

CSA1667-Data Warehousing And Data Mining

DAY-1

Q1:

```
R version 4.4.1 (2024-06-14 ucrt) -- "Race for Your Life"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
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.

> #age, frequency
> age<-c(5,15,20,50,80,110)
> frequency<-c(200,450,300,1500,700,44)
Error in frequency <- c(200, 450, 300, 1500, 700, 44) :
  comparison (<=) is possible only for atomic and list types
> frequency<-c(200,450,300,1500,700,44)
> median(frequency)
[1] 375
> median(age)
[1] 35
>
```

Q2:

```
File Edit Code View Plots Session Build Debug Profile Tools Help
Source
Console Terminal Background Jobs
R 4.4.1 - ~/r
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.

> #mean,median,mode,quartile
> 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)
> median
function (x, na.rm = FALSE, ...)
UseMethod("median")
<bytecode: 0x0000026ce77122e8>
<environment: namespace:stats>
> #mean,median,mode,quartile
> 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)
> mean(age)
[1] 29.96296
> median(age)
[1] 23
> mode.age<-names(table(age))[table(age)==max(table(age))]
> mode.age
[1] "25" "35"
> range(age)
[1] 13 70
> quantile(age,.25)
25%
20.5
> quantile(age,.75)
75%
35
>
```

Q3:

The screenshot shows an RStudio interface with a script file named 'Untitled1.R'. The code calculates a z-score based on a vector 'v' and prints the decimal scaling of the result. The output in the console shows the calculation steps and the final result.

```
v<-c(200,300,400,600,1000)
m<-0
max<-1
#min,max
min_max<-(35-min(v))/(max(v)-min(v))
print(min_max)
#z-score
m<-mean(v)
s<-12.94
z_score<-(35-m)/s
#decimal scaling
decimal_scaling
m<-35
j<-max(m)<1
decimal_scaling<-m/10^j
print(decimal_scaling)
[1] 35
```

The environment pane shows variables: decimal_scaling = 35, j = FALSE, m = 35, max = 1, min = 0, min_max = -0.20625, s = 12.94, v = num [1:5] 200 300 400 600 1000, and z_score = -35.035085007728.

Q4:

The screenshot shows an RStudio interface with a script file named 'Untitled1.R'. The code demonstrates the use of the 'mean_smooth' function from the 'dplyr' package to smooth a dataset 'data'. It shows how to create bins, calculate mean values for each bin, and then smooth the data by applying a median across bins.

```
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)
bin<-5
bin_indices<-cut(data,bins)
mean_smooth<-tapply(data,bin_indices,mean)
Error: object 'bin_indices' not found
mean_smooth<-tapply(data,bin_indices,mean)
Error: object 'bin_indices' not found
mean_smooth<-tapply(data,bin_indices,mean)
mean_smooth<-tapply(data,bin_indices,mean)
mean_smooth<-tapply(data,bin_indices,mean)
(10.9,23.8] (23.8,36.6] (36.6,49.4] (49.4,62.2]
17.78571 27.00000 43.75000 NA
(62.2,75.1]
72.75000
median_smooth<-tapply(data,bin_indices,median)
print(median_smooth)
(10.9,23.8] (23.8,36.6] (36.6,49.4] (49.4,62.2]
27.0 45.0 NA
(62.2,75.1]
72.5
min_max_smooth<-tapply(data,bin_indices,function(x)c(min(x),max(x)))
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
\$ (49.4,62.2]
NULL
\$ (62.2,75.1]
[1] 71 75
```

The environment pane shows variables: bin_indices = Factor w/ 5 levels "(10.9,23.8]", ..., bins = 5, data = num [1:24] 11 13 13 15 15 16 19 20 20 ..., mean_smooth = num [1:5] 17.8 27 43.8 NA 72.8, and median_smooth = num [1:5] 27 45 NA 72.5.

