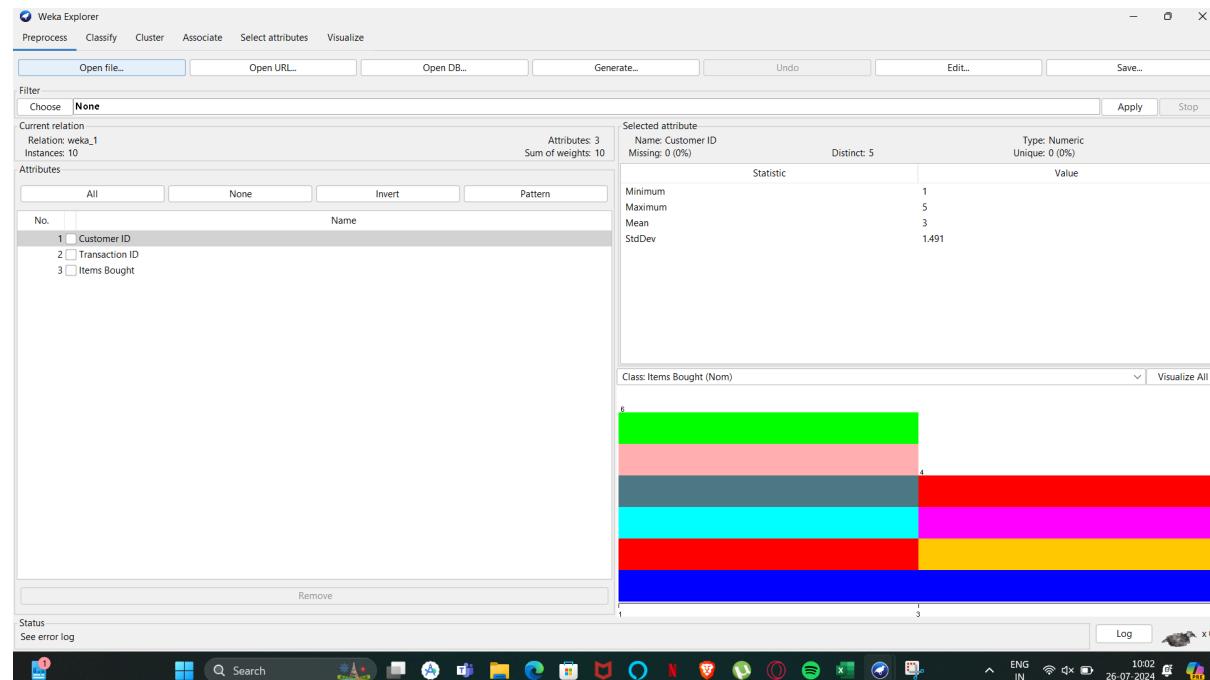
Code:

The screenshot shows the RStudio interface with several windows open:

- Code Editor:** Displays R code for reading the 'diabetes' dataset and creating a scatterplot. The scatterplot shows Blood Pressure vs Age.
- Environment:** Shows the global environment with objects like 'diabetes' (768 obs. of 10 variables), 'mtcars' (32 obs. of 11 variables), and various data frames and functions.
- Plots:** Displays two bar charts:
 - An average blood pressure bar chart by age group (20-30, 30-40, 40-50, 50-60, 60-70, 70-80, 80-90).
 - A second bar chart for 'Average Blood Pressure by Age Group'.
- Console:** Shows the R session history, including the execution of the R code from the Code Editor.

Day - 3

1. Consider the data set and perform the Apriori Algorithm and FP algorithm
support:3 and confidence=50%



Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associator

Choose: **FilteredAssociator** -F "weka.filters.MultiFilter -F "\weka.filters.unsupervised.attribute.ReplaceMissingValues \"-S 1\" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

Start Stop

Result list (right-click for op...)

100127 - FilteredAssociator

```
== Run information ==
Scheme: weka.associations.FilteredAssociator -F "weka.filters.MultiFilter -F "\weka.filters.unsupervised.attribute.ReplaceMissingValues \"-S 1\" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation: weka_1
Instances: 10
Attributes: 3
Customer ID
Transaction ID
Items Bought
```

Status See error log Log x 0

2.

Transaction ID	Items Bought
1	{Milk, Beer, Diapers}
2	{Bread, Butter, Milk}
3	{Milk, Diapers, Cookies}
4	{Bread, Butter, Cookies}
5	{Beer, Cookies, Diapers}
6	{Milk, Diapers, Bread, Butter}
7	{Bread, Butter, Diapers}
8	{Beer, Diapers}
9	{Milk, Diapers, Bread, Butter}
10	{Beer, Cookies}

Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associator

Choose: **FilteredAssociator** -F "weka.filters.MultiFilter -F "\weka.filters.unsupervised.attribute.ReplaceMissingValues \"-S 1\" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

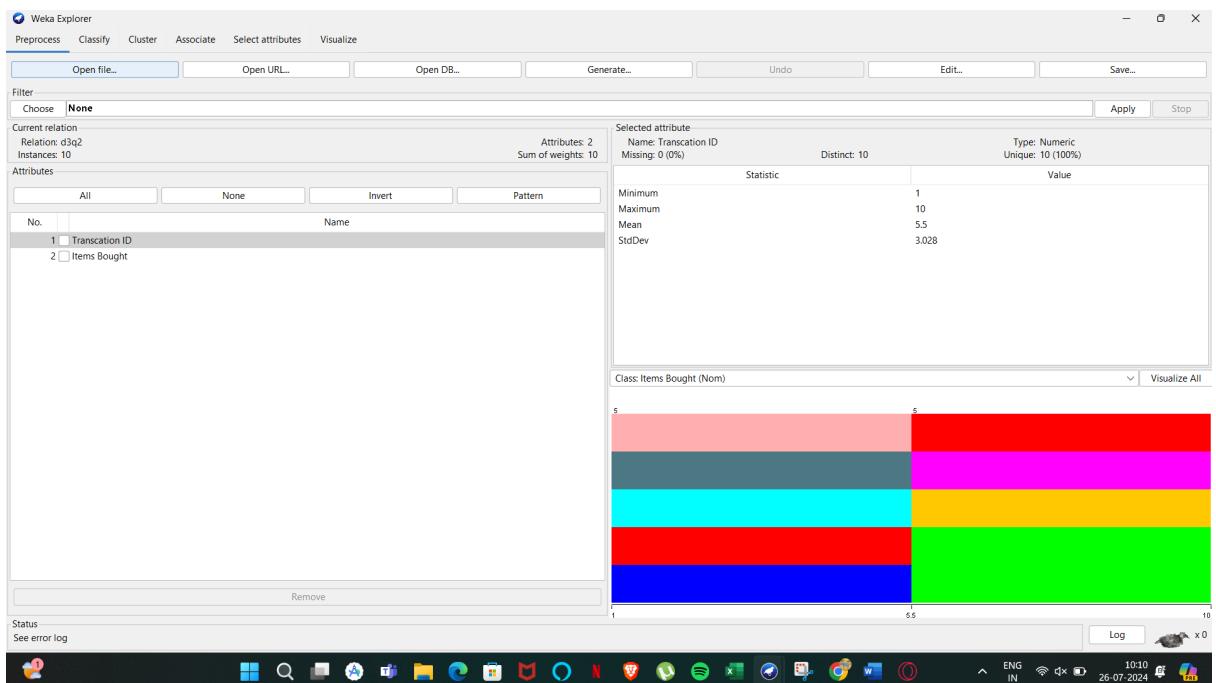
Start Stop

Result list (right-click for op...)

100954 - FilteredAssociator

```
== Run information ==
Scheme: weka.associations.FilteredAssociator -F "weka.filters.MultiFilter -F "\weka.filters.unsupervised.attribute.ReplaceMissingValues \"-S 1\" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation: d3g2
Instances: 10
Attributes: 2
Transaction ID
Items Bought
```

Status See error log Log x 0



3.

RID	age	income	student	credit_rating	Class: buys_computer
1	<=30	high	no	fair	no
2	<=30	high	no	excellent	no
3	31 ... 40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	31 ... 40	low	yes	excellent	yes
8	<=30	medium	no	fair	no
9	<=30	low	yes	fair	yes
10	>40	medium	yes	fair	yes
11	<=30	medium	yes	excellent	yes
12	31 ... 40	medium	no	excellent	yes
13	31 ... 40	high	yes	fair	yes
14	>40	medium	no	excellent	no

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose **DecisionStump**

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds 10
- Percentage split % 66
- [More options...](#)

(Nom) classbuys_computer

Start Stop

Result list (right-click for options)

10:12:57 - trees.DecisionStump

```
0.0 1.0
age != 31-40
no yes
0.4 0.6
age is missing
no yes
0.2857142857142857 0.7142857142857143
```

Time taken to build model: 0 seconds

==== Stratified cross-validation ====

==== Summary ====

	Correctly Classified Instances	10	71.4286 %
Incorrectly Classified Instances	4	28.5714 %	
Kappa statistic	0		
Mean absolute error	0.4751		
Root mean squared error	0.5717		
Relative absolute error	110.8466 %		
Root relative squared error	122.814 %		
Total Number of Instances	14		

==== Detailed Accuracy By Class ====

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	FRC Area	Class
0.000	0.000	?	0.000	?	?	?	0.200	0.243	no
1.000	1.000	0.714	1.000	0.833	?	?	0.200	0.613	yes
Weighted Avg.	0.714	0.714	?	0.714	?	?	0.200	0.507	

==== Confusion Matrix ====

```
a b <-- classified as
0 4 | a = no
0 10 | b = yes
```

Status OK

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose **NaiveBayes**

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds 10
- Percentage split % 66
- [More options...](#)

(Nom) classbuys_computer

Start Stop

Result list (right-click for options)

10:12:57 - trees.DecisionStump

10:13:40 - bayes.NaiveBayes

```
[total] 6.0 12.0
credit_rating
fair 3.0 7.0
excellent 3.0 5.0
[total] 6.0 12.0
```

Time taken to build model: 0 seconds

==== Stratified cross-validation ====

==== Summary ====

	Correctly Classified Instances	7	50 %
Incorrectly Classified Instances	7	50 %	
Kappa statistic	-0.3243		
Mean absolute error	0.5627		
Root mean squared error	0.6361		
Relative absolute error	131.5352 %		
Root relative squared error	136.6459 %		
Total Number of Instances	14		

==== Detailed Accuracy By Class ====

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	FRC Area	Class
0.000	0.300	0.000	0.000	0.000	-0.330	0.025	0.196	0.196	no
0.700	1.000	0.636	0.700	0.667	-0.330	0.025	0.538	0.538	yes
Weighted Avg.	0.500	0.800	0.455	0.500	0.476	-0.330	0.025	0.440	

==== Confusion Matrix ====

```
a b <-- classified as
0 4 | a = no
3 7 | b = yes
```

Status OK

4.Analysis the dataset “diabetes.csv” how the diabetes trend is for different age people, using linear regression and multiple regression.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose **Logistic -R 1.0E-6 -M -1 -num-decimal-places 4**

Test options
 Use training set
 Supplied test set Set...
 Cross-validation Folds **10**
 Percentage split % **66**
 More options...

(Nom) class Start Stop

Result list (right-click for options)
 10:17:30 - functions.Logistic

```

Time taken to build model: 0.05 seconds
==== stratified cross-validation ====
==== Summary ====
Correctly Classified Instances      593           77.2135 %
Incorrectly Classified Instances   175           22.7865 %
Kappa statistic                   0.4734
Mean absolute error               0.3094
Root mean squared error          0.3954
Relative absolute error           68.0818 %
Root relative squared error      82.9651 %
Total Number of Instances        768

==== Detailed Accuracy By Class ====


|               | Test Rate | FP Rate | Precision | Recall | F-Measure | MCC   | ROC Area | ERG Area | Class           |
|---------------|-----------|---------|-----------|--------|-----------|-------|----------|----------|-----------------|
|               | 0.880     | 0.489   | 0.793     | 0.880  | 0.834     | 0.480 | 0.632    | 0.692    | tested_negative |
|               | 0.571     | 0.120   | 0.718     | 0.571  | 0.636     | 0.480 | 0.632    | 0.715    | tested_positive |
| Weighted Avg. | 0.772     | 0.321   | 0.767     | 0.772  | 0.765     | 0.480 | 0.632    | 0.691    |                 |


==== Confusion Matrix ====


|     |  | a   | b   | <-- classified as |
|-----|--|-----|-----|-------------------|
|     |  | a   | b   | tested_negative   |
| 115 |  | 153 | 440 | 60                |
|     |  |     |     | tested_positive   |


```

Status OK Log x 0

Windows Taskbar: File Explorer, Edge, File Manager, Task View, Taskbar settings, Date/Time: 10:17:30, 26-07-2024

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose **MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a**

Test options
 Use training set
 Supplied test set Set...
 Cross-validation Folds **10**
 Percentage split % **66**
 More options...

(Nom) class Start Stop

Result list (right-click for options)
 10:18:10 - functions.MultilayerPerceptron

```

Building model for fold 4...
Time taken to build model: 0.73 seconds
==== Classifier output ====
Inputs          Weights
Threshold      -3.3787476603258035
Attrib insu     -3.0955093615541673
Attrib mass     -8.77502662333566
Attrib pedi     -5.1923601596094315
Attrib age      9.269528793152954
Sigmoid Node 5
Inputs          Weights
Threshold      -3.3787476603258035
Attrib preg     9.1820154610085302
Attrib plas     -12.442913800448132
Attrib insu     5.07804474752846
Attrib skin     -0.06085501320155377
Attrib insu     2.3070185070106217
Attrib mass     -5.22208001635601
Attrib pedi     -0.7354842913650297
Attrib age     -19.2656337017977
Sigmoid Node 6
Inputs          Weights
Threshold      0.05437383478947769
Attrib preg     12.836762781789405
Attrib plas     -6.062276016682751
Attrib pres     -1.3896840458164672
Attrib skin     0.34997008402063716
Attrib insu     -2.2194040147264896
Attrib mass     -0.9589656235929952
Attrib pedi     6.090003751353246
Attrib age     -8.833262465394625
Class tested_negative
Input          Node 0
Class tested_positive
Input          Node 1

==== Detailed Accuracy By Class ====


|               | Test Rate | FP Rate | Precision | Recall | F-Measure | MCC   | ROC Area | ERG Area | Class           |
|---------------|-----------|---------|-----------|--------|-----------|-------|----------|----------|-----------------|
|               | 0.880     | 0.489   | 0.793     | 0.880  | 0.834     | 0.480 | 0.632    | 0.692    | tested_negative |
|               | 0.571     | 0.120   | 0.718     | 0.571  | 0.636     | 0.480 | 0.632    | 0.715    | tested_positive |
| Weighted Avg. | 0.772     | 0.321   | 0.767     | 0.772  | 0.765     | 0.480 | 0.632    | 0.691    |                 |


==== Confusion Matrix ====


|     |  | a   | b   | <-- classified as |
|-----|--|-----|-----|-------------------|
|     |  | a   | b   | tested_negative   |
| 115 |  | 153 | 440 | 60                |
|     |  |     |     | tested_positive   |


```

Status Building model for fold 4... Log x 1

Windows Taskbar: File Explorer, Edge, File Manager, Task View, Taskbar settings, Date/Time: 10:18:10, 26-07-2024

5.Implement using WEKA for the given Suppose a database has five transactions. Let min sup= 50%(2) and min con f = 80%.

Transactions	Items
T1	(M, O, N, K, E, Y)

T2 **(D, O, N, K, E, Y)**

T3 **(M, A, K, E)**

T4 **(M, U, C, K, Y)**

T5 **(C, O, O, K, I, E)**

- **Find all frequent item sets using Apriori algorithm**
- **Also draw FP-Growth Tree**

Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associate

Choose **Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1**

Result list (right-click for ...)

10:20:35 - Apriori

Start Stop

Associate output

```
Schema: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1
Relation: d3q5B
Instances: 5
Attributes: 2
Transaction
items
==== Associator model (full training set) ====

Apriori
=====

Minimum support: 0.3 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10
Size of set of large itemsets L(2): 5

Best rules found:

1. items=M,O,N,K,E,Y 1 ==> Transaction=t1 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
2. Transaction=t1 1 ==> items=M,O,N,K,E,Y 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
3. items=D,O,N,K,E,Y 1 ==> Transaction=t2 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
4. Transaction=t2 1 ==> items=D,O,N,K,E,Y 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
5. items=M,A,K,E 1 ==> Transaction=t3 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
6. Transaction=t3 1 ==> items=M,A,K,E 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
7. items=M,U,C,K,Y 1 ==> Transaction=t4 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
8. Transaction=t4 1 ==> items=M,U,C,K,Y 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
9. items=C,O,O,K,I,E 1 ==> Transaction=t5 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
10. Transaction=t5 1 ==> items=C,O,O,K,I,E 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
```

Status OK Log x0

Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associate

Choose **FilteredAssociator -F "weka.filters.MultiFilter -F "weka.filters.unsupervised.attribute.ReplaceMissingValues \"-S 1" <-1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1**

Result list (right-click for op...)

10:20:35 - Apriori

10:21:05 - FilteredAssociator

Start Stop

Associate output

```
@relation d3q5-weka.filters.unsupervised.attribute.ReplaceMissingValues-weka.filters.MultiFilter-Weka.filters.unsupervised.attribute.ReplaceMissingValues-S1
@attribute Transaction {t1,t2,t3,t4,t5}
@attribute items {"M,O,N,K,E,Y","D,O,N,K,E,Y","M,A,K,E","M,U,C,K,Y","C,O,O,K,I,E"}

#data

Associate Model

Apriori
=====

Minimum support: 0.3 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14

Generated sets of large itemsets:

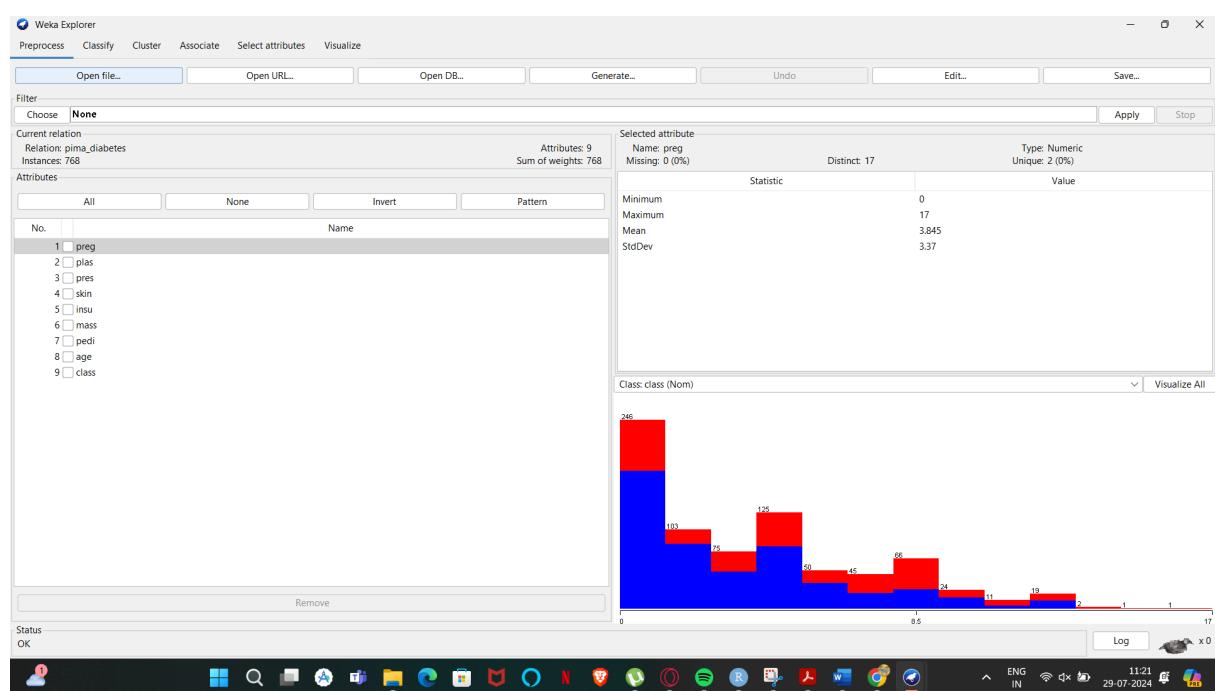
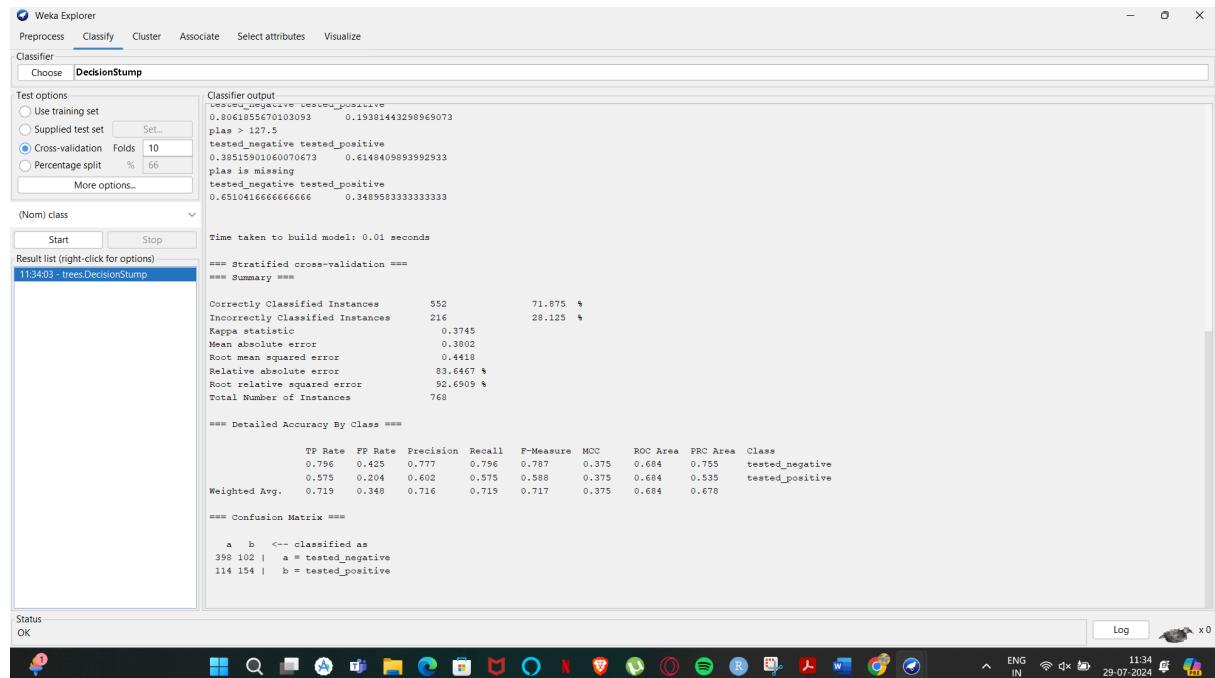
Size of set of large itemsets L(1): 10
Size of set of large itemsets L(2): 5

Best rules found:

1. items=M,O,N,K,E,Y ==> Transaction=t1 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
2. Transaction=t1 1 ==> items=M,O,N,K,E,Y 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
3. items=D,O,N,K,E,Y 1 ==> Transaction=t2 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
4. Transaction=t2 1 ==> items=D,O,N,K,E,Y 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
5. items=M,A,K,E 1 ==> Transaction=t3 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
6. Transaction=t3 1 ==> items=M,A,K,E 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
7. items=M,U,C,K,Y 1 ==> Transaction=t4 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
8. Transaction=t4 1 ==> items=M,U,C,K,Y 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
9. items=C,O,O,K,I,E 1 ==> Transaction=t5 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
10. Transaction=t5 1 ==> items=C,O,O,K,I,E 1 <conf:(1)> lift:(5) lev:(0.16) [0] conv:(0.8)
```

Status OK Log x0

6. Prediction of Categorical Data using Decision Tree Algorithm through WEKA using any datasets. a) Tree b) Preprocess c) Logistic



```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier
Choose Logistic -R 1.0E-8 -M 1 -num-decimal-places 4
Test options
○ Use training set
○ Supplied test set Set...
● Cross-validation Folds 10
○ Percentage split % 66
More options...
(Nom) class
Start Stop
Result list (right-click for options)
113403 - trees.DecisionStump
113747 - functions.Logistic

Classifier output
Time taken to build model: 0.03 seconds
==== Stratified cross-validation ====
==== Summary ====
Correctly Classified Instances 593 77.2135 %
Incorrectly Classified Instances 175 22.7865 %
Kappa statistic 0.4734
Mean absolute error 0.3094
Root mean squared error 0.3954
Relative absolute error 68.0818 %
Root relative squared error 82.9651 %
Total Number of Instances 768

==== Detailed Accuracy By Class ====
      TP Rate FP Rate Precision Recall F-Measure MCC ROC Area FRC Area Class
0.880 0.425 0.793 0.880 0.834 0.480 0.832 0.892 tested_negative
0.571 0.120 0.718 0.571 0.636 0.480 0.832 0.715 tested_positive
Weighted Avg. 0.772 0.321 0.767 0.772 0.765 0.480 0.832 0.831

==== Confusion Matrix ====
a b <-- classified as
440 60 | a = tested_negative
115 153 | b = tested_positive

```

7.Create the dataset using ARFF file format:

- a.Find the frequent itemsets and generate association rules on this. Assume that minimum support threshold (s = 33.33%) and minimum confident threshold (c =60%).
 b.List the various rule generated by apriori and FP tree algorithim ,mention wheather accepted or rejected.