Q5:

RStudio interface showing a scatter plot of age versus fat. The x-axis is labeled 'age' and ranges from 30 to 60. The y-axis is labeled 'fat' and ranges from 10 to 40. The plot shows a positive correlation.

```
R 4.4.1 --> 
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

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.

> age<-c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
> fat<-c(9.5,26.5,7.8,17.8,31.4,25.9,27.4,27.2,31.2,34.6,42.5,28.8,33.4,30.2,34.1,32.9,41.2,35.7)
> mean(age)
[1] 46.44444
> median(age)
[1] 51
> sd(age)
[1] 13.21862
> mean(fat)
[1] 28.78333
> median(fat)
[1] 30.7
> sd(fat)
[1] 9.34398
> #bxpplot(age,fat)
> #bxpplot(age,fat)
> #bxpplot(age,fat)
> #scatter plot
> scatter.smooth(age,fat)
> #qplot
> #qplot
> #qplot(age,fat)
> |
```

29°C Mostly cloudy ENG INTL 10:38 18-07-2024

Q6:

RStudio interface showing a scatter plot of v versus z_score. The x-axis is labeled 'v' and ranges from 0 to 35. The y-axis is labeled 'z_score' and ranges from -0.88 to 0.31. The plot shows a negative correlation.

```
R 4.4.1 (2024-06-14 ucrt) -- "Race for Your Life"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
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.

> vc<-c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
> min<-0
> max<-1
> #min_max
> min_max<-(35-min(v))/(max(v)-min(v))
> print(min_max)
[1] 0.3157895
> j<-1
> m<-mean(v)
> s<-12.94
> z_score<-(35-m)/s
> print(z_score)
[1] -0.8844238
> #decimal scaling
> v<-35
> j<-max(m)<1
> decimal_scaling<-m/10^j
> print(decimal_scaling)
[1] 35
>
```

29°C Mostly cloudy ENG INTL 10:42 18-07-2024

Q7:

```
R version 4.4.1 (2024-06-14 ucrt) -- "Race for Your Life"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

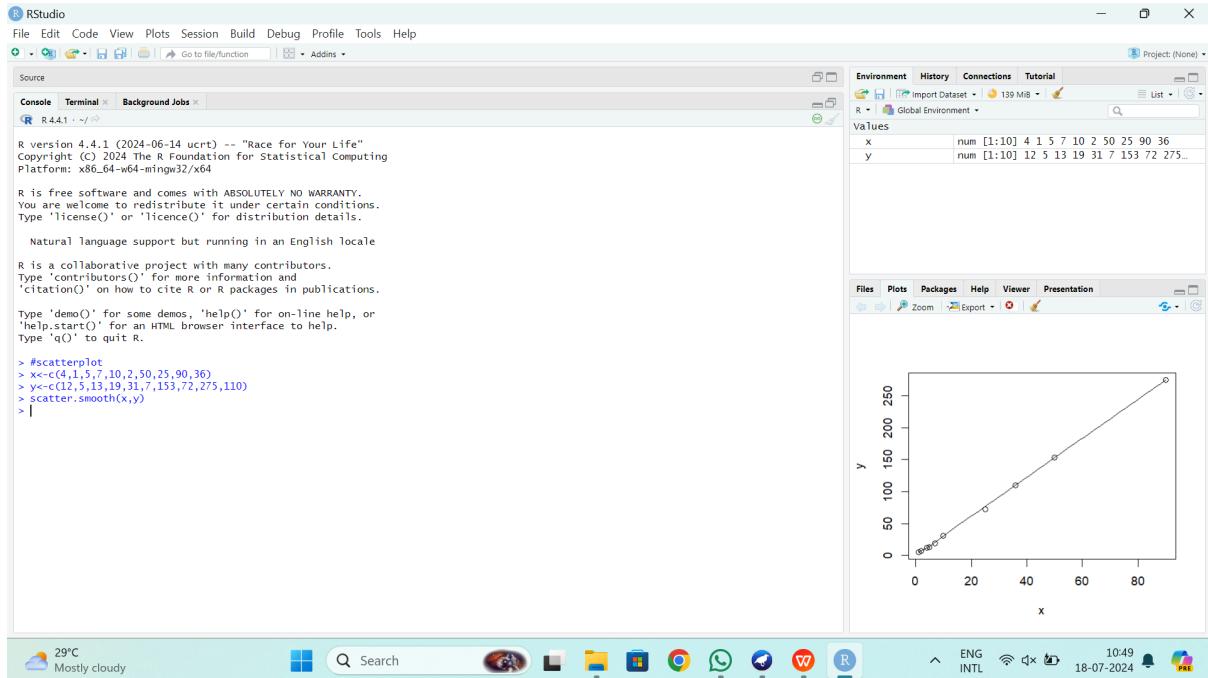
Natural language support but running in an English locale

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.

> pencils<-c(9,25,23,12,11,6,7,8,9,10)
> mean(pencils)
[1] 12
> median(pencils)
[1] 9.5
> x <- names(table(pencils))[table(pencils)==max(table(pencils))]
> mode
[1] "9"
>
```