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Associate
Choose Apriori -N 0 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1
Start Stop
Result list (right-click for options)
192416 - Apron

Associate output
breast:
breast-quad
irradiat
Class
==== Associate model (full training set) ====

Apriori
=====
Minimum support: 0.5 (143 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 10

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6
Size of set of large itemsets L(2): 6
Size of set of large itemsets L(3): 4
Size of set of large itemsets L(4): 1

Best rules found:

1. inv-nodes=0=2 irradiat=no Class=no-recurrence-events 147 ==> node-caps=no 145 <conf:(0.59)> lift:(1.27) lev:(0.11) [30] conv:(10.97)
2. inv-nodes=0=2 irradiat=no 183 ==> node-caps=no 177 <conf:(0.57)> lift:(1.25) lev:(0.12) [34] conv:(5.85)
3. node-caps=no irradiat=no Class=no-recurrence-events 151 ==> inv-nodes=0=2 145 <conf:(0.56)> lift:(1.29) lev:(0.11) [32] conv:(5.51)
4. inv-nodes=0=2 Class=no-recurrence-events 167 ==> node-caps=no 160 <conf:(0.56)> lift:(1.23) lev:(0.11) [30] conv:(4.67)
5. inv-nodes=0=2 213 ==> node-caps=no 201 <conf:(0.54)> lift:(1.22) lev:(0.12) [35] conv:(3.67)
6. node-caps=no irradiat=no 188 ==> inv-nodes=0=2 177 <conf:(0.54)> lift:(1.26) lev:(0.13) [36] conv:(4.4)
7. node-caps=no Class=no-recurrence-events 171 ==> inv-nodes=0=2 160 <conf:(0.54)> lift:(1.26) lev:(0.11) [32] conv:(3.64)
8. irradiat=no Class=no-recurrence-events 164 ==> node-caps=no 151 <conf:(0.52)> lift:(1.19) lev:(0.08) [23] conv:(2.62)
9. inv-nodes=0=2 node-caps=no Class=no-recurrence-events 164 ==> irradiat=no 145 <conf:(0.51)> lift:(1.19) lev:(0.08) [23] conv:(2.38)
10. node-caps=no 222 ==> inv-nodes=0=2 201 <conf:(0.51)> lift:(1.22) lev:(0.12) [35] conv:(2.58)


```

The screenshot shows the Weka Explorer interface with the 'Associate' tab selected. The command line at the top reads: 'Choose FilteredAssociator -F "weka.filters.MultiFilter -F "weka.filters.unsupervised.attribute.ReplaceMissingValues V" -S 1" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1'. The main window displays the 'Associator output' pane, which contains the results of the Apriori algorithm. It shows the model configuration ('Apriori', 'Minimum support: 0.35 (100 instances)', 'Minimum metric <confidence>: 0.9', 'Number of cycles performed: 13'), generated itemsets (L(1) to L(4)), and the best rules found. The rules listed include various combinations of attributes like 'inv-nodes', 'breast', 'irradiat', 'node-caps', etc., with their respective confidence, lift, and conviction values.

```

@data
#data

Associator Model

Apriori
=====
Minimum support: 0.35 (100 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 13

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10
Size of set of large itemsets L(2): 20
Size of set of large itemsets L(3): 8
Size of set of large itemsets L(4): 2

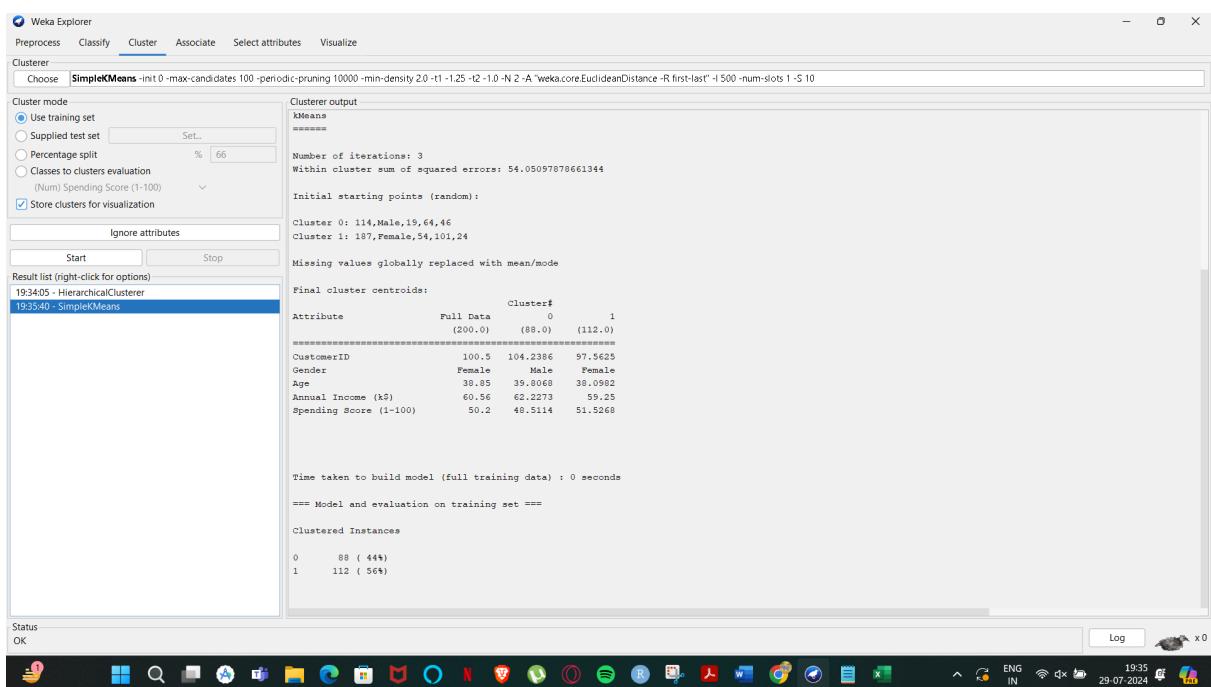
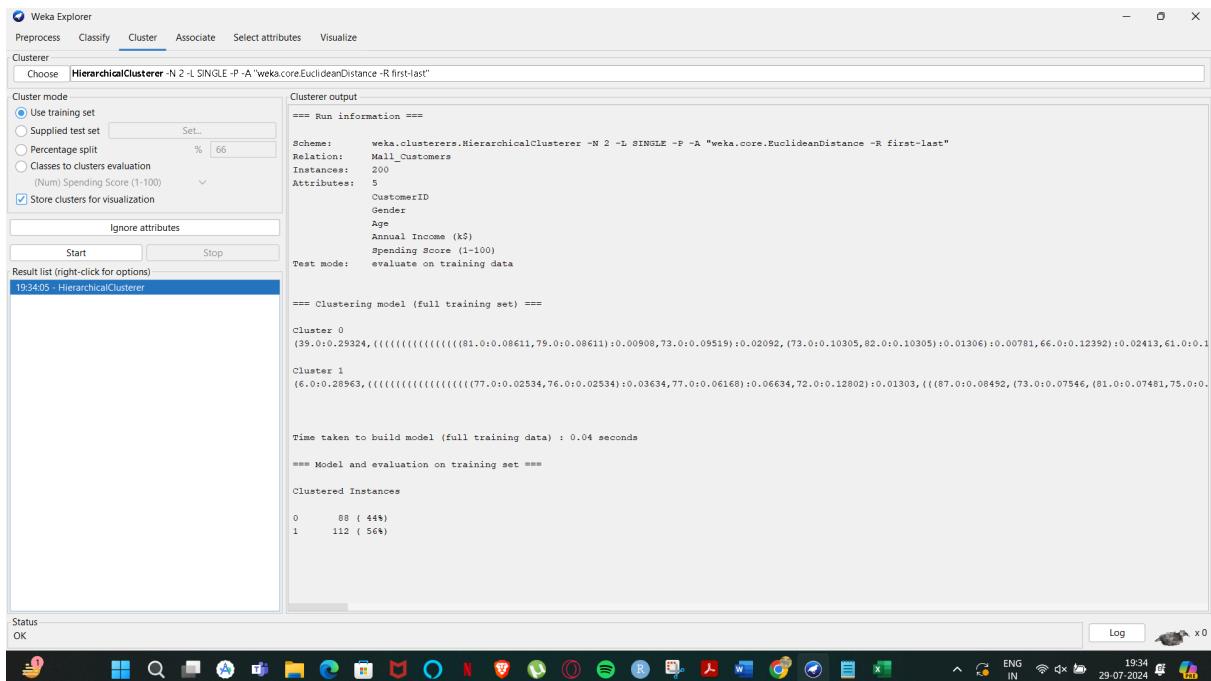
Best rules found:

1. inv-nodes=0-2 breast=left irradiat=no 101 ==> node-caps=no 100 <conf:(0.99)> lift:(1.23) lev:(0.07) [18] conv:(9.89)
2. inv-nodes=0-2 irradiat=no Class=no-recurrence-events 147 ==> node-caps=no 145 <conf:(0.59)> lift:(1.23) lev:(0.09) [26] conv:(9.55)
3. inv-nodes=0-2 breast=left 115 ==> node-caps=no 113 <conf:(0.58)> lift:(1.22) lev:(0.07) [20] conv:(7.51)
4. inv-nodes=0-2 irradiat=no 183 ==> node-caps=no 179 <conf:(0.88)> lift:(1.22) lev:(0.11) [31] conv:(7.17)
5. inv-nodes=0-2 Class=no-recurrence-events 167 ==> node-caps=no 161 <conf:(0.96)> lift:(1.2) lev:(0.09) [26] conv:(4.67)
6. node-caps=no irradiat=left irradiat=no 104 ==> inv-nodes=0-2 100 <conf:(0.96)> lift:(1.29) lev:(0.08) [22] conv:(5.31)
7. node-caps=no irradiat=no Class=no-recurrence-events 151 ==> inv-nodes=0-2 145 <conf:(0.96)> lift:(1.29) lev:(0.11) [32] conv:(5.51)
8. inv-nodes=0-2 213 ==> node-caps=no 204 <conf:(0.96)> lift:(1.19) lev:(0.11) [32] conv:(4.17)
9. menopause=prenmeno inv-nodes=0-2 112 ==> node-caps=no 106 <conf:(0.95)> lift:(1.18) lev:(0.06) [15] conv:(3.13)
10. node-caps=no irradiat=no 190 ==> inv-nodes=0-2 179 <conf:(0.94)> lift:(1.26) lev:(0.13) [37] conv:(4.04)

```

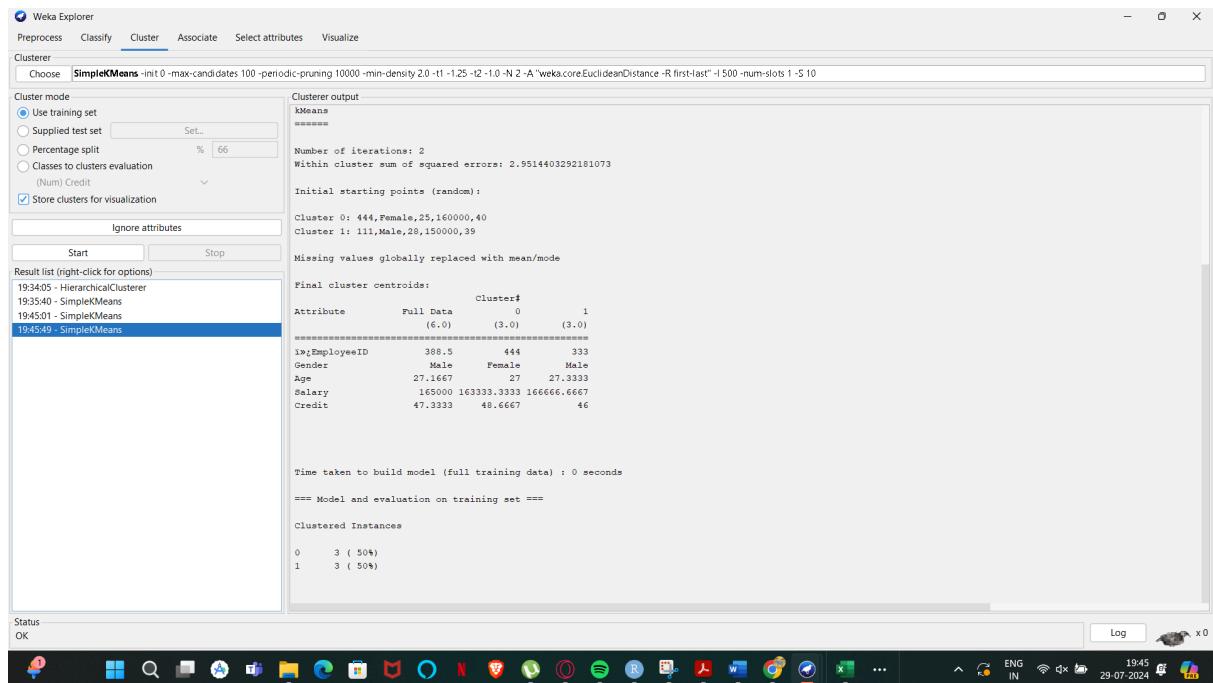
Day - 4

1. Consider that you are owning a supermarket mall and through membership cards, you have some basic data about your customers like Customer ID, age, gender, annual income and spending score. For the above scenario, the Problem Statement was You want to understand the customers who can easily converge [Target Customers] so that the data can be given to the marketing team and plan the strategy accordingly. For the above scenario prepare a dataset and perform Clustering Analysis to segment the customers in the Mall. There are clearly Five segments of Customers based on their Annual Income and Spending Score namely *Usual Customers, Priority Customers, Senior Citizen Target Customers, and Young Target Customers*.

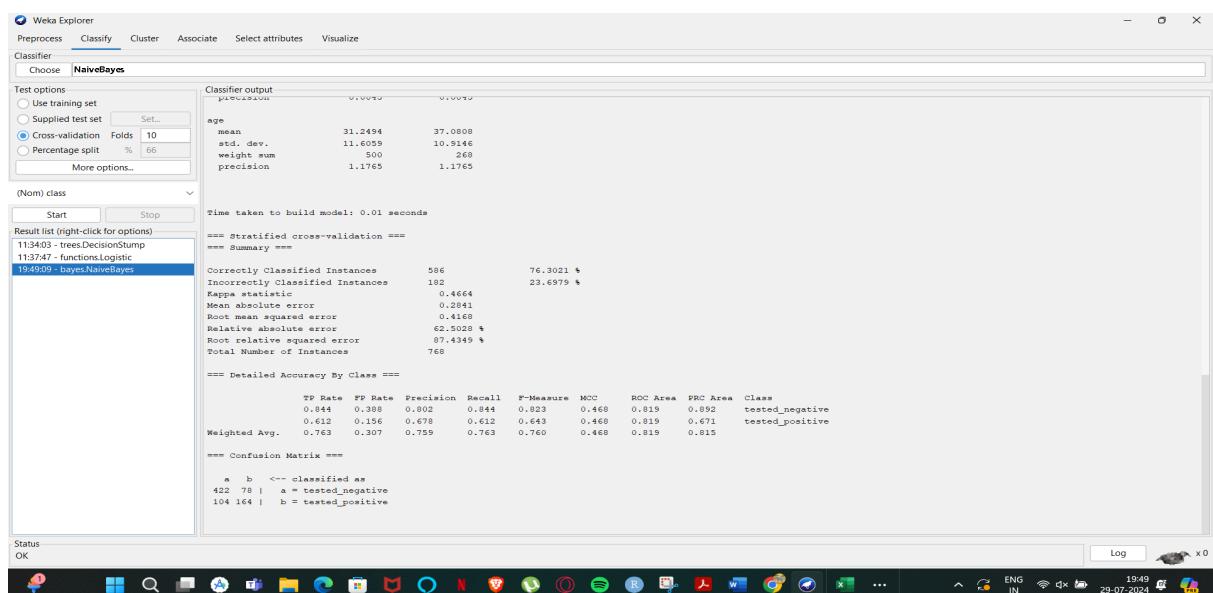


2. Create the following dataset using CSV file format. To perform cluster analysis using K-Means in WEKA. To change the cluster size and plot the graph and illustrate the visualization of cluster.

<u>EmployeeID</u>	Gender	Age	Salary	Credit
111	Male	28	150000	39
222	Male	25	150000	27
333	Female	26	160000	42
444	Female	25	160000	40
555	Female	30	170000	64
666	Male	29	200000	72



3.Prediction of categorical data using Naïve Bayes classification through WEKA using any datasets. Compare the Naïve Bayes algorithm with SVM using the summary of results given by the classifiers and plot the graph.



```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1
Test options
 Use training set
 Supplied test set Set...
 Cross-validation Folds 10
 Percentage split % 66
More options...
(Nom) class Start Stop
Result list (right-click for options)
113403 - treesDecisionStump
113747 - functionsLogistic
194909 - bayesNaiveBayes
195021 - treesRandomForest
Classifier output
test mode: 10-fold cross-validation
==== Classifier model (full training set) ====
RandomForest
Bagging with 100 iterations and base learner
weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities
Time taken to build model: 0.22 seconds
==== Stratified cross-validation ====
==== Summary ====
Correctly Classified Instances 582 75.7813 %
Incorrectly Classified Instances 186 24.2188 %
Kappa statistic 0.4866
Mean absolute error 0.3106
Root mean squared error 0.4031
Relative absolute error 68.3405 %
Root relative squared error 84.5604 %
Total Number of Instances 768
==== Detailed Accuracy By Class ====


|               | TP    | Rate  | FP    | Rate  | Precision | Recall | F-Measure | MCC   | ROC Area | PRC Area | Class           |
|---------------|-------|-------|-------|-------|-----------|--------|-----------|-------|----------|----------|-----------------|
| 0             | 0.836 | 0.388 | 0.801 | 0.836 | 0.818     | 0.458  | 0.820     | 0.886 | 0.820    | 0.886    | tested_negative |
| 1             | 0.612 | 0.164 | 0.667 | 0.612 | 0.638     | 0.458  | 0.820     | 0.679 | 0.820    | 0.679    | tested_positive |
| Weighted Avg. | 0.758 | 0.310 | 0.754 | 0.758 | 0.755     | 0.458  | 0.820     | 0.814 |          |          |                 |

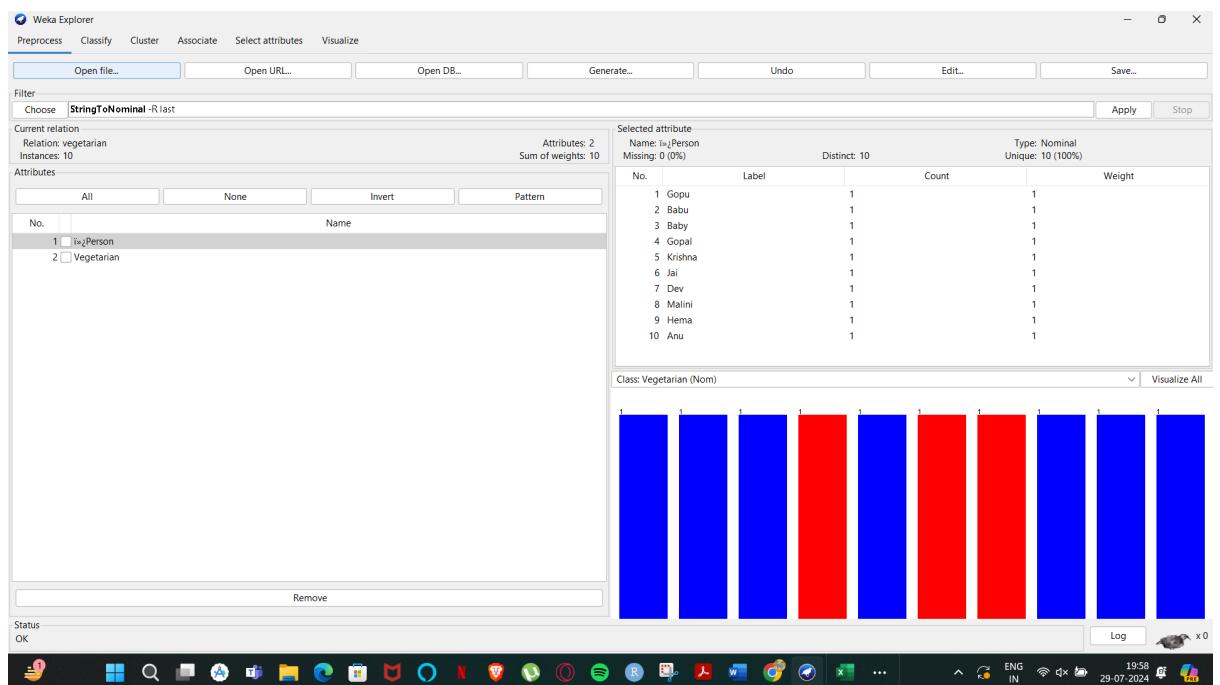

==== Confusion Matrix ====


|   |     | a   | b | <-- classified as   |
|---|-----|-----|---|---------------------|
|   |     | 1   | 0 | a = tested_negative |
|   |     | 1   | 0 | b = tested_positive |
| 1 | 418 | 82  | 1 | a = tested_negative |
| 0 | 104 | 164 | 1 | b = tested_positive |


```

4. The following list of persons with vegetarian or not details is given in the table. How will you find out how many of them are vegetarian and how many of them are non-vegetarian? Which type of the person's total count has greater value

Person	Gopu	Babu	Baby	Gopal	Krishna	Jai	Dev	Malini	Hema	Anu
Vegetarian	yes	yes	yes	no	yes	no	no	yes	yes	yes



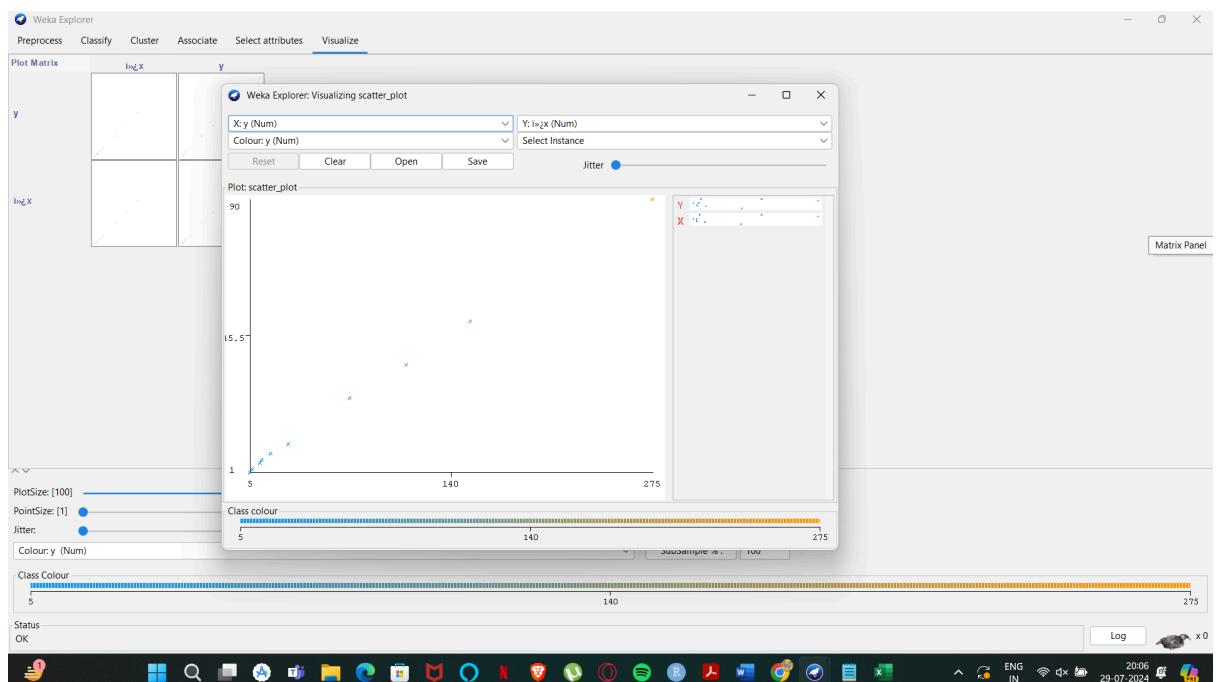
```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1
Test options
○ Use training set
○ Supplied test set Set...
● Cross-validation Folds 10
○ Percentage split % 66
More options...
(Nom) Vegetarian
Start Stop
Result list (right-click for options)
113403 - tree.DecisionStump
113747 - functions.Logistic
194099 - bayes.NaiveBayes
195021 - trees.RandomForest
195801 - trees.RandomForest
Classifier output
weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities
Time taken to build model: 0 seconds
==== Stratified cross-validation ====
==== Summary ====
Correctly Classified Instances 7 70 %
Incorrectly Classified Instances 3 30 %
Kappa statistic 0
Mean absolute error 0.4447
Root mean squared error 0.4942
Relative absolute error 94.0641 %
Root relative squared error 98.5938 %
Total Number of Instances 10
==== Detailed Accuracy By Class ====
          TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRG Area Class
1.000   1.000   0.700   1.000   0.824 ? 0.286 0.609 yes
0.000   0.000   ? 0.000   ? ? 0.286 0.341 no
Weighted Avg. 0.700 0.700 ? 0.700 ? ? 0.286 0.529
==== Confusion Matrix ====
a b <-- classified as
7 0 | a = yes
3 0 | b = no

```

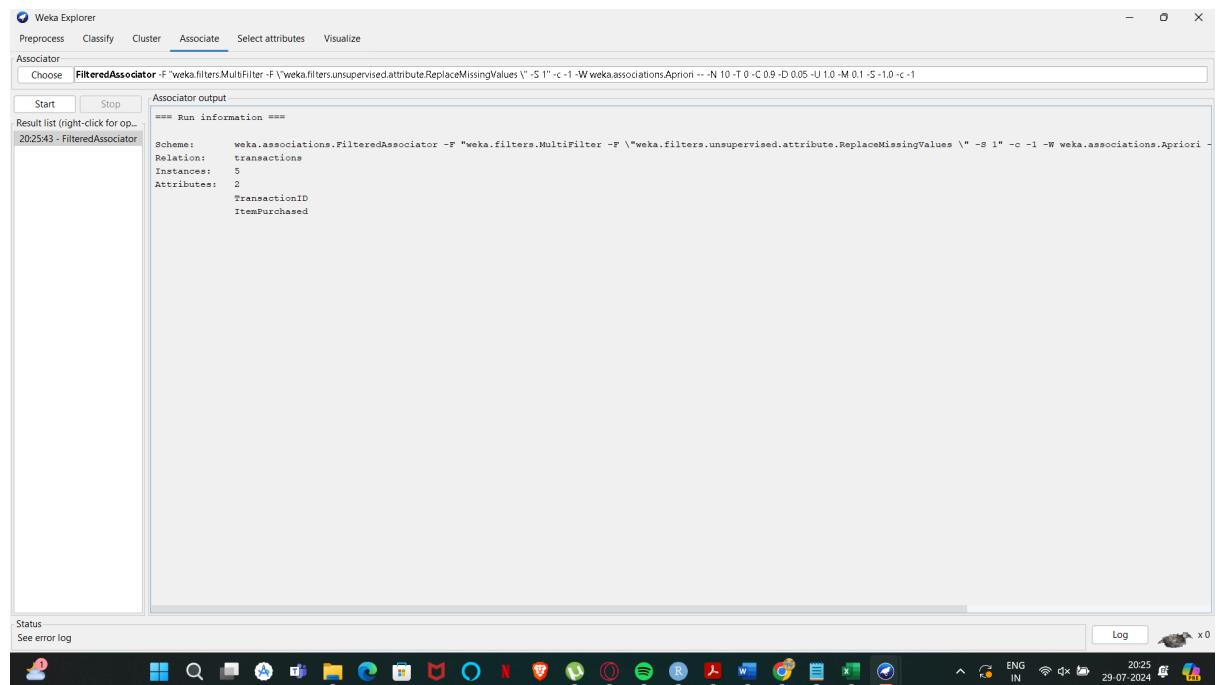
5. The following table would be plotted as (x,y) points, with the first column being the x values as number of mobile phones sold and the second column being the y values as money. To use the scatter plot for how many mobile phones sold.

x	4	1	5	7	10	2	50	25	90	36
y	12	5	13	19	31	7	153	72	275	110



6. Generate rules using FP growth algorithm using the given dataset which has the following transactions with items purchased: Consider the values as support=50% and confidence=75%.

Transaction ID	Items Purchased
1	Bread, Cheese, Egg, Juice
2	Bread, Cheese, Juice
3	Bread, Milk, Yogurt
4	Bread, Juice, Milk
5	Cheese, Juice, Milk



7. Prediction of Diabetes Data using Decision tree classifier in WEKA. Compare it with Support Vector Machine classifier. Show the result accuracy and F1 measure calculation .Plot the graph and explain the summary of results.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose **DecisionStump**

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds **10**
- Percentage split % **66**
- [More options...](#)

(Nom) class

Start Stop

Result list (right-click for options)

20:29:54 - trees.DecisionStump

```
Classifier output:
=====  
weka.classifiers.trees.DecisionStump  
0.8061555670103093 0.19381443258969073  
plus > 127.5  
tested_negative tested_positive  
0.38515901060070673 0.6148409893992933  
plus is missing  
tested_negative tested_positive  
0.6510416666666666 0.3489583333333333  
=====  
Time taken to build model: 0.01 seconds  
=====  
Stratified cross-validation ====  
==== Summary ====  
  
Correctly Classified Instances 552 71.875 %  
Incorrectly Classified Instances 216 28.125 %  
Kappa statistic 0.3745  
Mean absolute error 0.3802  
Root mean squared error 0.4418  
Relative absolute error 82.6467 %  
Root relative squared error 92.6909 %  
Total Number of Instances 768  
  
==== Detailed Accuracy By Class ====  
  
 TP Rate FP Rate Precision Recall F-Measure MCC ROC Area FRC Area Class  
 0.796 0.425 0.777 0.796 0.787 0.375 0.684 0.755 tested_negative  
 0.575 0.204 0.602 0.575 0.588 0.375 0.684 0.535 tested_positive  
Weighted Avg. 0.719 0.348 0.716 0.719 0.717 0.375 0.684 0.678  
  
==== Confusion Matrix ====  
  
 a b <-- classified as  
 398 102 | a = tested_negative  
 114 154 | b = tested_positive  
=====  
Status OK
```

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose **RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1**

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds **10**
- Percentage split % **66**
- [More options...](#)

(Nom) class

Start Stop

Result list (right-click for options)

20:29:54 - trees.DecisionStump

20:31:43 - trees.RandomForest

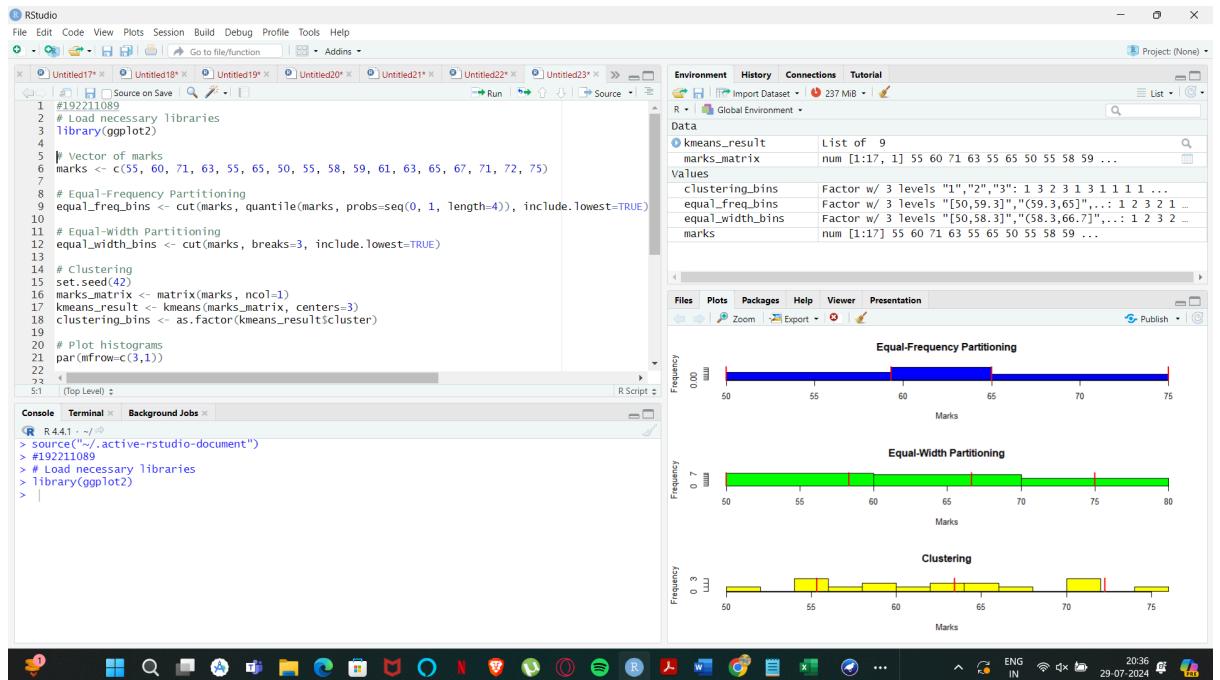
```
Classifier output:
=====  
weka.classifiers.trees.DecisionStump  
=====  
weka.classifiers.trees.RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities  
=====  
Time taken to build model: 0.22 seconds  
=====  
Stratified cross-validation ====  
==== Summary ====  
  
Correctly Classified Instances 582 75.7813 %  
Incorrectly Classified Instances 186 24.2108 %  
Kappa statistic 0.4566  
Mean absolute error 0.3106  
Root mean squared error 0.4031  
Relative absolute error 68.3405 %  
Root relative squared error 84.5604 %  
Total Number of Instances 768  
  
==== Detailed Accuracy By Class ====  
  
 TP Rate FP Rate Precision Recall F-Measure MCC ROC Area FRC Area Class  
 0.836 0.388 0.801 0.836 0.818 0.458 0.820 0.886 tested_negative  
 0.612 0.164 0.667 0.612 0.638 0.458 0.820 0.679 tested_positive  
Weighted Avg. 0.758 0.310 0.754 0.758 0.755 0.458 0.820 0.814  
  
==== Confusion Matrix ====  
  
 a b <-- classified as  
 419 82 | a = tested_negative  
 104 164 | b = tested_positive  
=====  
Status OK
```

8.Implement of the R script using marks scored by a student in his model exam has been sorted as follows: 55, 60, 71, 63, 55, 65, 50, 55,58,59,61,63,65,67,71,72,75. Partition them into three bins by each of the following methods. Plot the data points using histogram.

(a) equal-frequency (equi-depth) partitioning

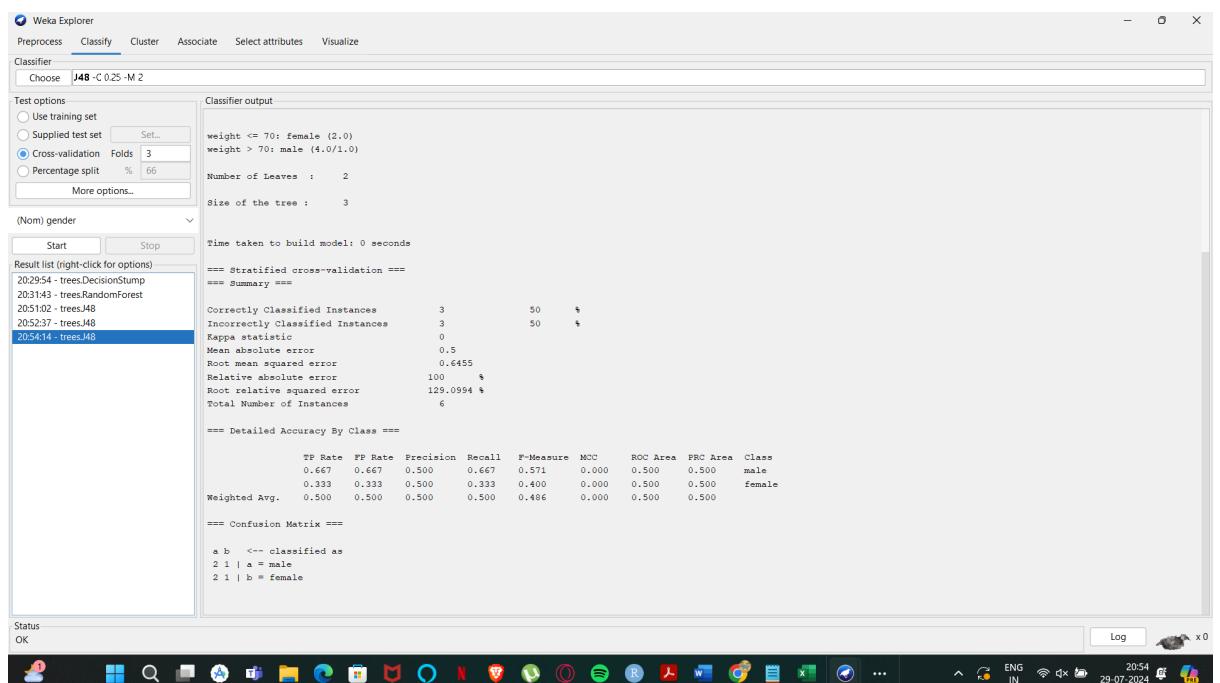
(b) equal-width partitioning

(c) clustering



9.Consider this Decision tree :

- create the data set for the below tree using ARFF format and calculate accuracy and decision for the same
- Using this decision tree generate the rules based on rule based induction.
- Compare both the algorithms and plot the confusion matrix.



10. Create an ARFF file for the table below and implement for the Apriori Algorithm and FP growth algorithm and compare the rules generated by both the algorithms. Identify the unique rules generated by the above algorithms.

NOTE: Assume Min_sup=2 and confidence= 50%

T.ID	ITEMS
T1	SONY, BPL, LG
T2	BPL, SAMSUNG
T3	BPL, ONIDA
T4	SONY, BPL, SAMSUNG
T5	SONY, ONIDA
T6	BPL, ONIDA
T7	SONY, ONIDA
T8	SONY, BPL, ONIDA, LG
T9	SONY, BPL, ONIDA

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Associate

Choose **Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1**

Start Stop Associate output Run information

Result list (right-click for ...)

205944 - Apriori

```

Scheme: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation: electronics
Instances: 9
Attributes: 2
    1>T.ID
    ITEMS
==== Associator model (full training set) ====

Apriori
=====

Minimum support: 0.16 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 16

Size of set of large itemsets L(2): 9

Best rules found:

1. ITEMS=SONY, BPL, LG 1 ==> 1>T.ID=T1 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
2. 1>T.ID=T1 1 ==> ITEMS=SONY, BPL, LG 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
3. ITEMS=BPL, SAMSUNG 1 ==> 1>T.ID=T2 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
4. 1>T.ID=T2 1 ==> ITEMS=BPL, SAMSUNG 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
5. 1>T.ID=T3 1 ==> ITEMS=BPL, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
6. ITEMS=SONY, BPL, SAMSUNG 1 ==> 1>T.ID=T4 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
7. 1>T.ID=T4 1 ==> ITEMS=SONY, BPL, SAMSUNG 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
8. 1>T.ID=T5 1 ==> ITEMS=SONY, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
9. 1>T.ID=T6 1 ==> ITEMS=BPL, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
10. 1>T.ID=T7 1 ==> ITEMS=SONY, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)

```

Status OK Log x 0

205944 - Apriori

29-07-2024 20:59 ENG IN

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Associate
Choose FilteredAssociator -F "weka.filters.MultiFilter -F '\weka.filters.unsupervised.attribute.ReplaceMissingValues \'-S 1\' -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1"
Start Stop
Associate output
@relation electronics-weka.filters.unsupervised.attribute.ReplaceMissingValues-weka.filters.MultiFilter-Weka.filters.unsupervised.attribute.ReplaceMissingValues-S1
@attribute Ix:T.ID {"1,2,3,4,5,6,7,8,9}
@attribute ITEMS {"SONY", "BPL", "LG", "BPL", "SAMSUNG", "SONY", "BPL", "SAMSUNG", "SONY", "ONIDA", "SONY", "BPL", "ONIDA", "LG", "SONY", "BPL", "ONIDA"}
@data
Associate Model
Apriori
=====
Minimum support: 0.16 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:
Size of set of large itemsets L(1): 16
Size of set of large itemsets L(2): 9
Best rules found:
1. ITEMS=SONY, BPL, LG 1 ==> Ix:T.ID=T1 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
2. Ix:T.ID=T1 1 ==> ITEMS=SONY, BPL, LG 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
3. ITEMS=BPL, SAMSUNG 1 ==> Ix:T.ID=T2 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
4. Ix:T.ID=T2 1 ==> ITEMS=BPL, SAMSUNG 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
5. Ix:T.ID=T3 1 ==> ITEMS=BPL, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
6. ITEMS=SONY, BPL, SAMSUNG 1 ==> Ix:T.ID=T4 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
7. Ix:T.ID=T4 1 ==> ITEMS=SONY, BPL, SAMSUNG 1   <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
8. Ix:T.ID=T5 1 ==> ITEMS=SONY, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
9. Ix:T.ID=T6 1 ==> ITEMS=BPL, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
10. Ix:T.ID=T7 1 ==> ITEMS=SONY, ONIDA 1   <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)

```

11. The given are the strike-rates scored by a batsman in season 1 in different tournaments. 100, 70, 60, 90, 90

- (a) min-max normalization by setting min = 0 and max = 1
- (b) z-score normalization
- (c) z-score normalization using the mean absolute deviation instead of standard deviation
- (d) normalization by decimal scaling

```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled18* Untitled19* Untitled20* Untitled21* Untitled22* Untitled23* Untitled24* Run Source
# Define the strike rates
strike_rates <- c(100, 70, 60, 90, 90)

# Min-Max Normalization
min_max_normalized <- (strike_rates - min(strike_rates)) / (max(strike_rates) - min(strike_rates))
print("Min-Max Normalization:")
print(min_max_normalized)
print(min_max_normalized)

# Z-Score Normalization
mean_sr <- mean(strike_rates)
sd_sr <- sd(strike_rates)
z_score_normalized <- (strike_rates - mean_sr) / sd_sr
print("Z-Score Normalization:")
print(z_score_normalized)
print(z_score_normalized)

# Z-Score Normalization using Mean Absolute Deviation
mad_sr <- mean(abs(strike_rates - mean_sr))
z_score_mad_normalized <- (strike_rates - mean_sr) / mad_sr
print("Z-Score Normalization using Mean Absolute Deviation:")
print(z_score_mad_normalized)
print(z_score_mad_normalized)

# Normalization by Decimal Scaling
decimal_scaling_norm <- 10^(1:5) * (strike_rates - 1) / 9
print(decimal_scaling_norm)
print(decimal_scaling_norm)

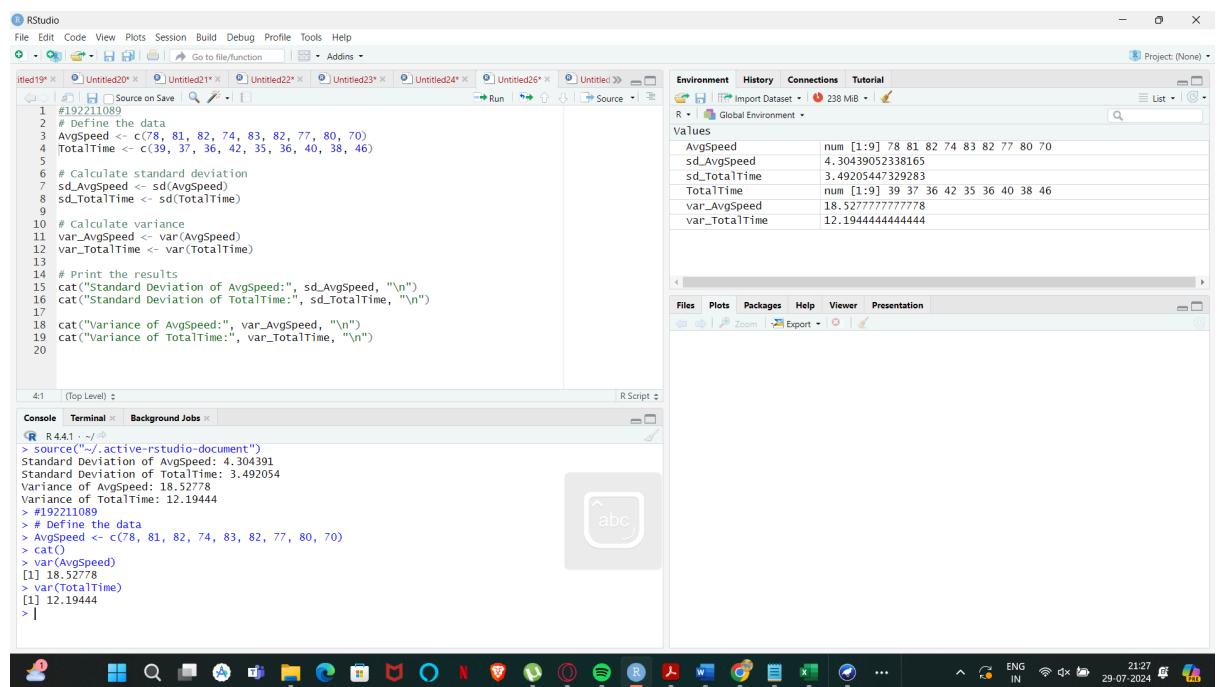
# Define the strike rates
strike_rates <- c(100, 70, 60, 90, 90)
print(min_max_normalized)
print(z_score_normalized)
print(z_score_mad_normalized)
print(decimal_scaling_norm)

```

12. Suppose some car is tested for the AvgSpeed and TotalTime data for 9 randomly selected car with the following result

AvgSpeed (in kph)	78	81	82	74	83	82	77	80	70
TotalTime (in mins)	39	37	36	42	35	36	40	38	46

- a) Calculate the standard deviation of AvgSpeed and TotalTime.
- b) Calculate the Variance of AvgSpeed and TotalTime for the above dataset.



The screenshot shows the RStudio interface with the following details:

- Code Editor:** Contains R script code for defining the data and calculating statistics. The code is as follows:

```

1 #192211089
2 # Define the data
3 AvgSpeed <- c(78, 81, 82, 74, 83, 82, 77, 80, 70)
4 TotalTime <- c(39, 37, 36, 42, 35, 36, 40, 38, 46)
5
6 # Calculate standard deviation
7 sd_AvgSpeed <- sd(AvgSpeed)
8 sd_TotalTime <- sd(TotalTime)
9
10 # Calculate variance
11 var_AvgSpeed <- var(AvgSpeed)
12 var_TotalTime <- var(TotalTime)
13
14 # Print the results
15 cat("Standard Deviation of AvgSpeed:", sd_AvgSpeed, "\n")
16 cat("Standard Deviation of TotalTime:", sd_TotalTime, "\n")
17
18 cat("Variance of AvgSpeed:", var_AvgSpeed, "\n")
19 cat("Variance of TotalTime:", var_TotalTime, "\n")
20

```

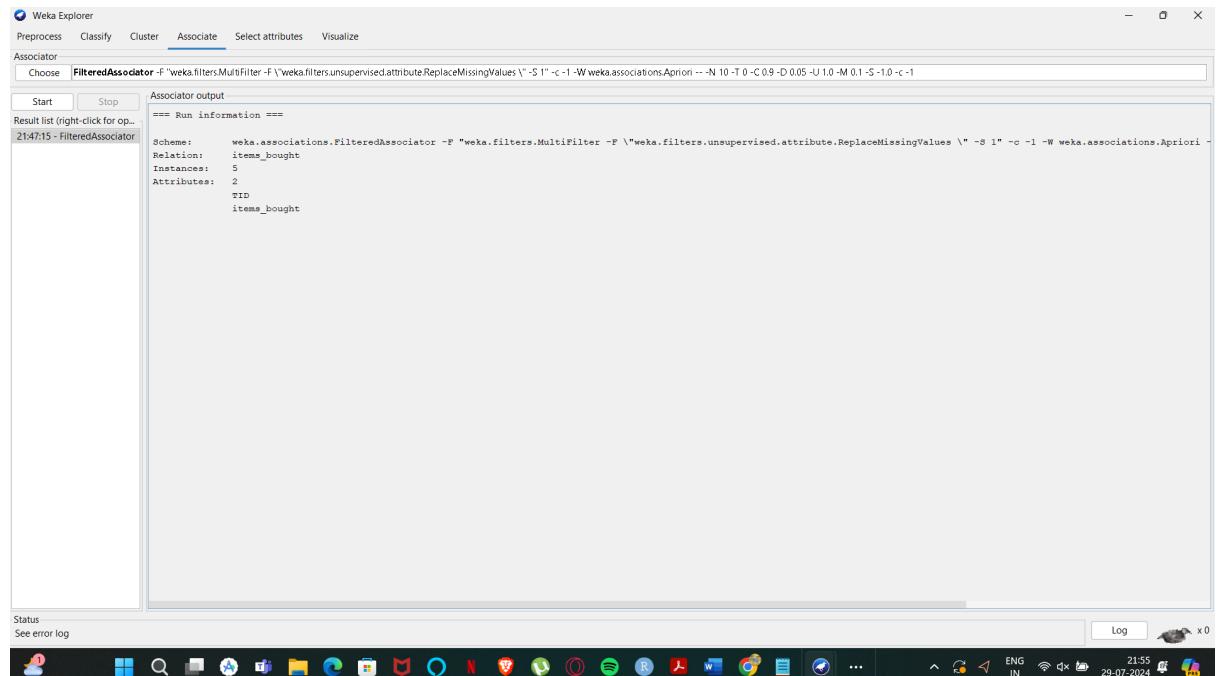
- Environment View:** Shows the variables defined in the session, their types, and values. The variables are:

Variable	Type	Value
AvgSpeed	num [1:9]	78 81 82 74 83 82 77 80 70
sd_AvgSpeed	num	4.30439052338165
sd_TotalTime	num	3.49205447329283
TotalTime	num [1:9]	39 37 36 42 35 36 40 38 46
var_AvgSpeed	num	18.52777777777778
var_TotalTime	num	12.19444444444444

- Console View:** Displays the R command history and the output of the script. The output includes the calculated statistics: Standard Deviation of AvgSpeed: 4.30439052338165, Standard Deviation of TotalTime: 3.49205447329283, Variance of AvgSpeed: 18.52777777777778, and Variance of TotalTime: 12.19444444444444.

13) Consider this table

- a) TID items bought
- b) T100 {M, O, N, K, E, Y}
- c) T200 {D, O, N, K, E, Y }
- d) T300 {M, A, K, E}
- e) T400 {M, U, C, K, Y}
- f) T500 {C, O, O, K, I ,E}
- g) (a) Find all frequent item set using Apriori and FP-growth, respectively. Compare the efficiency of the two mining processes.
- h) (b) List all of the strong association rules (with support s and confidence c) matching the following metarule, where X is a variable representing customers, and itemi denotes variables representing items (e.g., "A", "B", etc.):
- i) $\forall x \in \text{transaction}, \text{buys}(X, \text{item}1) \wedge \text{buys}(X, \text{item}2) \Rightarrow \text{buys}(X, \text{item}3)$



QUESTION BANK :

1. Implement of the R script using a group of 12 sales price records has been sorted as follows: 5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215. Partition them into three bins by each of the following methods.

(a) equal-frequency (equi depth) partitioning

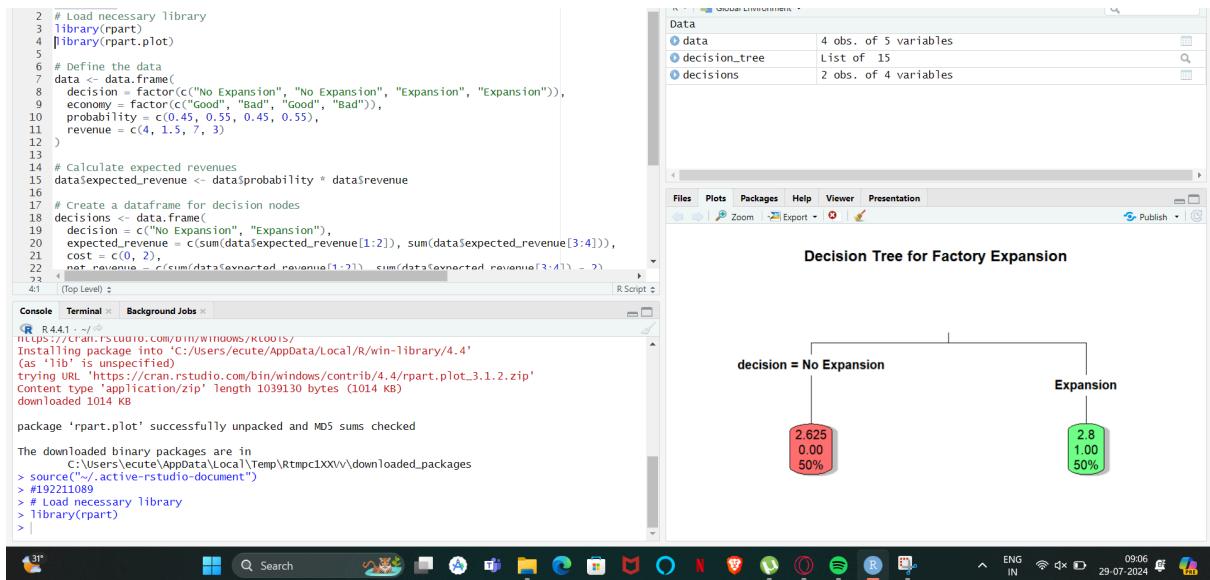
(b) equal-width partitioning

(c) clustering

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Displays R code for generating three types of partitions from a dataset of 12 sales prices. The code includes:
 - # Load necessary library
 - library(cluster)
 - # Sales price records
 - sales_prices <- c(5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215)
 - # Number of bins
 - num_bins <- 3
 - # Equal-Frequency (Equi-Depth) Partitioning
 - # Create bins so that each bin has approximately the same number of records
 - bin_indices_eq_freq <- cut(sales_prices, breaks=num_bins, labels=FALSE)
 - # Equal-width Partitioning
 - # Calculate width of each bin
 - range_width <- (max(sales_prices) - min(sales_prices)) / num_bins
 - # Define bin edges
 - bin_edges <- seq(min(sales_prices), max(sales_prices), length.out=num_bins+1)
 - # Create bins
 - bin_indices_eq_width <- cut(sales_prices, breaks=bin_edges, include.lowest=TRUE, labels=FALSE)
- Environment Pane:** Shows the global environment with the following objects:
 - kmeans_result
 - partitioned_data
 - sales_matrix
 - values
 - bin_edges
 - bin_indices_eq_freq
 - bin_indices_eq_width
 - cluster_assignments
 - num_bins
 - range_width
- Console:** Displays the R session history, including the execution of the code and the output of the print command for the partitioned data.
- System Tray:** Shows system information like temperature (31°C), weather (Partly sunny), date (29-07-2024), and time (09:00).

2. A gadget factory has been quite successful for the past 10 years and Ms.Marry, the manager of the company wondering whether to expand the factory this year or not. The cost to expand factory is \$2M. With no expansion, expected revenue is \$4M if the economy stays good; while only \$1.5M if the economy is bad. If manager expands the factory, expected to receive \$7M. if economy is good and \$3M if economy is bad. Assume that there is a 45% chance of a good economy and a 55% chance of a bad economy. Draw a Decision Tree showing these choices.



3. Implement using WEKA for the given Apply Apriori Algorithm for given database below Assume Min_sup=2 TID Items

TID	Items
1	Bread, Peanuts, Milk, Fruit, Jam
2	Bread, Jam, Soda, Chips, Milk, Fruit
3	Steak, Jam, Soda, Chips, Bread
4	Jam, Soda, Peanuts, Milk, Fruit
5	Jam, Soda, Chips, Milk, Bread
6	Fruit, Soda, Chips, Milk
7	Fruit, Soda, Peanuts, Milk
8	Fruit, Peanuts, Cheese, Yogurt

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Associate
Choose: Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Start Stop
Associator output
Result list (right-click for)
205944 - Apriori
Scheme: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation: electronics
Instances: 9
Attributes: 2
ix2T.ID
ITEMS
==== Associator model (full training set) ====
Apriori
=====
Minimum support: 0.16 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17
Generated sets of large itemsets:
Size of set of large itemsets L(1): 16
Size of set of large itemsets L(2): 9
Best rules found:
1. ITEMS=SONY, BFL, LG 1 ==> ix2T.ID=TI 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
2. ix2T.ID=1 1 ==> ITEMS=SONY, BFL, LG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
3. ITEMS=BFL, SAMSUNG 1 ==> ix2T.ID=P2 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
4. ix2T.ID=P2 1 ==> ITEMS=BFL, SAMSUNG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
5. ix2T.ID=P3 1 ==> ITEMS=BFL, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
6. ix2T.ID=P4 1 ==> ITEMS=SONY, BFL, SAMSUNG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
7. ix2T.ID=P5 1 ==> ITEMS=SONY, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
8. ix2T.ID=P6 1 ==> ITEMS=BFL, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
9. ix2T.ID=P7 1 ==> ITEMS=SONY, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
10. ix2T.ID=P7 1 ==> ITEMS=SONY, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
Status
OK
Log x0

```

4. Use following group of data: 200, 300, 400, 600, 1000

(a) min-max normalization by setting min = 0 and max = 1 (b)

(b) z-score normalization

(c) (c) z-score normalization using the mean absolute deviation instead of standard deviation (d) normalization by decimal scaling

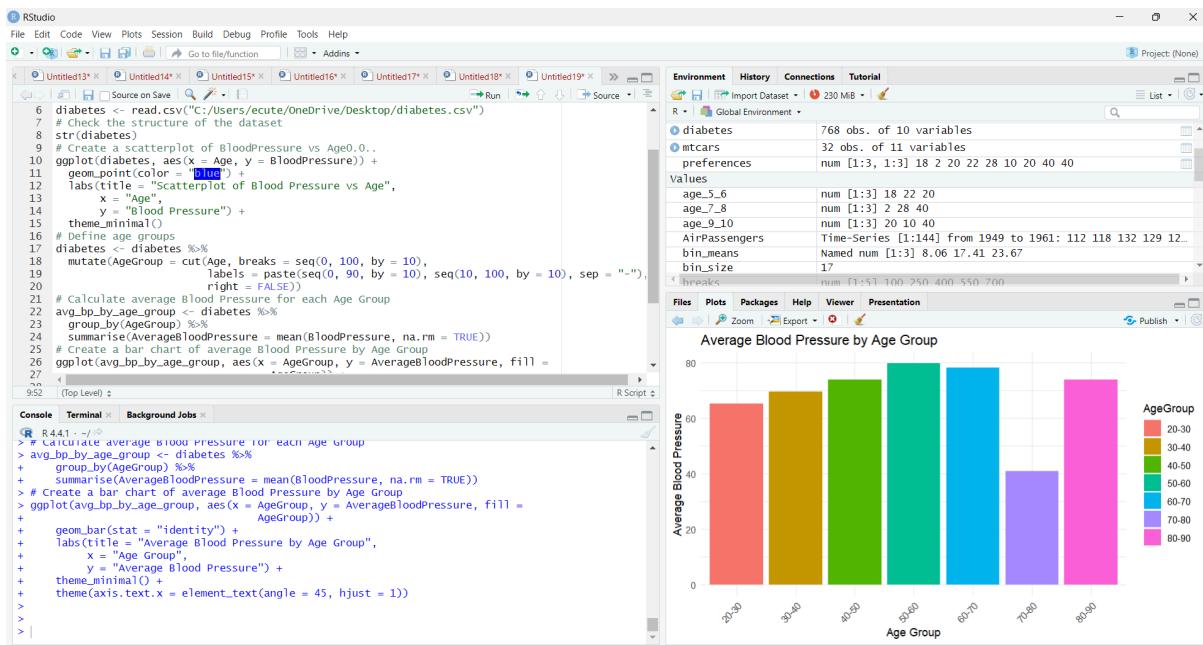
```

# Data
3 data <- c(200, 300, 400, 600, 1000)
4
5 # (a) Min-Max Normalization
6 min_val <- min(data)
7 max_val <- max(data)
8 min_max_normalized <- (data - min_val) / (max_val - min_val)
9 cat("Min-Max Normalization:\n")
10 print(min_max_normalized)
11
12 # (b) Z-Score Normalization
13 mean_val <- mean(data)
14 sd_val <- sd(data)
15 z_score_normalized <- (data - mean_val) / sd_val
16 cat("Z-Score Normalization:\n")
17 print(z_score_normalized)
18
19 # (c) Z-Score Normalization using Mean Absolute Deviation (MAD)
20 mad_val <- mean(abs(data - mean_val))
21 z_score_mad_normalized <- (data - mean_val) / mad_val
22 cat("Z-Score Normalization using MAD:\n")
23 print(z_score_mad_normalized)
51 [Top Level] <

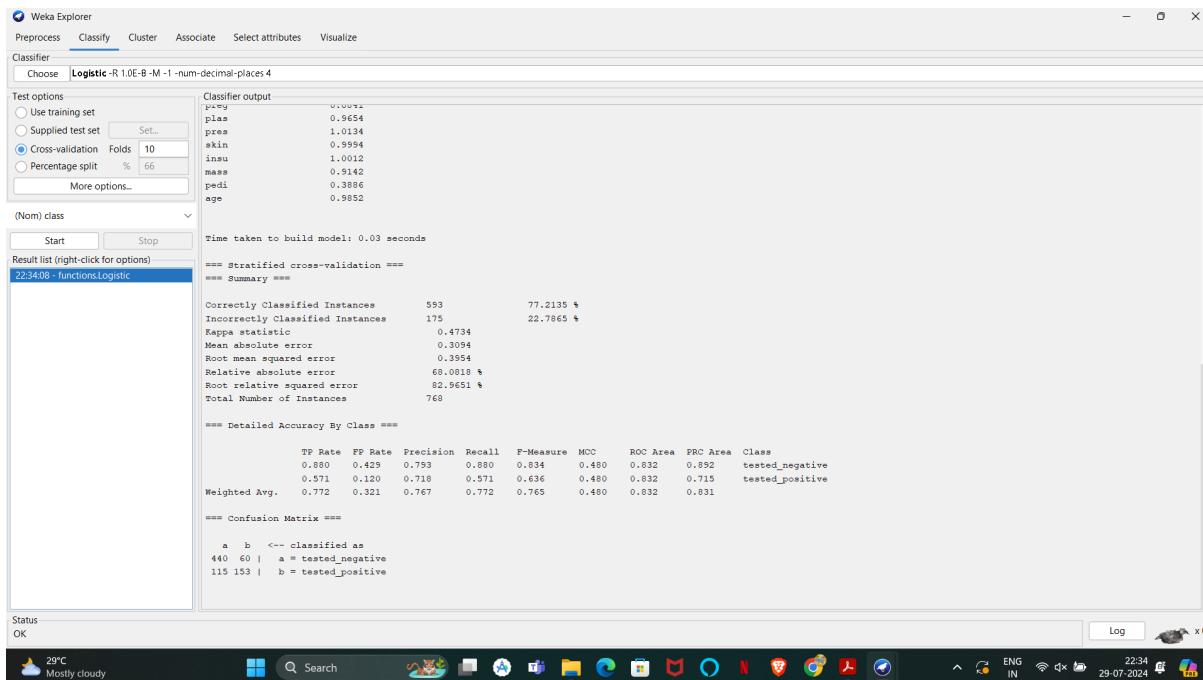
```

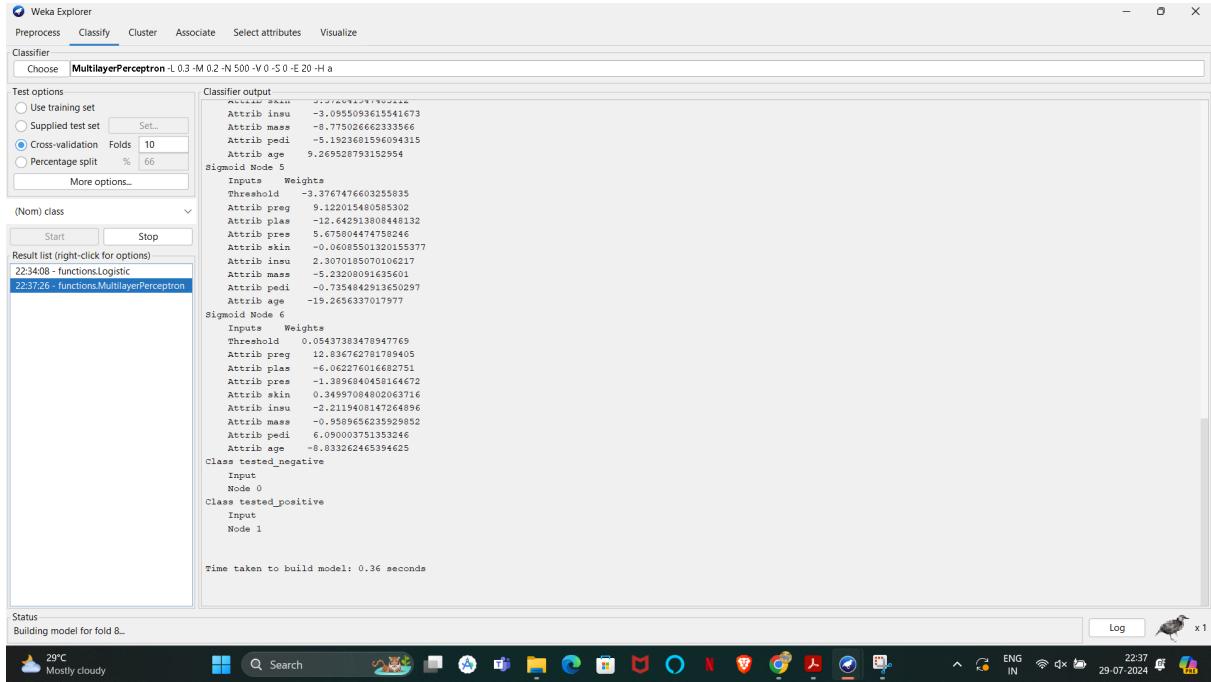
	Values
data	num [1:5] 200 300 400 600 1000
decimal_scaled	num [1:5] 0.2 0.3 0.4 0.6 1
mad_val	240
max_abs_val	1000
max_val	1000
mean_val	500
min_max_normalized	num [1:5] 0 0.125 0.25 0.5 1
min_val	200
scale_factor	1000
sd_val	316.227766016838

5. Implement using R language in which age group of people are affected by blood pressure based on the diabetes dataset show it using scatterplot and bar chart (that is BloodPressure vs Age using dataset “diabetes.csv”)



6. Analysis the dataset “diabetes.csv” how the diabetes trend is for different age people, using linear regression and multiple regression.





7. Implement using WEKA for the given Suppose a database has five transactions. Let min sup= 50%(2) and min con f = 80%.

Transactions Items

T1 (M, O, N, K, E, Y)

T2 (D, O, N, K, E, Y)

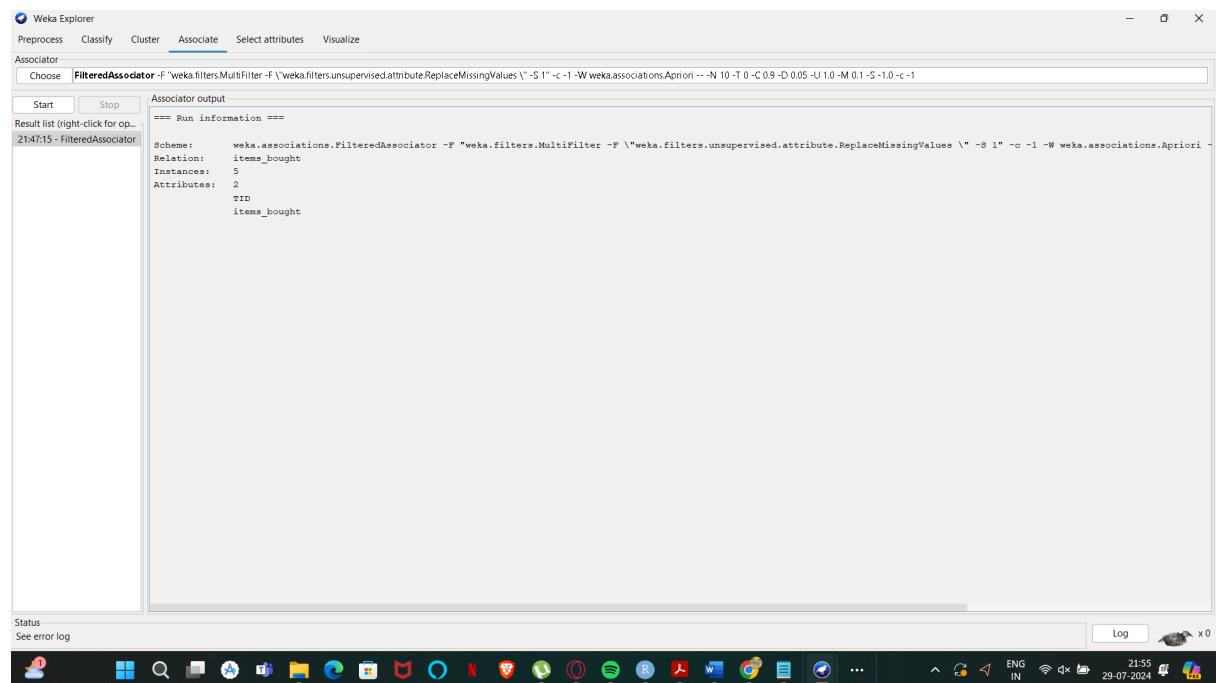
T3 (M, A, K, E)

T4 (M, U, C, K, Y)

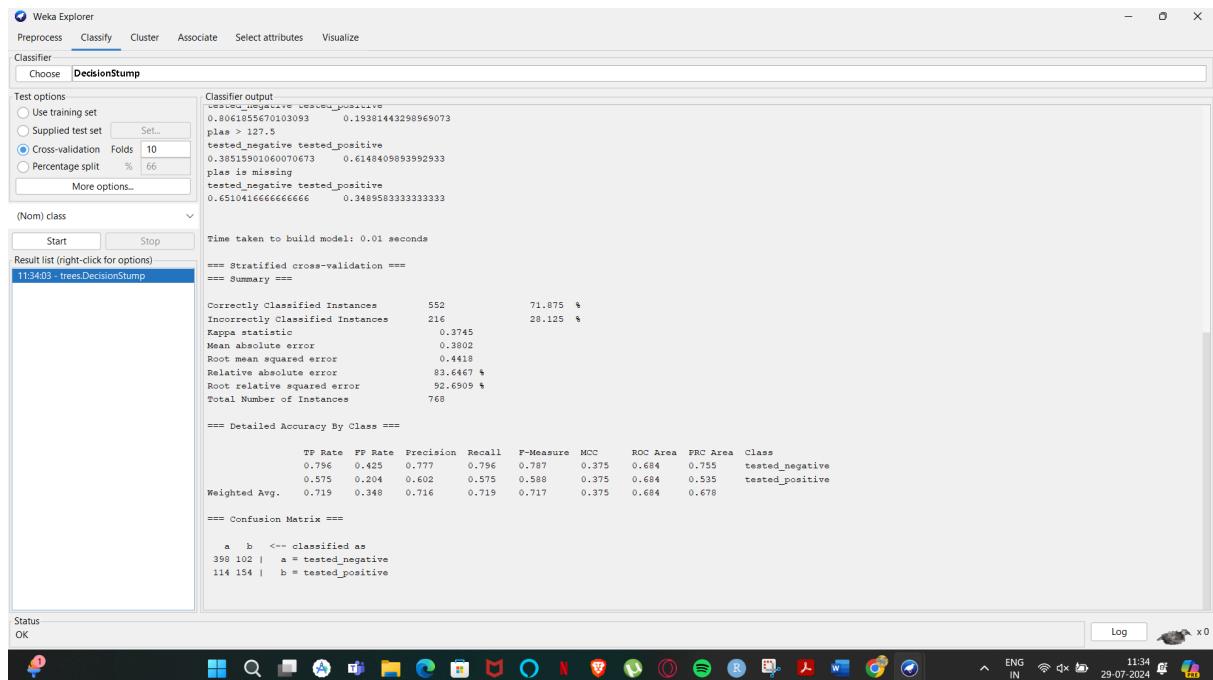
T5 (C,O, O, K, I ,E)

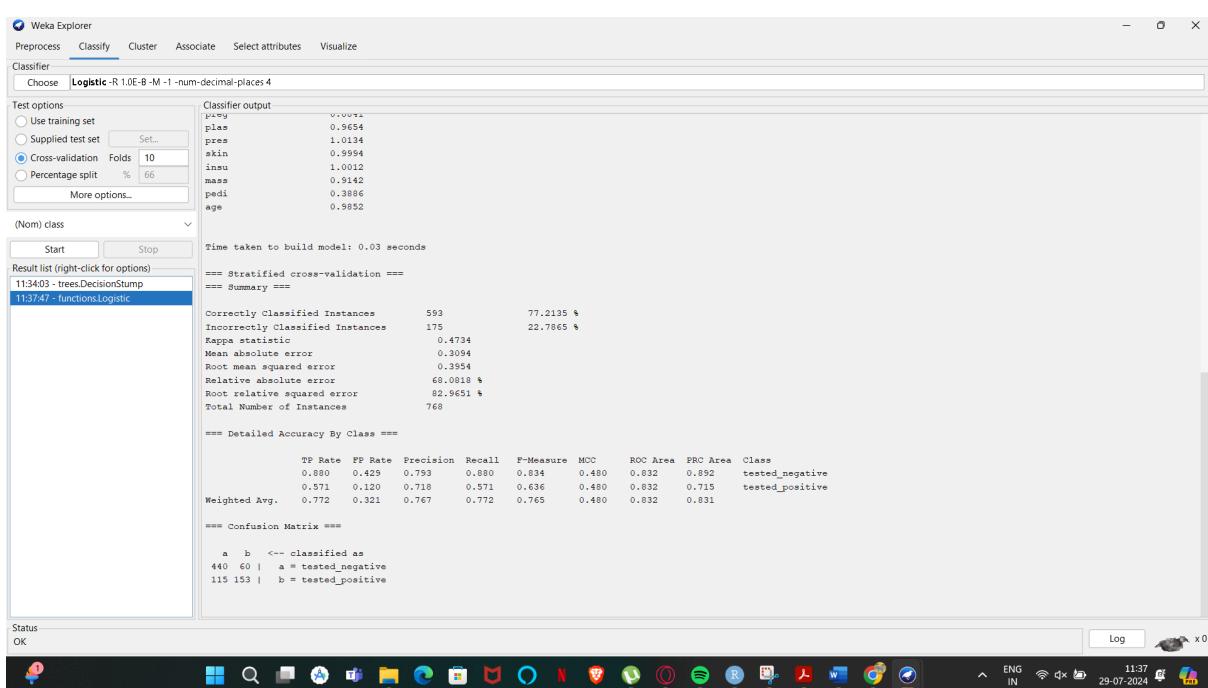
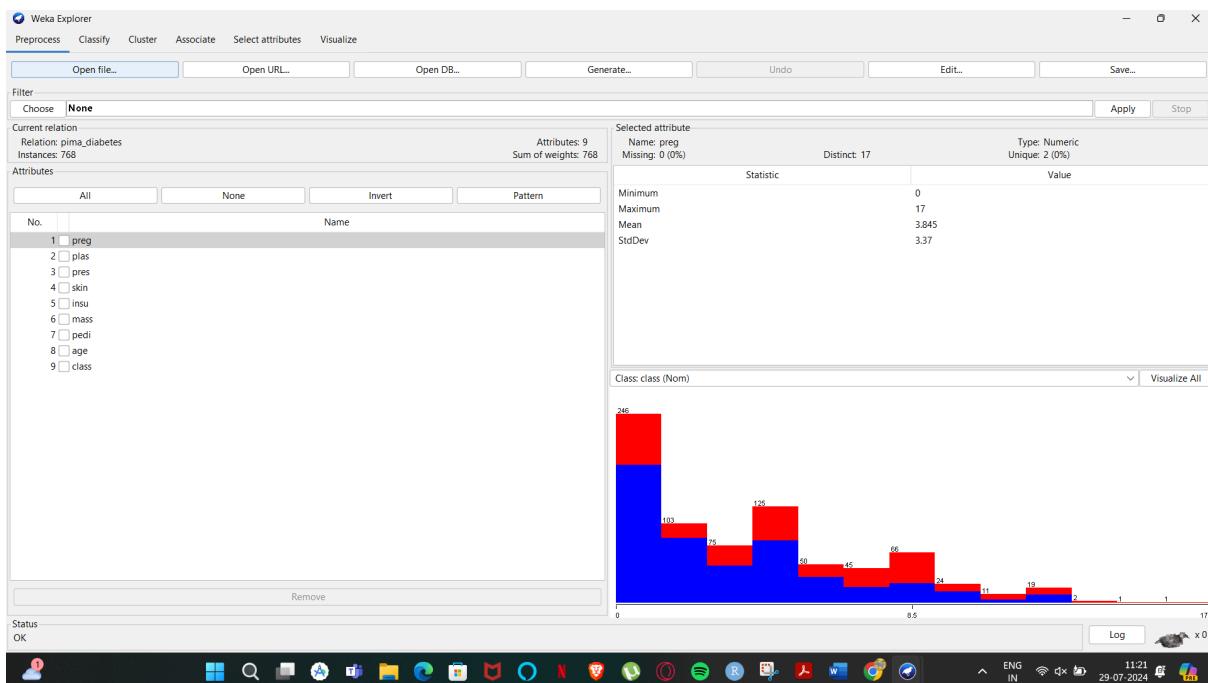
- Find all frequent item sets using Apriori algorithm

- Also draw FP-Growth Tree



8. Prediction of Categorical Data using Decision Tree Algorithm through WEKA using any datasets. a) Tree b) Preprocess c) Logistic

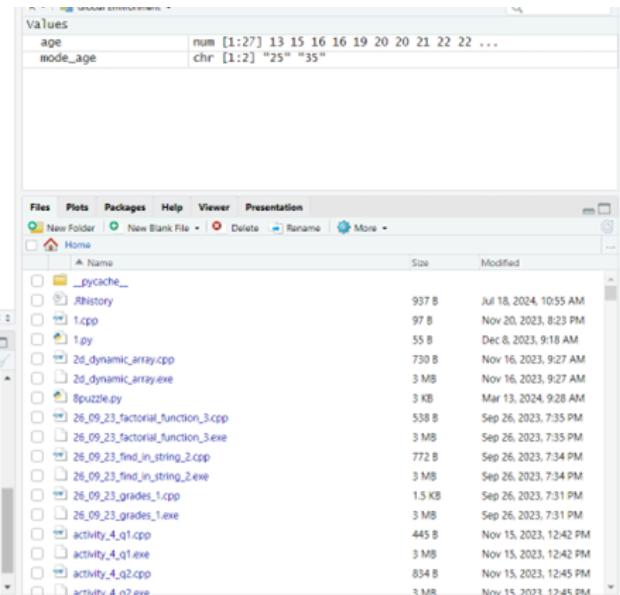




9. Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?

```
2 #mean, median, mode, quantile  
3 age<-c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)  
4 mean(age)  
5 median(age)  
6 mode_age<-names(table(age))[table(age)==max(table(age))]  
7 mode_age  
8 range(age)  
9 quantile(age,.25)  
10 quantile(age,.75)  
11
```



10. Download the Dataset "water" From R dataset Link. Find out whether there is a linear relation between attributes "mortality" and "hardness" by plot function. Fit the Data into the Linear Regression model. Predict the mortality for the hardness=88.

```
 2 # Install and load the HSAUR package
 3 install.packages("HSAUR")
 4 library(HSAUR)
 5
 6 # Load the water dataset
 7 data("water", package = "HSAUR")
 8
 9 # Plot the relationship between mortality and hardness
10 plot(water$mortality ~ water$hardness,
11       main = "Mortality vs Water Hardness",
12       xlab = "Water Hardness (ppm)",
13       ylab = "Mortality (per 100,000 males)",
14       col = as.factor(water$location))
15
16 # Fit a linear regression model
17 model <- lm(mortality ~ hardness, data = water)
18 summary(model)
19
20 # Predict mortality for a hardness of 88 ppm
21 new_data <- data.frame(hardness = 88)
22 prediction <- predict(model, new_data)
23 print(prediction)
41 [Top Level] R Script
```

Console Terminal × Background Jobs ×

R 4.4.1 -- -- / --

100%|██████████| 0/0 BDLIST[install.packages("HSAUR")]

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/eucete/AppData/Local/R/win-library/4.4'

(as 'lib' is unspecified)

trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.4/HSAUR_1.3-10.zip'

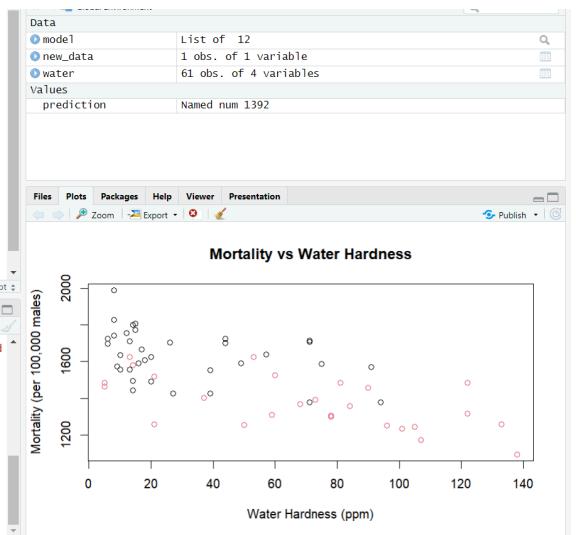
Content type 'application/zip' length 2160529 bytes (2.1 MB)

downloaded 2.1 MB

package 'HSAUR' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:\Users\euclite\AppData\Local\Temp\Rtmp0whbOA\downloaded_packages

> |



11. Create the dataset using ARFF file format

a. Find the frequent itemsets and generate association rules on this. Assume that minimum support threshold ($s = 33.33\%$) and minimum confident threshold ($c = 60\%$).

b.List the various rule generated by apriori and FP tree algorithim ,mention wheather accepted or rejected.

Transaction ID	Items
T1	Hot Dogs, Buns, Ketchup
T2	Hot Dogs, Buns
T3	Hot Dogs, Coke, Chips
T4	Chips, Coke
T5	Chips, Ketchup
T6	Hot Dogs, Coke, Chips

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Associate
Choose Apriori-N 10-T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1
Start Stop Associate output
Result list (right-click for ...)
192416 - Aprori
==== Associator model (full training set) ====
Apriori
=====
Minimum support: 0.5 (143 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 10
Generated sets of large itemsets:
Size of set of large itemsets L(1): 6
Size of set of large itemsets L(2): 6
Size of set of large itemsets L(3): 4
Size of set of large itemsets L(4): 1
Best rules found:
1. inv-nodes=0-2 irradiat=no Class=no-recurrence-events 147 => node-caps=no 145 <conf:(0.59)> lift:(1.27) lev:(0.11) [30] conv:(10.97)
2. inv-nodes=0-2 irradiat=no 183 => node-caps=no 177 <conf:(0.57)> lift:(1.25) lev:(0.12) [34] conv:(5.85)
3. node-caps=no irradiat=no Class=no-recurrence-events 151 => inv-nodes=0-2 145 <conf:(0.56)> lift:(1.29) lev:(0.11) [32] conv:(5.51)
4. inv-nodes=0-2 Class=no-recurrence-events 167 => node-caps=no 160 <conf:(0.56)> lift:(1.23) lev:(0.11) [30] conv:(4.67)
5. inv-nodes=0-2 213 => node-caps=no 201 <conf:(0.94)> lift:(1.22) lev:(0.12) [35] conv:(3.67)
6. node-caps=no irradiat=no 188 => inv-nodes=0-2 177 <conf:(0.94)> lift:(1.26) lev:(0.13) [36] conv:(4)
7. node-caps=no Class=no-recurrence-events 171 => inv-nodes=0-2 160 <conf:(0.94)> lift:(1.26) lev:(0.11) [32] conv:(3.64)
8. irradiat=no Class=no-recurrence-events 164 => node-caps=no 151 <conf:(0.92)> lift:(1.19) lev:(0.08) [23] conv:(2.62)
9. inv-nodes=0-2 node-caps=no Class=no-recurrence-events 160 => irradiat=no 148 <conf:(0.91)> lift:(1.19) lev:(0.08) [23] conv:(2.38)
10. node-caps=no 222 ==> inv-nodes=0-2 201 <conf:(0.91)> lift:(1.22) lev:(0.12) [35] conv:(2.58)

```

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Associate

Choose `FilteredAssociator -F "weka.filters.MultiFilter -F "weka.filters.unsupervised.attribute.ReplaceMissingValues \\"-S 1\\" -c -1 -W weka.associations.Apriori -- -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1`

Start Stop Associate output

Result list (right-click for options)

19:24:16 - Apriori
19:24:59 - filteredAssociator

Associate Model

Apriori

=====

```
Minimum support: 0.35 (100 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 13

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10
Size of set of large itemsets L(2): 20
Size of set of large itemsets L(3): 8
Size of set of large itemsets L(4): 2

Best rules found:
```

1. inv-nodes=0-2 breast=left irradiat=no 101 => node=caps=no 100 <conf:(0.99)> lift:(1.23) lev:(0.07) [18] conv:(9.89)
2. inv-nodes=0-2 irradiat=no Class=no_recurrence-events 147 => node=caps=no 145 <conf:(0.59)> lift:(1.23) lev:(0.09) [26] conv:(9.55)
3. inv-nodes=0-2 breast=left 115 => node=caps=no 113 <conf:(0.58)> lift:(1.22) lev:(0.07) [20] conv:(7.51)
4. inv-nodes=0-2 irradiat=no 183 => node=caps=no 179 <conf:(0.58)> lift:(1.22) lev:(0.11) [31] conv:(7.17)
5. inv-nodes=0-2 Class=no_recurrence-events 167 => node=caps=no 161 <conf:(0.56)> lift:(1.2) lev:(0.09) [26] conv:(4.67)
6. node=caps=no breast=left irradiat=no 104 => inv-nodes=0-2 100 <conf:(0.96)> lift:(1.29) lev:(0.08) [22] conv:(5.31)
7. node=caps=no irradiat=no Class=no_recurrence-events 151 => inv-nodes=0-2 145 <conf:(0.56)> lift:(1.29) lev:(0.11) [32] conv:(5.51)
8. inv-nodes=0-2 213 => node=caps=no 204 <conf:(0.56)> lift:(1.19) lev:(0.11) [32] conv:(4.17)
9. menopause=pmeno inv-nodes=0-2 112 => node=caps=no 106 <conf:(0.95)> lift:(1.18) lev:(0.06) [15] conv:(3.13)
10. node=caps=no irradiat=no 190 => inv-nodes=0-2 179 <conf:(0.54)> lift:(1.26) lev:(0.13) [37] conv:(4.04)

Status OK Log x0

12. Prediction of Categorical Data using Rule base classification and decision tree classification through WEKA using any datasets. Compare the accuracy using two algorithm and plot the graph

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose `DecisionStump`

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds 10
- Percentage split % 66
- More options...

(Nom) class

Start Stop

Result list (right-click for options)

20:29:54 - trees.DecisionStump

Classifier output

```
0.8061855670103093 0.19801443298965073
p1as 127.8
tested_negative tested_positive
0.39515901060070873 0.6148409893992933
p1as is missing
tested_negative tested_positive
0.6510416666666666 0.3469583333333333
```

Time taken to build model: 0.01 seconds

==== Stratified cross-validation ====
==== Summary ====

	Correctly Classified Instances	71.875 %
Incorrectly Classified Instances	216	28.125 %
Kappa statistic	0.3745	
Mean absolute error	0.3802	
Root mean squared error	0.4418	
Relative absolute error	83.6467 %	
Root relative squared error	92.6909 %	
Total Number of Instances	768	

==== Detailed Accuracy By Class ====

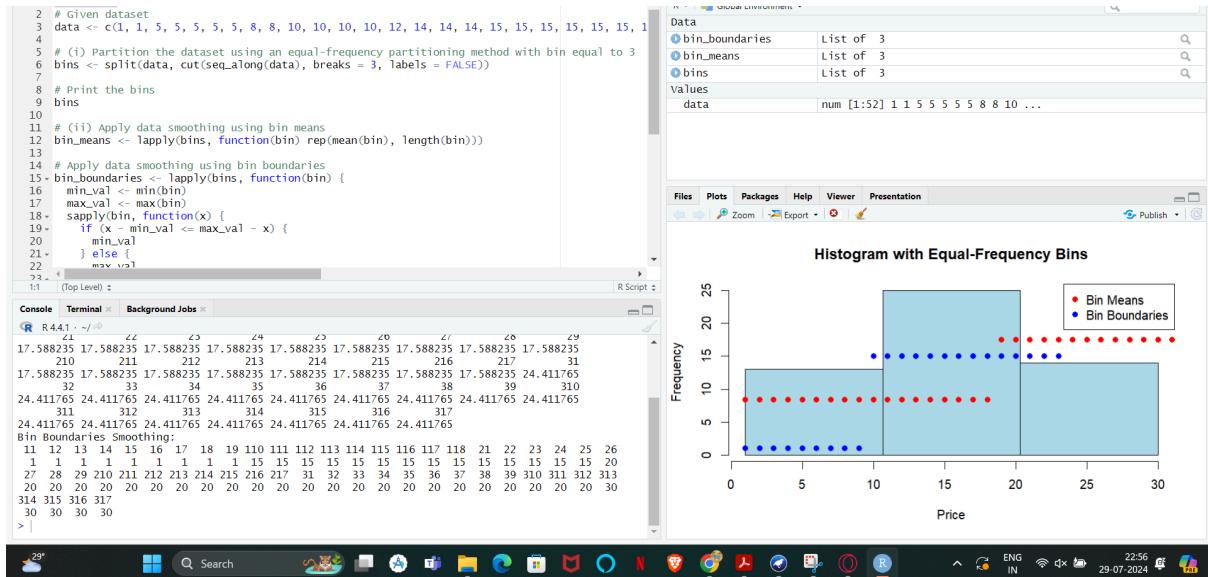
	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.796	0.425	0.777	0.796	0.787	0.375	0.684	0.755		tested_negative
0.575	0.204	0.602	0.575	0.588	0.375	0.684	0.535		tested_positive
Weighted Avg.	0.719	0.349	0.716	0.719	0.717	0.375	0.684	0.678	

==== Confusion Matrix ====

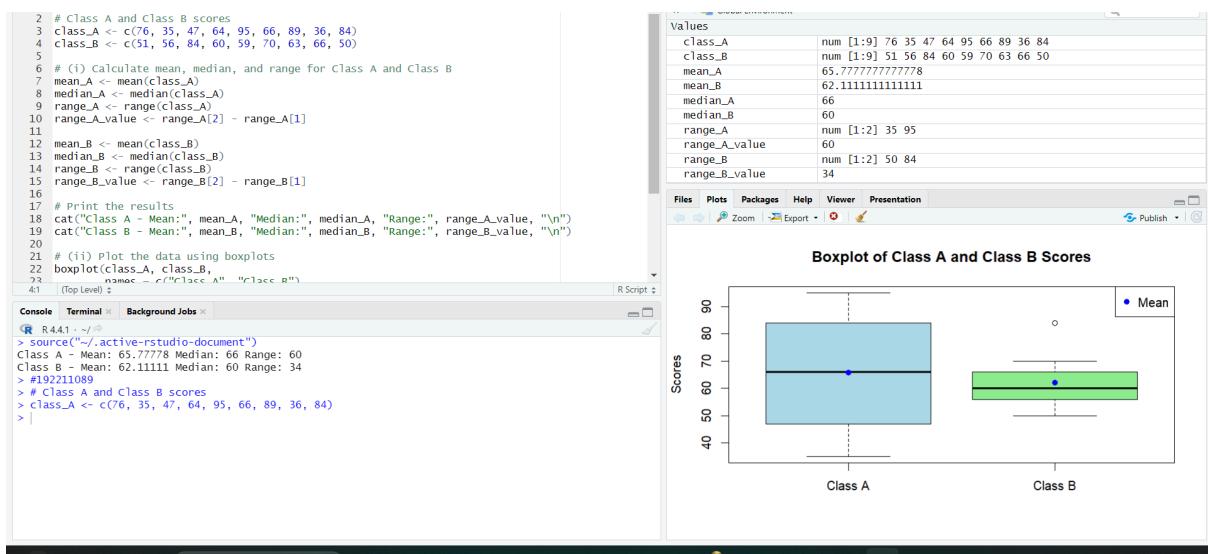
```
a b <- classified as
398 102 | a = tested_negative
114 154 | b = tested_positive
```

Status OK Log x0

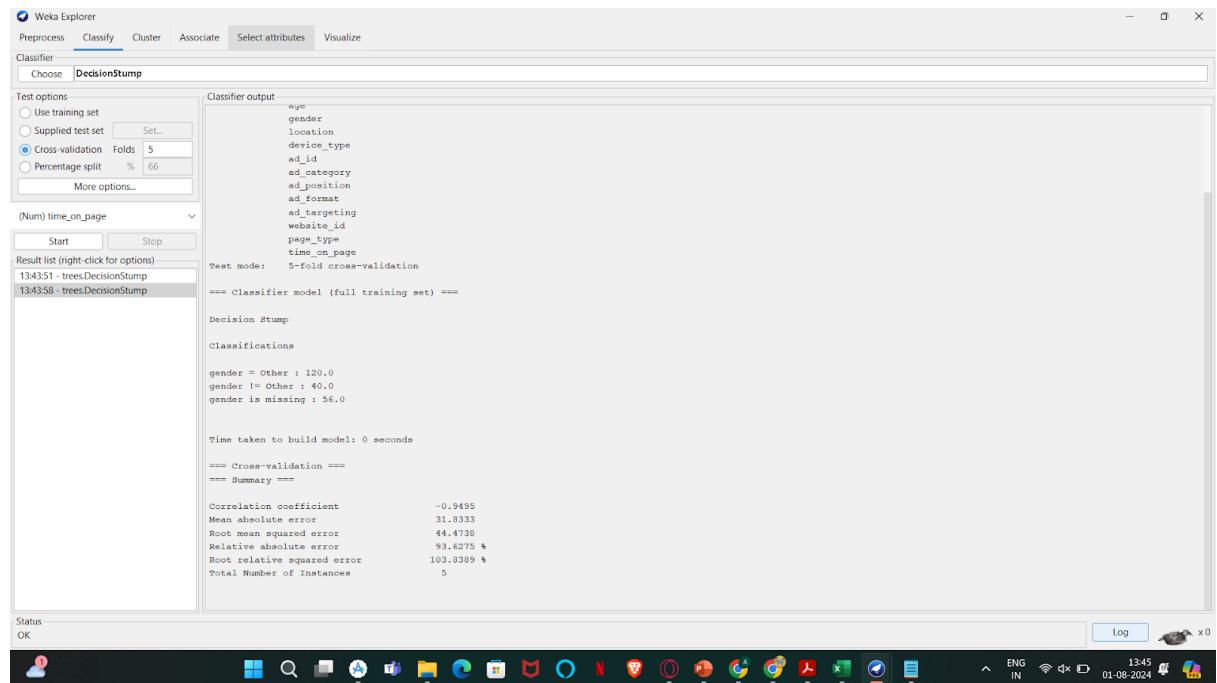
13. Imagine that you have selected data from the All Electronics data warehouse for analysis. The data set will be huge! The following data are a list of All Electronics prices for commonly sold items (rounded to the nearest dollar). The numbers have been sorted: 1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 21, 21, 21, 25, 25, 25, 25, 28, 28, 30, 30, 30. (i) Partition the dataset using an equal-frequency partitioning method with bin equal to 3 (ii) apply data smoothing using bin means and bin boundary. (iii) Plot Histogram for the above frequency division



14. Two Maths teachers are comparing how their Year 9 classes performed in the end of year exams. Their results are as follows: Class A: 76, 35, 47, 64, 95, 66, 89, 36, 84, 76, 35, 47, 64, 95, 66, 89, 36, 84 Class B: 51, 56, 84, 60, 59, 70, 63, 66, 50, 51, 56, 84, 60, 59, 70, 63, 66, 50 (i) Find which class had scored higher mean, median and range. (ii) Plot above in boxplot and give the inferences



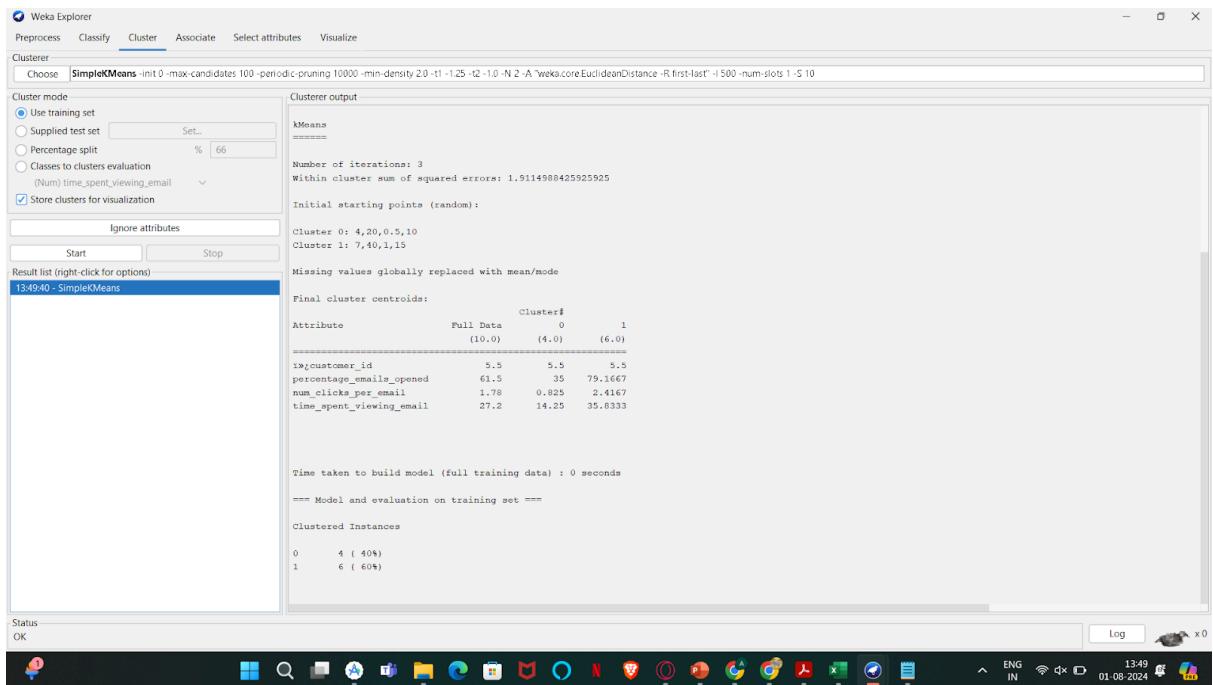
15. Consider a Binary classification model that can be used to predict whether one or more ads on the website will be clicked or not. The models are used to optimize the ad inventory on websites by selecting which ads will have a better chance of being clicked.



16. Consider that Many businesses use cluster analysis to identify consumers who are similar to each other so they can tailor their emails sent to consumers in such a way that maximizes their revenue. Consider a business may collect the following information about consumers

- Percentage of emails opened
- Number of clicks per email
- Time spent viewing email

Using these metrics, a business can perform various cluster analyses to identify consumers who use email in similar ways and tailor the types of emails and frequency of emails they send to different clusters of customers. Compare the performance of the applied clustering algorithm.

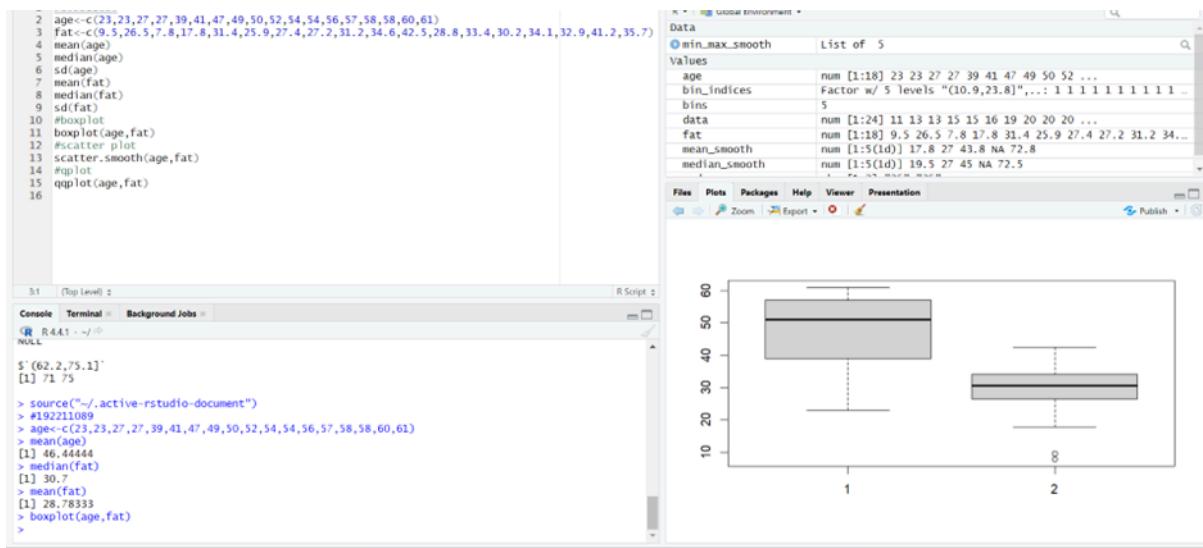


17. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

<i>age</i>	23	23	27	27	39	41	47	49	50
<i>%fat</i>	9.5	26.5	7.8	17.8	31.4	25.9	27.4	27.2	31.2
<i>age</i>	52	54	54	56	57	58	58	60	61
<i>%fat</i>	34.6	42.5	28.8	33.4	30.2	34.1	32.9	41.2	35.7

- (a) Calculate the mean, median, and standard deviation of *age* and *%fat*.
- (b) Draw the boxplots for *age* and *%fat*.
- (c) Draw a *scatter plot* and a *q-q plot* based on these two variables.

Perform the above functions using R – tool

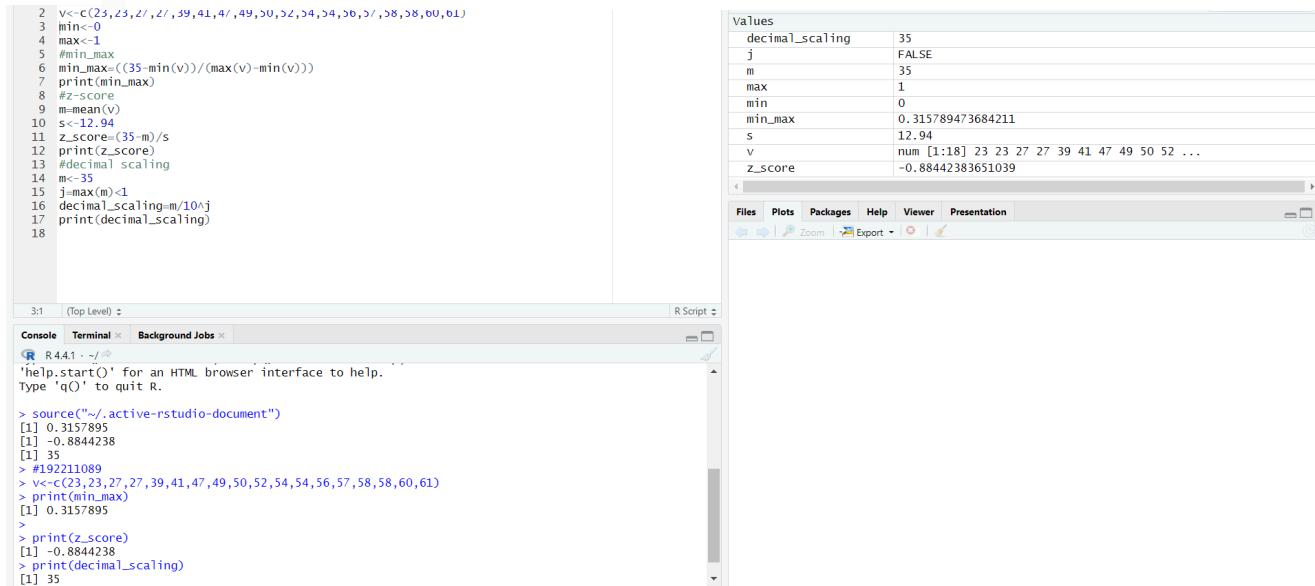


18. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

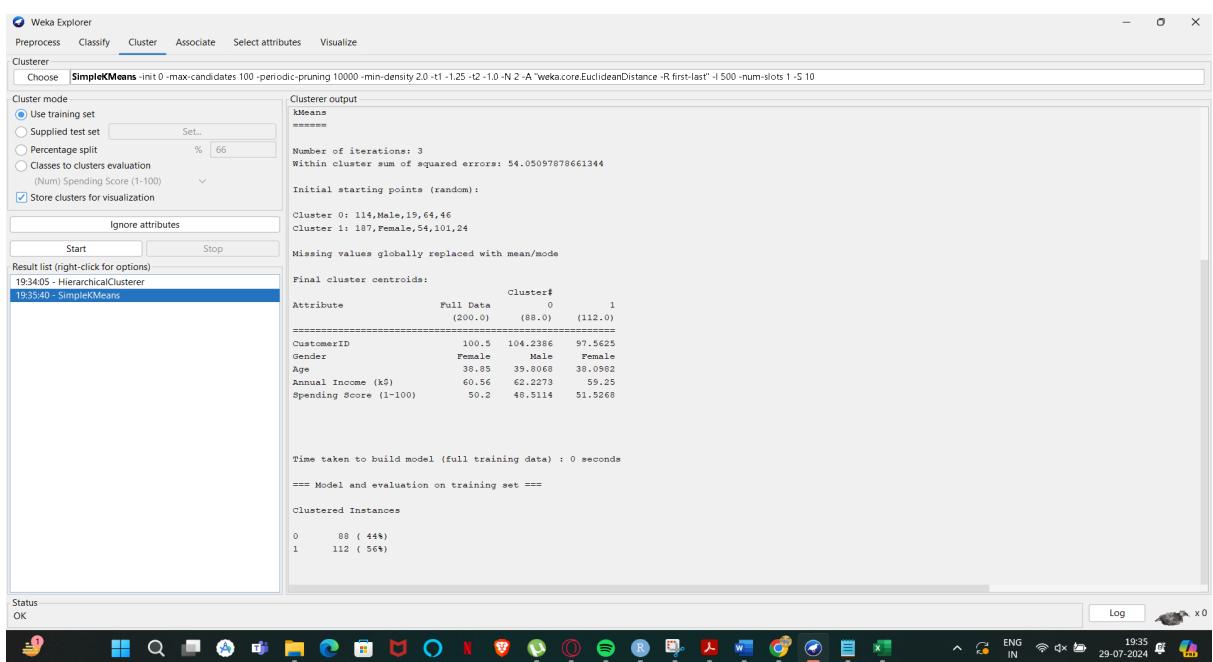
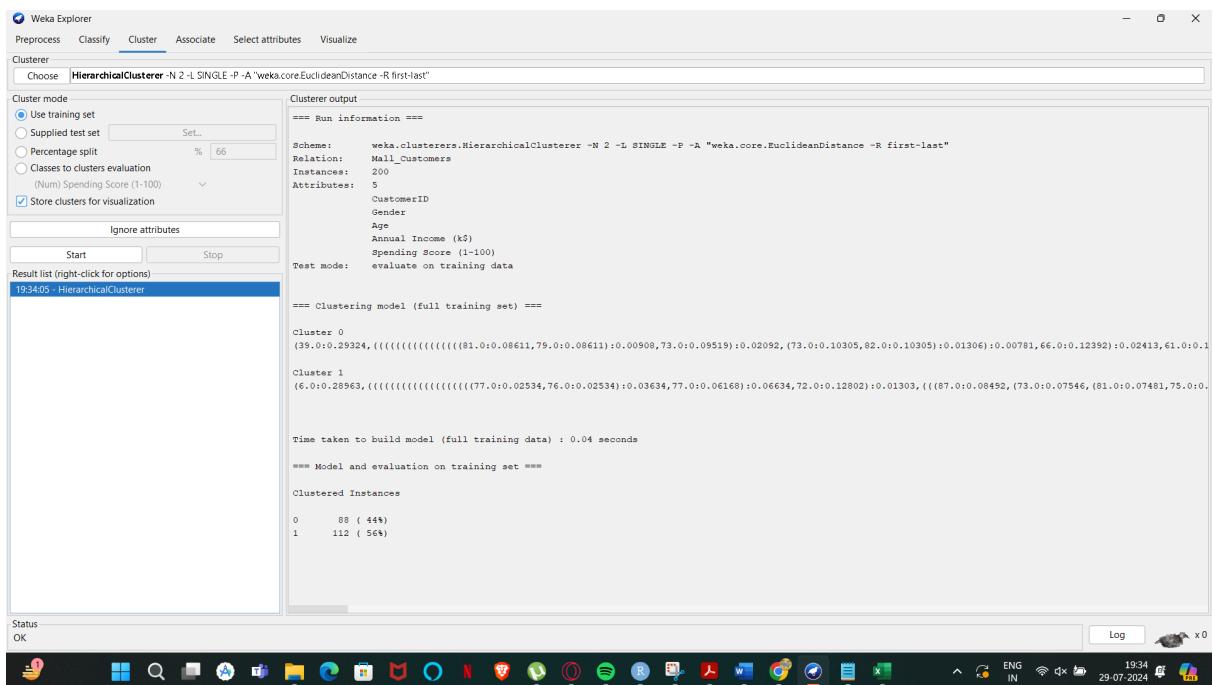
<i>age</i>	23	23	27	27	39	41	47	49	50
<i>%fat</i>	9.5	26.5	7.8	17.8	31.4	25.9	27.4	27.2	31.2
<i>age</i>	52	54	54	56	57	58	58	60	61
<i>%fat</i>	34.6	42.5	28.8	33.4	30.2	34.1	32.9	41.2	35.7

- (i) Use min-max normalization to transform the value 35 for *age* onto the range [0.0, 1.0].
- (ii) Use z-score normalization to transform the value 35 for *age*, where the standard deviation of *age* is 12.94 years.
- (iii) Use normalization by decimal scaling to transform the value 35 for *age*.

Perform the above functions using R – tool



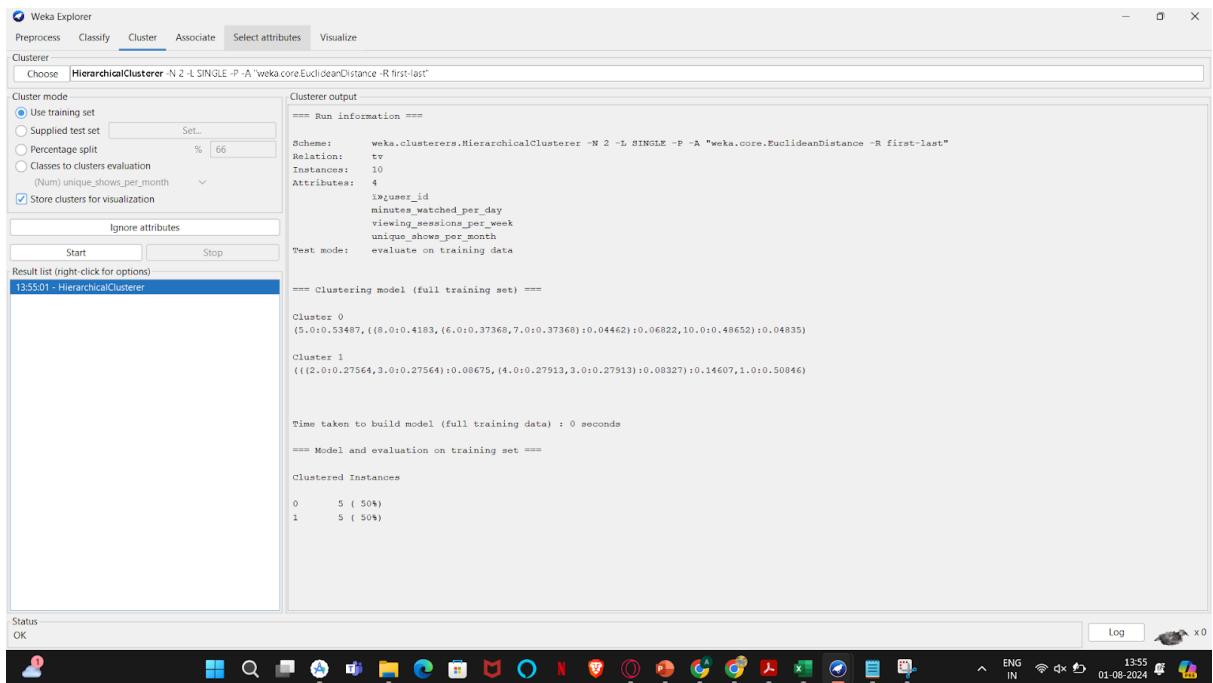
19. Consider that you are owning a supermarket mall and through membership cards, you have some basic data about your customers like Customer ID, age, gender, annual income and spending score. For the above scenario, the Problem Statement was You want to understand the customers who can easily converge [Target Customers] so that the data can be given to the marketing team and plan the strategy accordingly. For the above scenario prepare a dataset and perform Clustering Analysis to segment the customers in the Mall. There are clearly Five segments of Customers based on their Annual Income and Spending Score namely Usual Customers, Priority Customers, Senior Citizen Target Customers, and Young Target Customers. Sample data



20. Streaming services often use clustering analysis to identify viewers who have similar behavior. The streaming service may collect the following data about individuals

- Minutes watched per day
- Total viewing sessions per week
- Number of unique shows viewed per month

Using these metrics, a streaming service can perform cluster analysis to identify high-usage and low-usage users so that they can know whom they should spend most of their advertising dollars on. Apply the Hierarchical clustering algorithm and EM clustering algorithm to identify and compare the performance of the clustering technique.



21. The following values are the number of pencils available in the different boxes. Create a vector and find out the mean, median and mode values of set of pencils in the given data.

Box1	Box2	Box3	Box4	Box5	Box6	Box7	Box8	Box9
25	23	12	11	6	7	8	9	10

The screenshot shows the RStudio interface. On the left, the 'Console' tab displays R code and its execution results. On the right, the 'Data View' window shows a table with columns 'values', 'mode', and 'pencils'. The 'pencils' column contains numerical values from 9 to 10.

```

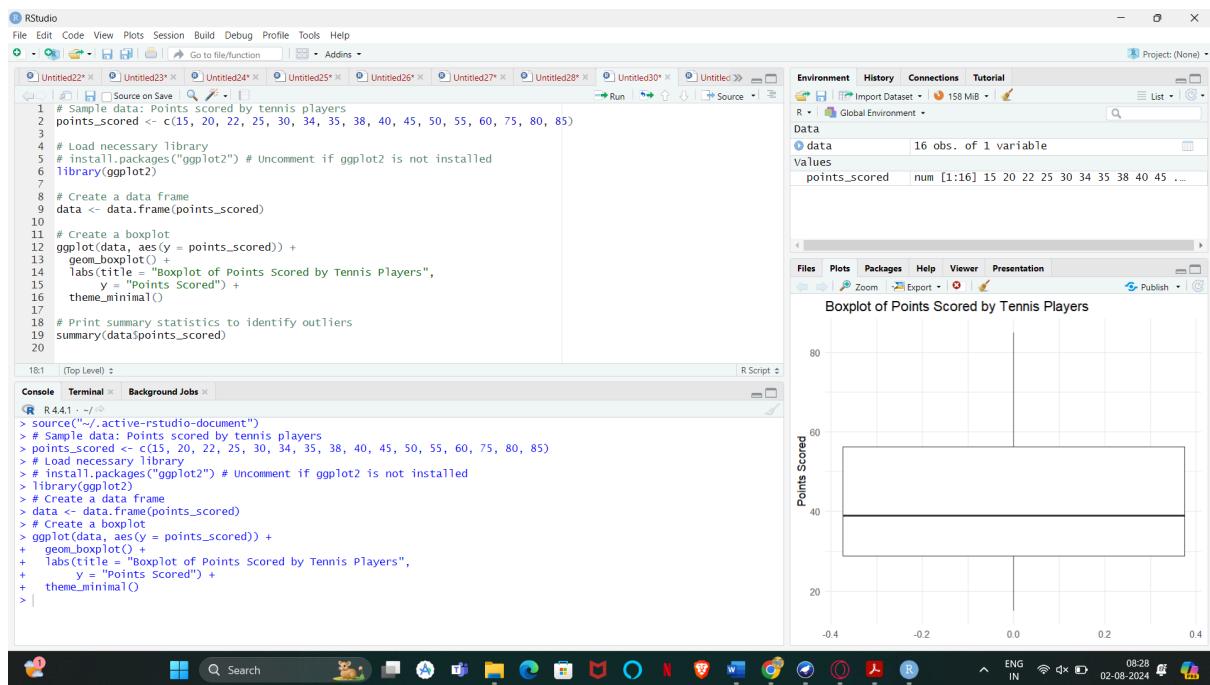
2 pencils<-c(9,25,23,12,11,6,7,8,9,10)
3 mean(pencils)
4 median(pencils)
5 mode<-names(table(pencils))[table(pencils)==max(table(pencils))]
6 mode
7

[1] 12
> mean(pencils)
[1] 9.5
> mode(pencils)
[1] "numeric"
> table(pencils)
pencils
 6 7 8 9 10 11 12 23 25
1 1 1 2 1 1 1 1 1
> |

```

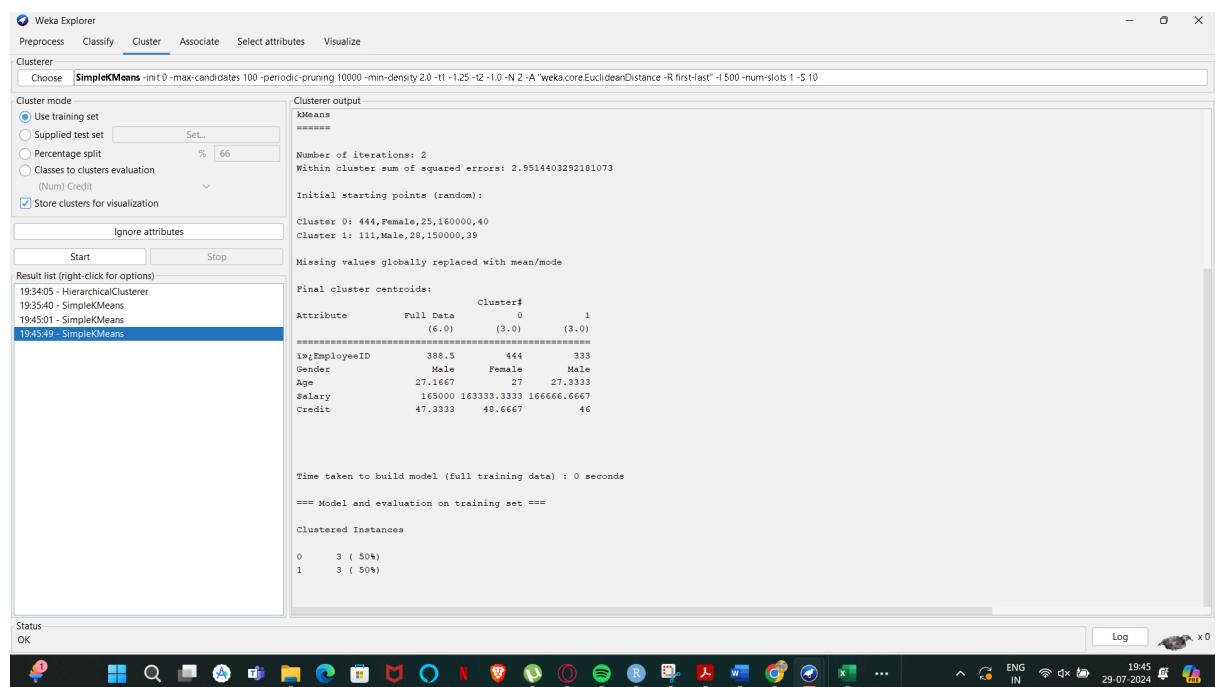
values	mode	pencils
	"9"	num [1:10] 9 25 23 12 11 6 7 8 9 10

22. Assume the Tennis coach wants to determine if any of his team players are scoring outliers. To visualize the distribution of points scored by his players, then how can he decide to develop the box plot? Give suitable example using Boxplot visualization technique.



23. Create the following dataset using CSV file format. To perform cluster analysis using K- Means in WEKA. To change the cluster size and plot the graph and illustrate the visualization of cluster.

EmployeeID	Gender	Age	Salary	Credit
111	Male	28	150000	39
222	Male	25	150000	27
333	Female	26	160000	42
444	Female	25	160000	40
555	Female	30	170000	64
666	Male	29	200000	72



24. Prediction of categorical data using Naïve Bayes classification through WEKA using any datasets. Compare the Naïve Bayes algorithm with SVM using the summary of results given by the classifiers and plot the graph.

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier Choose NaiveBayes
Test options
 Use training set
 Supplied test set Set...
 Cross-validation Folds 10
 Percentage split % 66
More options...
(Nom) class
Start Stop
Result list (right-click for options)
113403 - trees.DecisionStump
113747 - functions.Logistic
194909 - bayes.NaiveBayes
195021 - trees.RandomForest
Classifier output
precision 0.0000 0.0000
age
mean 31.2494 37.0808
std. dev. 11.6059 10.9146
weight sum 500 268
precision 1.1765 1.1765
Time taken to build model: 0.01 seconds
== Stratified cross-validation ==
== Summary ==
Correctly Classified Instances 586 76.3021 %
Incorrectly Classified Instances 182 23.6979 %
Kappa statistic 0.4664
Mean absolute error 0.2841
Root mean squared error 0.4168
Relative absolute error 62.5028 %
Root relative squared error 87.4349 %
Total Number of Instances 768
== Detailed Accuracy By Class ==
      TP Rate FP Rate Precision Recall F-Measure MCC ROC Area FRC Area Class
      0.844 0.388 0.802 0.944 0.823 0.468 0.819 0.892 tested_negative
      0.612 0.156 0.678 0.612 0.643 0.468 0.819 0.671 tested_positive
Weighted Avg. 0.763 0.307 0.759 0.763 0.760 0.468 0.819 0.815
== Confusion Matrix ==
      a b <-- classified as
      422 78 | a = tested_negative
      104 164 | b = tested_positive
Status OK Log x0

```

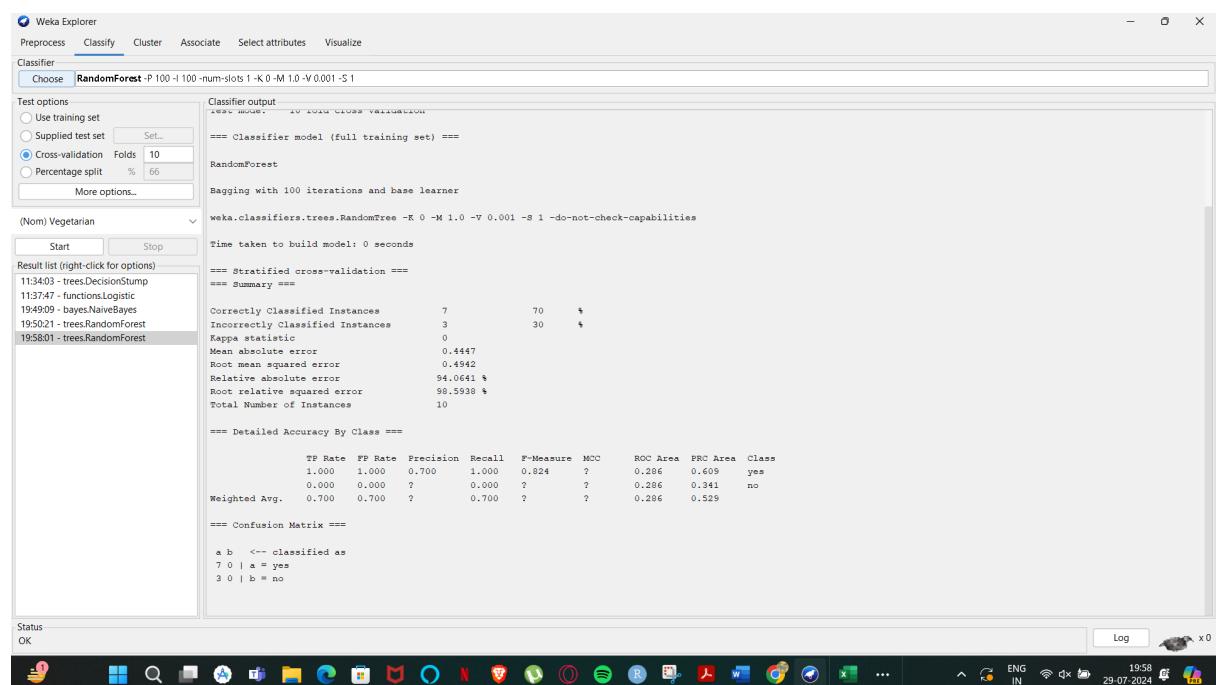
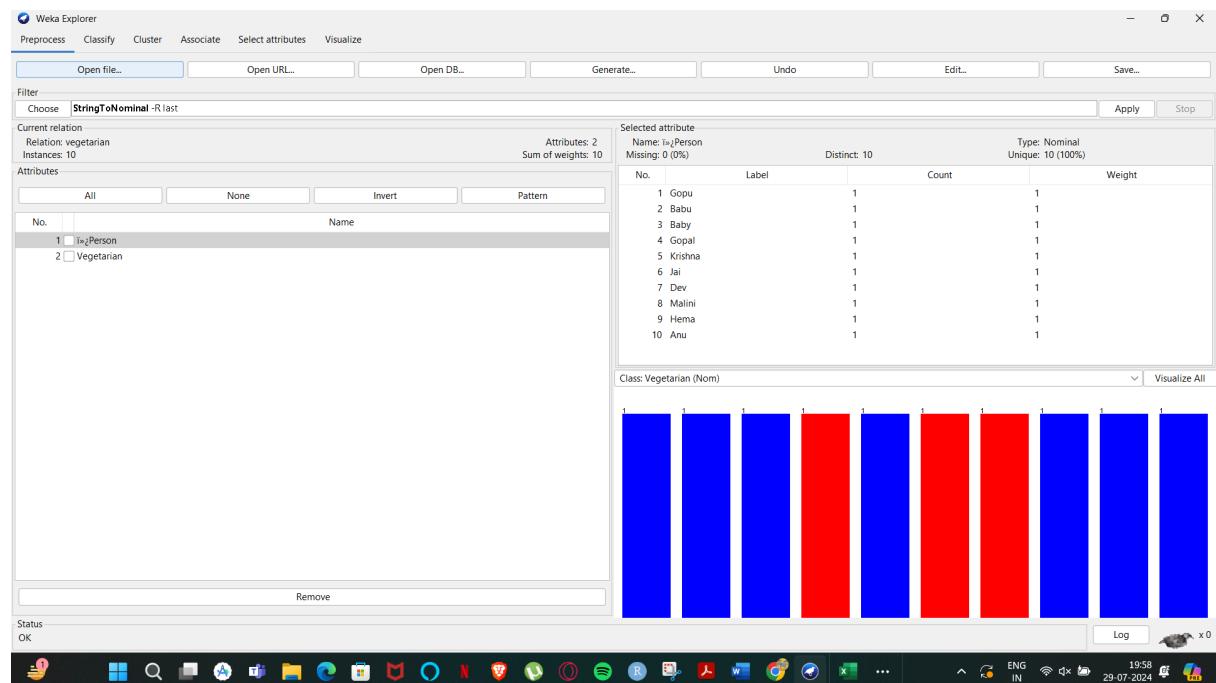
```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1
Test options
 Use training set
 Supplied test set Set...
 Cross-validation Folds 10
 Percentage split % 66
More options...
(Nom) class
Start Stop
Result list (right-click for options)
113403 - trees.DecisionStump
113747 - functions.Logistic
194909 - bayes.NaiveBayes
195021 - trees.RandomForest
Classifier output
weka mode: -I 100 CROSS VALIDATION
== Classifier model (full training set) ==
RandomForest
Bagging with 100 iterations and base learner
weka.classifiers.trees.RandomTree -R 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities
Time taken to build model: 0.22 seconds
== Stratified cross-validation ==
== Summary ==
Correctly Classified Instances 582 75.7813 %
Incorrectly Classified Instances 186 24.2188 %
Kappa statistic 0.4566
Mean absolute error 0.3106
Root mean squared error 0.4031
Relative absolute error 68.3405 %
Root relative squared error 84.5604 %
Total Number of Instances 768
== Detailed Accuracy By Class ==
      TP Rate FP Rate Precision Recall F-Measure MCC ROC Area FRC Area Class
      0.836 0.388 0.901 0.936 0.818 0.458 0.820 0.886 tested_negative
      0.612 0.164 0.667 0.612 0.638 0.458 0.820 0.679 tested_positive
Weighted Avg. 0.758 0.310 0.754 0.758 0.755 0.458 0.820 0.814
== Confusion Matrix ==
      a b <-- classified as
      418 82 | a = tested_negative
      104 164 | b = tested_positive
Status OK Log x1

```

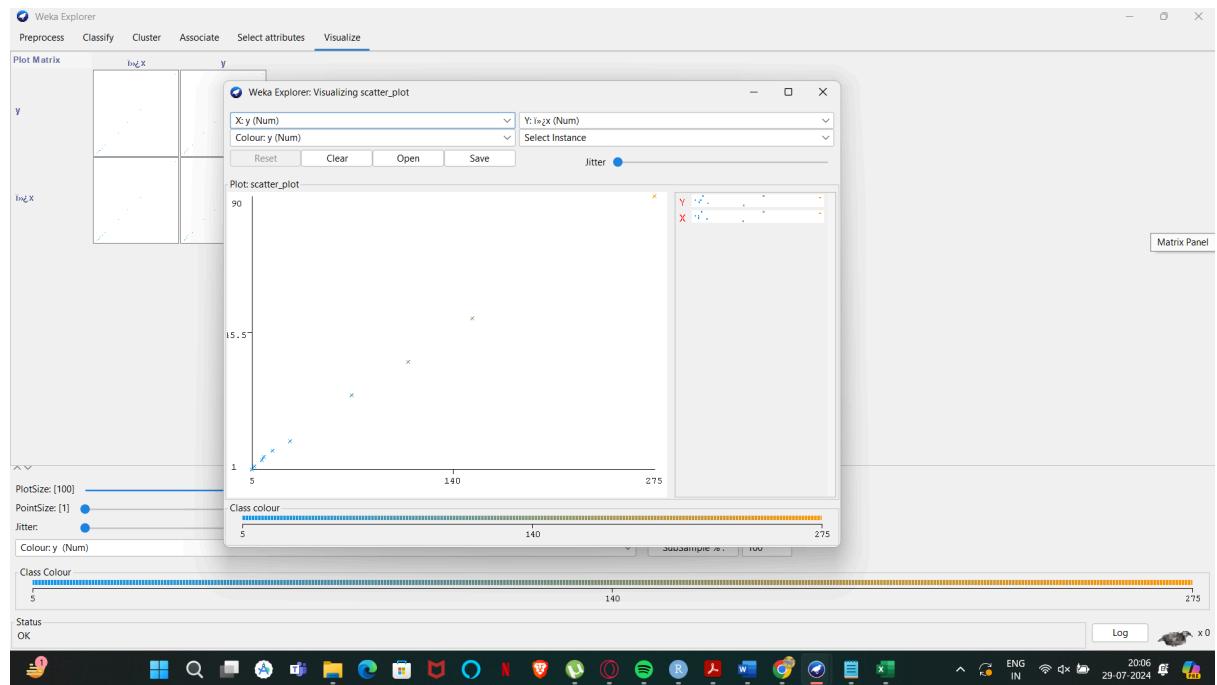
25. The following list of persons with vegetarian or not details given in the table. How will you find out how many of them are vegetarian and how many of them are non-vegetarian? Which type of the person total count is greater value?

Person	Gopu	Babu	Baby	Gopal	Krishna	Jai	Dev	Malini	Hema	Anu
Vegetarian	yes	yes	yes	no	yes	no	no	yes	yes	yes



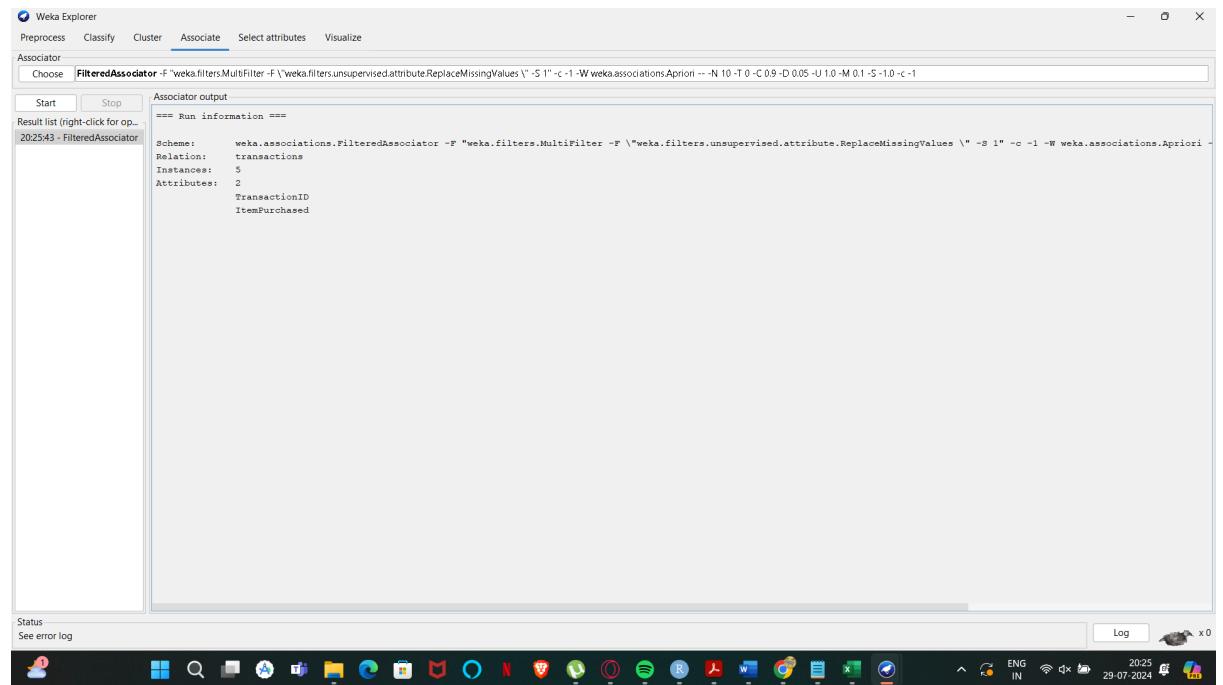
26. The following table would be plotted as (x,y) points, with the first column being the x values as number of mobile phones sold and the second column being the y values as money. To use the scatter plot for how many mobile phones sold.

x	4	1	5	7	10	2	50	25	90	36
y	12	5	13	19	31	7	153	72	275	110

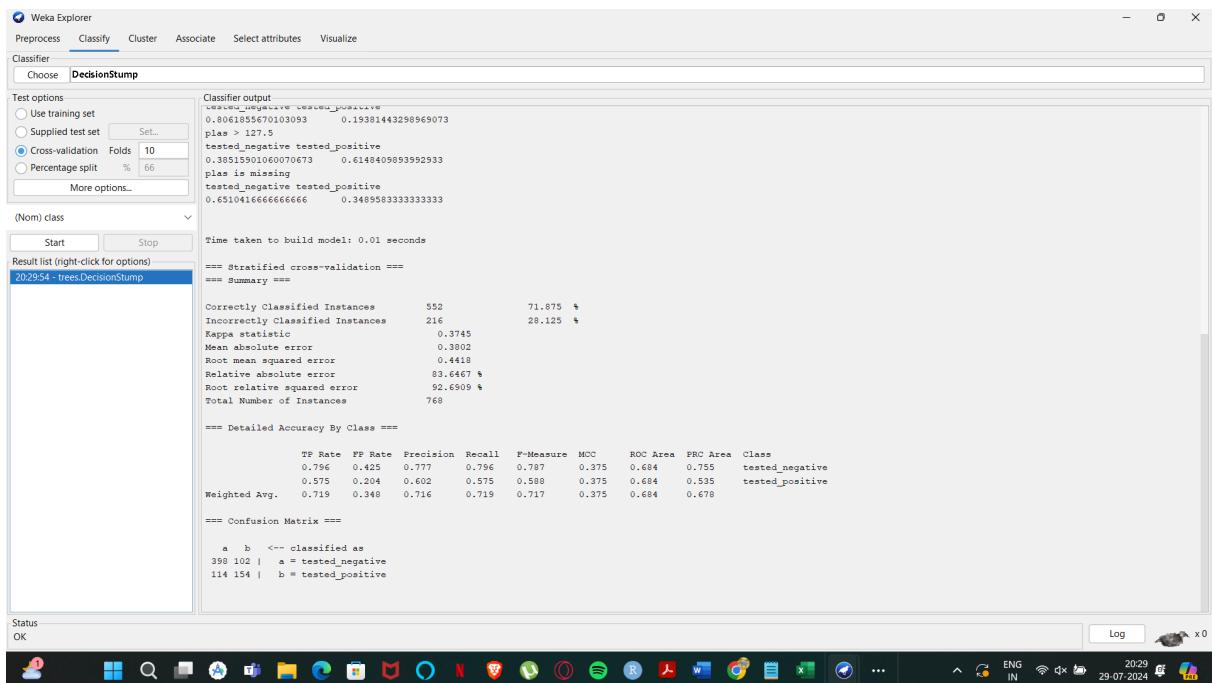


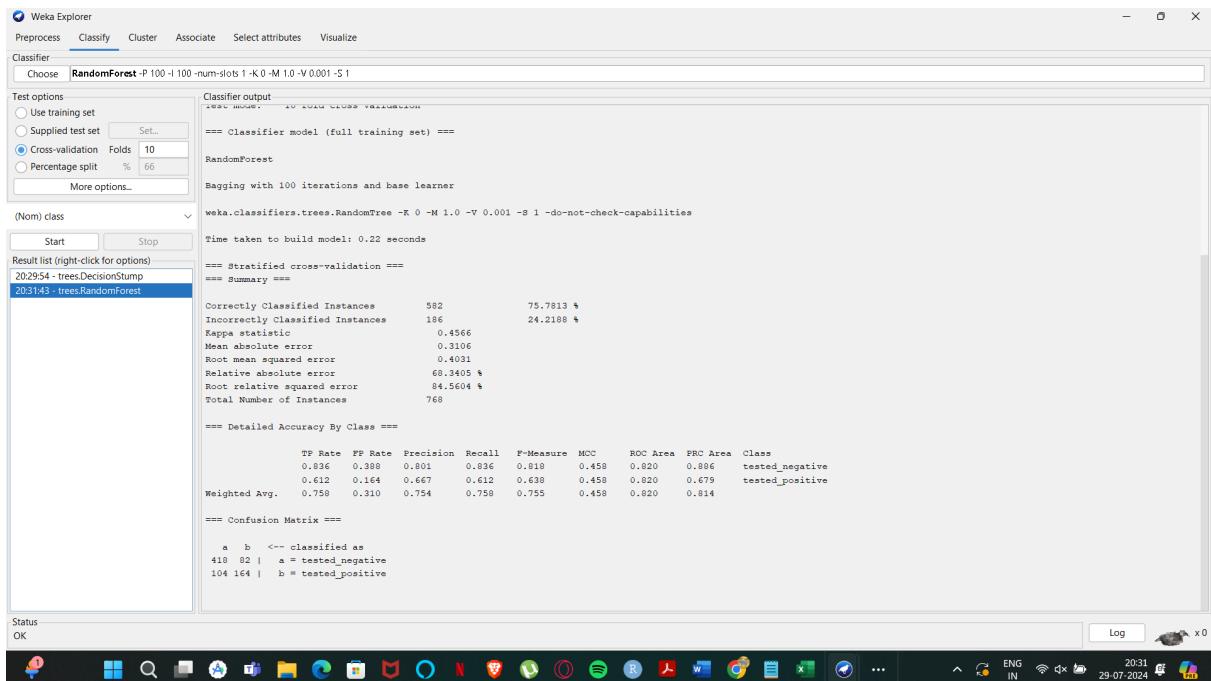
27. Generate rules using FP growth algorithm using the given dataset which has the following transactions with items purchased: Consider the values as support=50% and confidence=75%.

Transaction ID	Items Purchased
1	Bread, Cheese, Egg, Juice
2	Bread, Cheese, Juice
3	Bread, Milk, Yogurt
4	Bread, Juice, Milk
5	Cheese, Juice, Milk



28. Prediction of Diabetes Data using Decision tree classifier in WEKA. Compare it with Support Vector Machine classifier. Show the result accuracy and F1 measure calculation .Plot the graph and explain the summary of results.



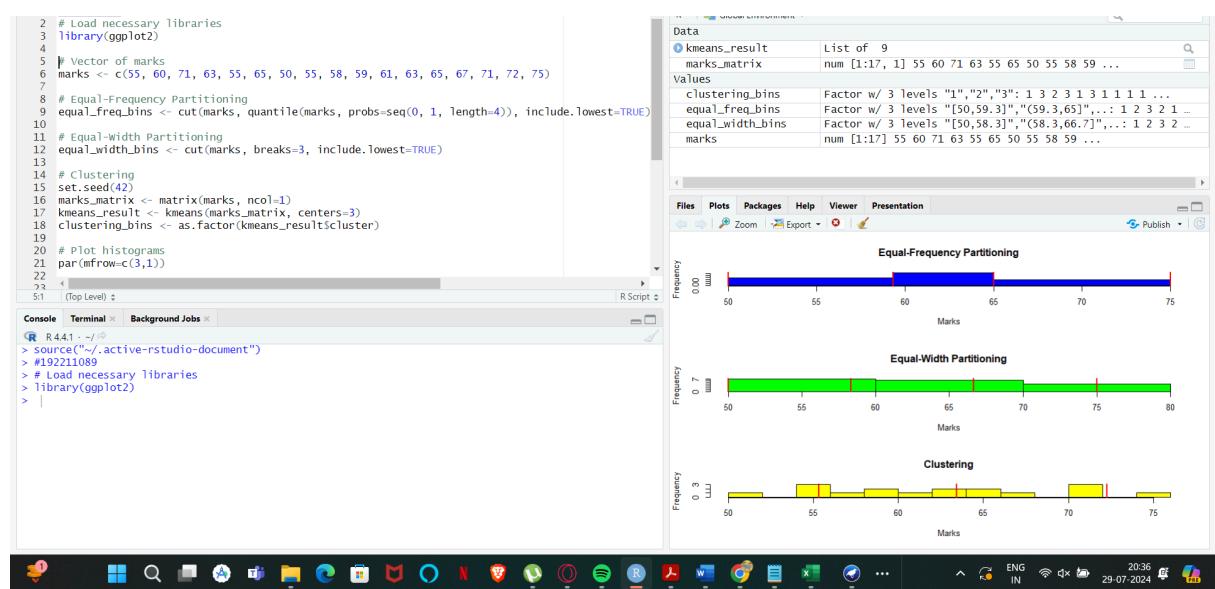


29. Implement of the R script using marks scored by a student in his model exam has been sorted as follows: 55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65, 67, 71, 72, 75. Partition them into three bins by each of the following methods. Plot the data points using histogram.

(a) equal-frequency (equi-depth) partitioning

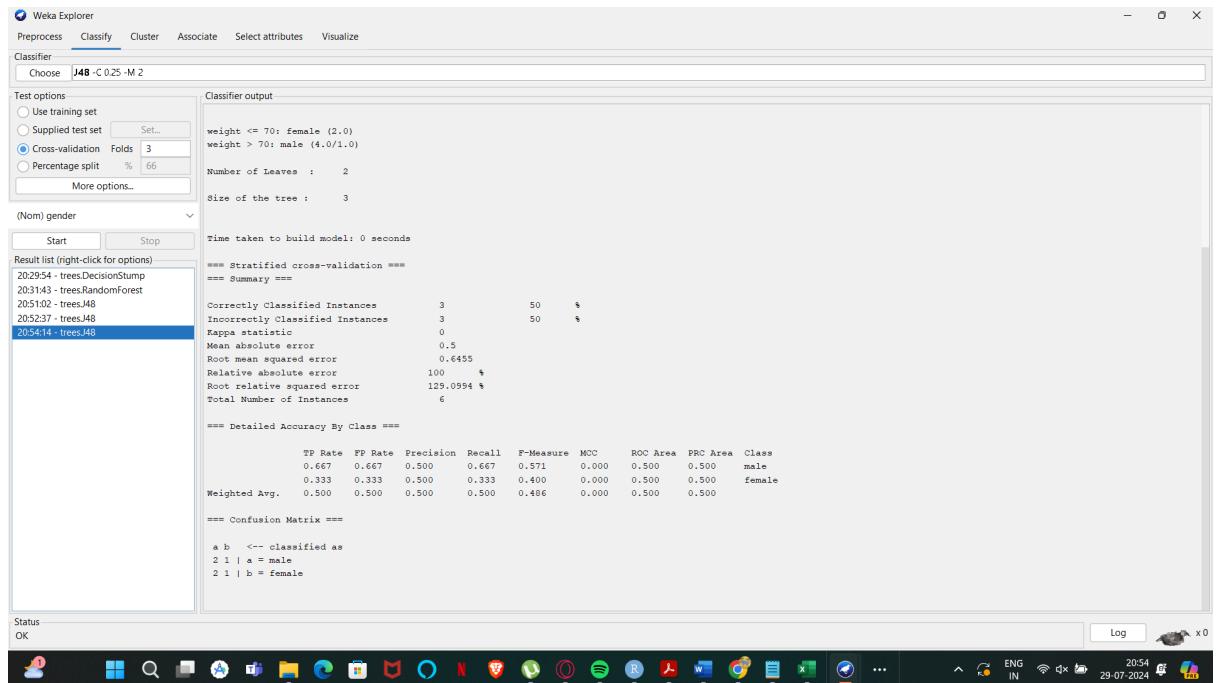
(b) equal-width partitioning

(c) clustering



30. Consider this Decision tree :

- create the data set for the below tree using ARFF format and calculate accuracy and decision for the same
- Using this decision tree generate the rules based on rule based induction.
- Compare both the algorithms and plot the confusion matrix.



31. Create an ARFF file for the table below and implement for the Apriori Algorithm and FP growth algorithm and compare the rules generated by both the algorithms. Identify the unique rules generated by the above algorithms.

T.ID	ITEMS
T1	SONY, BPL, LG
T2	BPL, SAMSUNG
T3	BPL, ONIDA
T4	SONY, BPL, SAMSUNG
T5	SONY, ONIDA
T6	BPL, ONIDA
T7	SONY, ONIDA
T8	SONY, BPL, ONIDA, LG
T9	SONY, BPL, ONIDA

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Associate
Choose Apriori-N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1
Start Stop
Associate output
Result list (right-click for)
205944 - Apriori
Schema: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S 1.0 -c 1
Relation: electronics
Instances: 9
Attributes: 2
ix2T.ID
ITEMS
==== Associator model (full training set) ====
Apriori
=====
Minimum support: 0.16 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17
Generated sets of large itemsets:
Size of set of large itemsets L(1): 16
Size of set of large itemsets L(2): 9
Best rules found:
1. ITEMS=SONY, BFL, LG 1 ==> ix2T.ID=P1 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
2. ix2T.ID=P1 1 ==> ITEMS=SONY, BFL, LG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
3. ITEMS=BPL, SAMSUNG 1 ==> ix2T.ID=P2 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
4. ix2T.ID=P2 1 ==> ITEMS=BPL, SAMSUNG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
5. ix2T.ID=P3 1 ==> ITEMS=BPL, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
6. ix2T.ID=P4 1 ==> ITEMS=SONY, BFL, SAMSUNG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
7. ix2T.ID=P5 1 ==> ITEMS=SONY, BFL, SAMSUNG 1    <conf:(1)> lift:(9) lev:(0.1) [0] conv:(0.89)
8. ix2T.ID=P6 1 ==> ITEMS=SONY, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
9. ix2T.ID=P7 1 ==> ITEMS=BPL, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)
10. ix2T.ID=P8 1 ==> ITEMS=SONY, CNIDA 1    <conf:(1)> lift:(4.5) lev:(0.09) [0] conv:(0.78)

```

32. The given are the strike-rates scored by a batsman in season 1 in different tournaments. 100, 70, 60, 90, 90

(a) min-max normalization by setting min = 0 and max = 1

(b) z-score normalization

(c) z-score normalization using the mean absolute deviation instead of standard deviation

(d) normalization by decimal scaling

```

2 # Define the strike rates
3 strike_rates <- c(100, 70, 60, 90, 90)
4
5 # Min-Max Normalization
6 min_max_normalized <- (strike_rates - min(strike_rates)) / (max(strike_rates) - min(strike_rates))
7 print("Min-Max Normalization:")
8 print(min_max_normalized)
9
10 # Z-Score Normalization
11 mean_sr <- mean(strike_rates)
12 sd_sr <- sd(strike_rates)
13 z_score_normalized <- (strike_rates - mean_sr) / sd_sr
14 print("Z-Score Normalization:")
15 print(z_score_normalized)
16
17 # Z-Score Normalization using Mean Absolute Deviation
18 mad_sr <- mean(abs(strike_rates - mean_sr))
19 z_score_mad_normalized <- (strike_rates - mean_sr) / mad_sr
20 print("Z-Score Normalization using Mean Absolute Deviation:")
21 print(z_score_mad_normalized)
22
23
24
25
26
27
28
29
29

```

values
decimal_scaling_norm_num [1:5] 1 0.7 0.6 0.9 0.9
j 2
mad_sr 13.6
max_abs_sr 100
mean_sr 82
min_max_normalized num [1:5] 1 0.25 0 0.75 0.75
sd_sr 16.431676725155
strike_rates num [1:5] 100 70 60 90 90
z_score_mad_normalized num [1:5] 1.324 -0.882 -1.618 0.588 0.588
z_score_normalized num [1:5] 1.095 -0.73 -1.339 0.487 0.487

33. Suppose some car is tested for the AvgSpeed and TotalTime data for 9 randomly selected car with the following result

a) Calculate the standard deviation of AvgSpeed and TotalTime.

b) Calculate the Variance of AvgSpeed and TotalTime for the above dataset.

AvgSpeed (in kph)	78	81	82	74	83	82	77	80	70
TotalTime (in mins)	39	37	36	42	35	36	40	38	46

The screenshot shows the RStudio interface with the following details:

- Code Editor (Left):** Contains R code for calculating statistics on the provided dataset.
- Console (Bottom Left):** Shows the execution of the script and the output of standard deviation and variance calculations for both AvgSpeed and TotalTime.
- Environment (Top Right):** Displays the values of the variables used in the calculations.
- Plots (Bottom Right):** Shows a small preview of a plot with an 'abc' logo.
- System Tray (Bottom):** Includes icons for battery, signal, and system status.

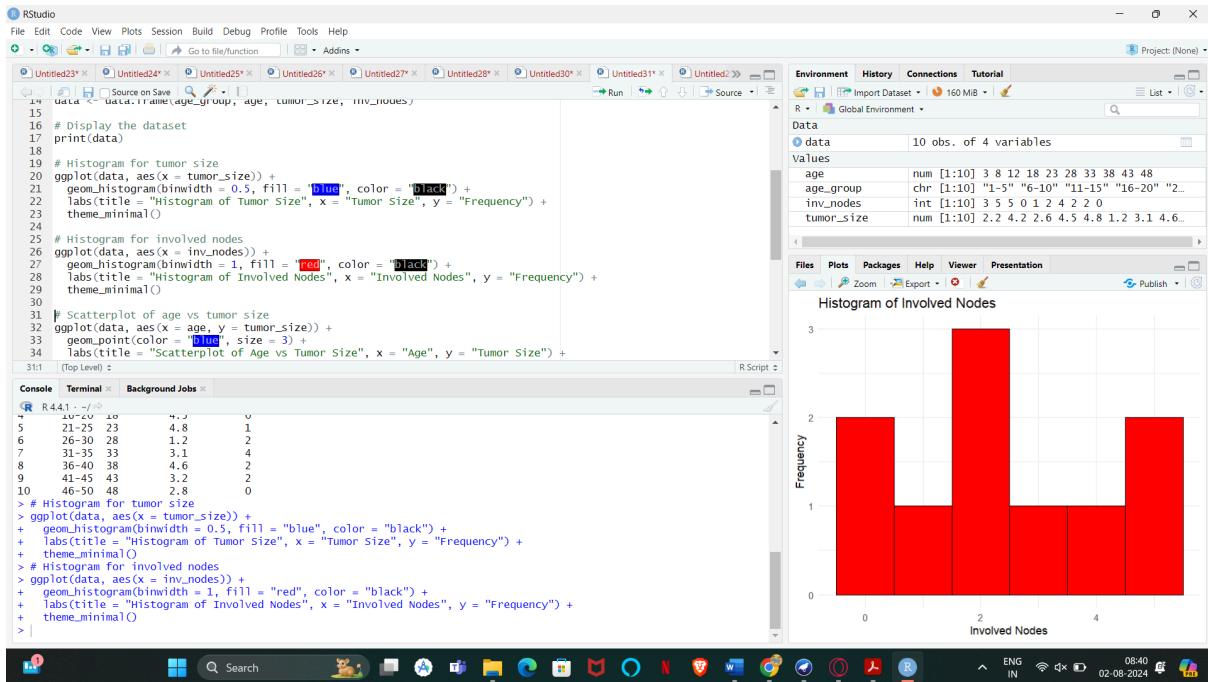
```

3 AvgSpeed <- c(78, 81, 82, 74, 83, 82, 77, 80, 70)
4 TotalTime <- c(39, 37, 36, 42, 35, 36, 40, 38, 46)
5 # Calculate standard deviation
6 sd_AvgSpeed <- sd(AvgSpeed)
7 sd_TotalTime <- sd(TotalTime)
8
9 # Calculate variance
10 var_AvgSpeed <- var(AvgSpeed)
11 var_TotalTime <- var(TotalTime)
12
13 # Print the results
14 cat("Standard Deviation of AvgSpeed:", sd_AvgSpeed, "\n")
15 cat("Standard Deviation of TotalTime:", sd_TotalTime, "\n")
16
17 cat("Variance of AvgSpeed:", var_AvgSpeed, "\n")
18 cat("Variance of TotalTime:", var_TotalTime, "\n")
19
20
  
```

values	
AvgSpeed	num [1:9] 78 81 82 74 83 82 77 80 70
sd_AvgSpeed	4.30439052338165
sd_TotalTime	3.4920547329283
TotalTime	num [1:9] 39 37 36 42 35 36 40 38 46
var_AvgSpeed	18.527777777778
var_TotalTime	12.194444444444

34. Consider a person want to take a census / plot for the breast-cancer affected people through the years. Create a own dataset with this parameters age, tumorsize,inv-nodes [example between age 1-5 = no.of.count, 6-10=no.of.count,etc]

Draw the Histogram, scatterplot,boxplot.



35. Create the Confusion matrix using this scenario:

A shepherd boy gets bored tending the town's flock. To have some fun, he cries out, "Wolf!" even though no wolf is in sight. The villagers run to protect the flock, but then get really mad when they realize the boy was playing a joke on them. One night, the shepherd boy sees a real wolf approaching the flock and calls out, "Wolf!" The villagers refuse to be fooled again and stay in their houses. The hungry wolf turns the flock into lamb chops. The town goes hungry. Panic ensues.

True Positive (TP):

- Reality: A wolf threatened.
- Shepherd said: "Wolf."
- Outcome: Shepherd is a hero.

False Positive (FP):

- Reality: No wolf threatened.
- Shepherd said: "Wolf."
- Outcome: Villagers are angry at shepherd for waking them up.

False Negative (FN):

- Reality: A wolf threatened.
- Shepherd said: "No wolf."
- Outcome: The wolf ate all the sheep.

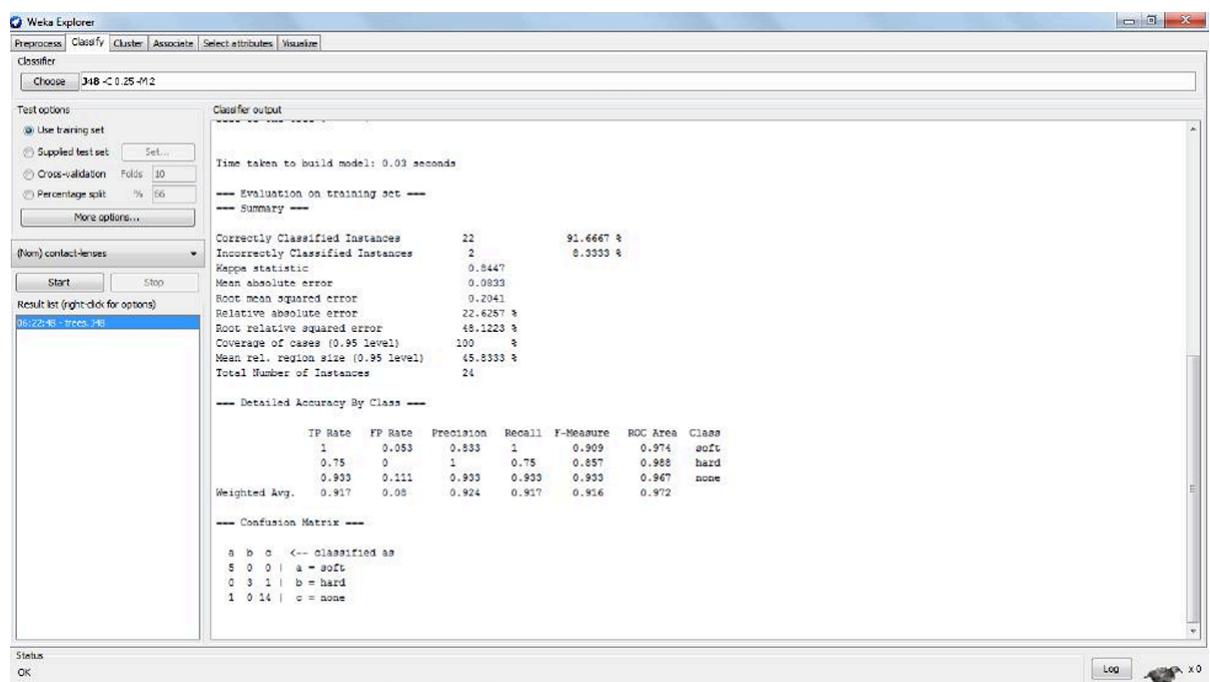
True Negative (TN):

- Reality: No wolf threatened.
- Shepherd said: "No wolf."
- Outcome: Everyone is fine.

36. Create the ARFF data set for the below mentioned dataset perform the bayes theorem in addition to that compare the same with decision tree. identify the effiecient classifier with accuracy with F1 Score.

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



37.

- a) Suppose that the “Diabetes data set ” data for analysis includes the attribute age. The age values for the data are (in increasing order) 30, 57, 68, 96, 39, 40, 20, 19, 42, 12, 25, 25, 65, 35, 30, 23, 23, 35, 45, 85. What is the mean?

b) Suppose that the speed car is mentioned in different driving style.

Regular Speed	78.3	81.8	82	74.2	83.4	84.5	82.9	77.5	80.9	70.6
---------------	------	------	----	------	------	------	------	------	------	------

```

1 # a) Calculation of Mean Age
2 ages <- c(30, 57, 68, 96, 39, 40, 20, 19, 42, 12, 25, 25, 65, 35, 30, 23, 23, 35, 45, 85)
3 mean_age <- mean(ages)
4 print(paste("Mean Age:", mean_age))
5
6 # b) Calculation of IQR and Standard Deviation for Car Speeds
7 car_speeds <- c(78.3, 81.8, 82, 74.2, 83.4, 84.5, 82.9, 77.5, 80.9, 70.6)
8
9 iqr_speeds <- IQR(car_speeds)
10 std_dev_speeds <- sd(car_speeds)
11
12 print(paste("Interquartile Range (IQR) of Car Speeds:", iqr_speeds))
13 print(paste("Standard Deviation of Car Speeds:", std_dev_speeds))
14

```

14:1 (Top Level) R Script

```

R 4.4.1 · ~/active/studio-document
> source("~/active/studio-document")
[1] "Mean Age: 40.7"
[1] "Interquartile Range (IQR) of Car Speeds: 4.97500000000001"
[1] "Standard Deviation of Car Speeds: 4.44583450484208"
> # a) Calculation of Mean Age
> ages <- c(30, 57, 68, 96, 39, 40, 20, 19, 42, 12, 25, 25, 65, 35, 30, 23, 23, 35, 45, 85)
> mean_age <- mean(ages)
> print(paste("Mean Age:", mean_age))
[1] "Mean Age: 40.7"
> # b) Calculation of IQR and Standard Deviation for Car Speeds
> car_speeds <- c(78.3, 81.8, 82, 74.2, 83.4, 84.5, 82.9, 77.5, 80.9, 70.6)
> iqr_speeds <- IQR(car_speeds)
> std_dev_speeds <- sd(car_speeds)
> print(paste("Interquartile Range (IQR) of Car Speeds:", iqr_speeds))
[1] "Interquartile Range (IQR) of Car Speeds: 4.97500000000001"
> print(paste("Standard Deviation of Car Speeds:", std_dev_speeds))
[1] "Standard Deviation of Car Speeds: 4.44583450484208"
>

```

Console Terminal Background Jobs

Windows Taskbar: Search, File Explorer, Edge, Netflix, etc.

38) a) Let us consider one example to make the calculation method clear. Assume that the minimum and maximum values for the feature F are \$50,000 and \$100,000 correspondingly. It needs to range F from 0 to 1. In accordance with min-max normalization, $v = \$80$,

a) Use the two methods below to normalize the following group of data: 200, 300, 400, 600, 1000

b) min-max normalization by setting min = 0 and max = 1

c) z-score normalization

```

1 # Given data
2 data <- c(200, 300, 400, 600, 1000)
3
4 # a) Min-Max Normalization
5 min_d <- min(data)
6 max_d <- max(data)
7 min_max_normalized <- (data - min_d) / (max_d - min_d)
8 print("Min-Max Normalized Data:")
9 print(min_max_normalized)
10
11 # b) Z-Score Normalization
12 mean_d <- mean(data)
13 std_dev_d <- sd(data)
14 z_score_normalized <- (data - mean_d) / std_dev_d
15 print("Z-Score Normalized Data:")
16 print(z_score_normalized)
17

```

17:1 (Top Level) R Script

```

R 4.4.1 · ~
> # Given data
> data <- c(200, 300, 400, 600, 1000)
> # a) Min-Max Normalization
> min_d <- min(data)
> max_d <- max(data)
> min_max_normalized <- (data - min_d) / (max_d - min_d)
> print("Min-Max Normalized Data:")
[1] "Min-Max Normalized Data:"
[1] 0.000 0.125 0.250 0.500 1.000
> # b) Z-Score Normalization
> mean_d <- mean(data)
> std_dev_d <- sd(data)
> z_score_normalized <- (data - mean_d) / std_dev_d
> print("Z-Score Normalized Data:")
[1] "Z-Score Normalized Data:"
> print(z_score_normalized)
[1] -0.9486835 -0.6324555 -0.3162278 0.3162278 1.5811388

```

Console Terminal Background Jobs

Windows Taskbar: Search, File Explorer, Edge, Netflix, etc.

39. Consider this table

TID items bought

T100 {M, O, N, K, E, Y}

T200 {D, O, N, K, E, Y }

T300 {M, A, K, E}

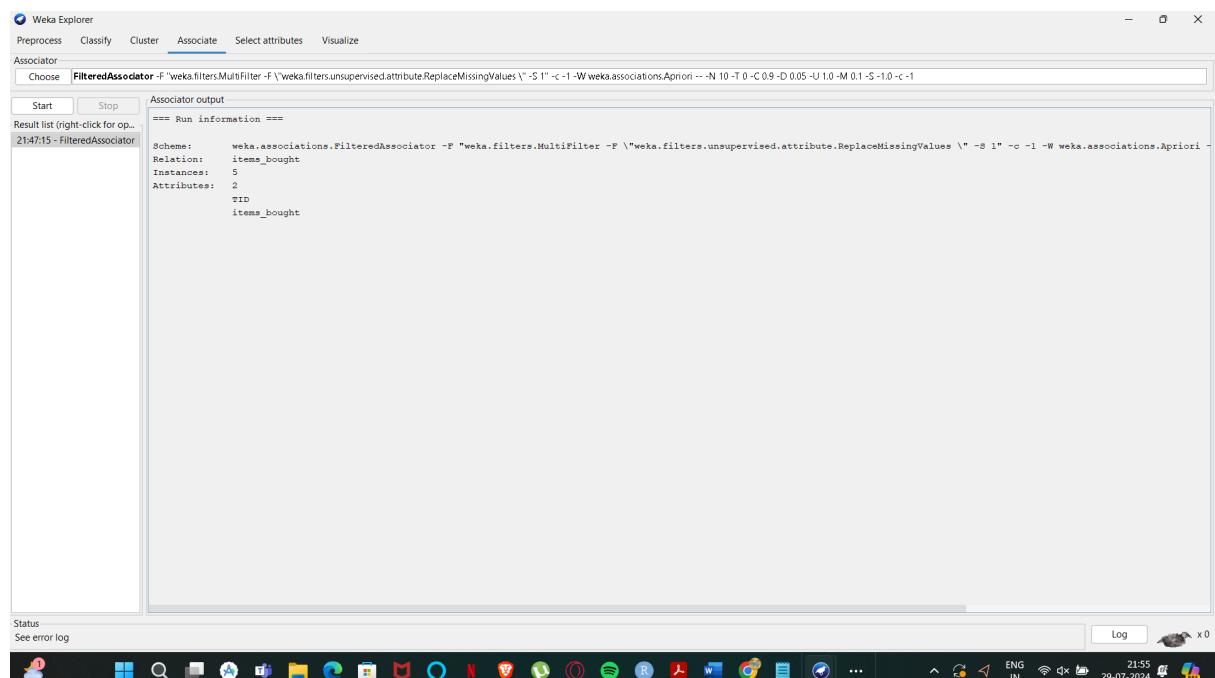
T400 {M, U, C, K, Y}

T500 {C, O, O, K, I ,E}

(a) Find all frequent item set using Apriori and FP-growth, respectively. Compare the efficiency of the two mining processes.

(b) List all of the strong association rules (with support s and confidence c) matching the following metarule, where X is a variable representing customers, and itemi denotes variables representing items (e.g., “A”, “B”, etc.):

$$\forall x \in \text{transaction}, \text{buys}(x, \text{item}1) \wedge \text{buys}(x, \text{item}2) \Rightarrow \text{buys}(x, \text{item}3)$$



40. Suppose we want to classify potential bank customers as good creditors or bad creditors for loan applications. We have a training dataset describing past customers using the following attributes: Marital status {married, single, divorced}, Gender {male, female}, Age {[18..30[, [30..50[, [50..65[, [65+]}], Income {[10K..25K[, [25K..50K[, [50K..65K[, [65K..100K[, [100K+]}}]. Using Weka tool solve this problem.