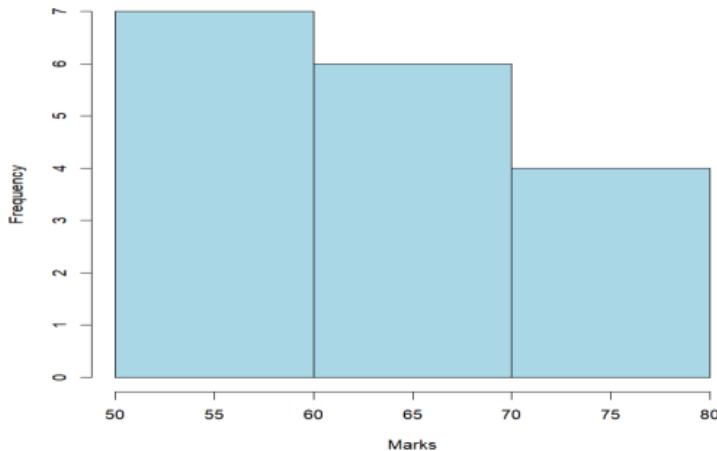
Q8:



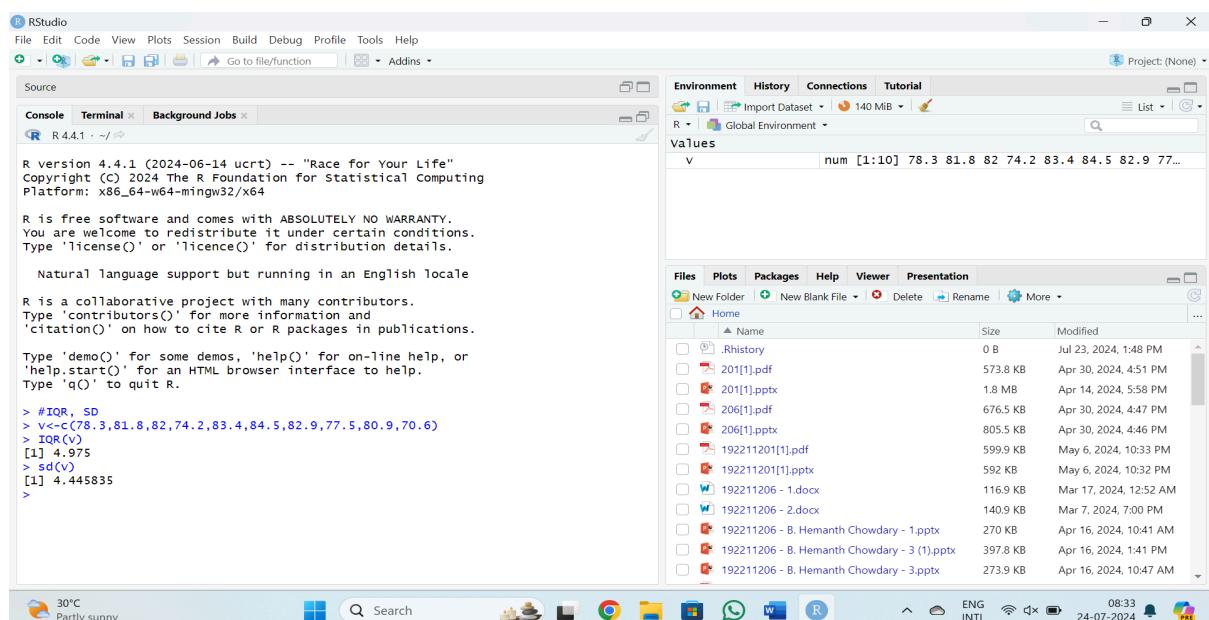
Q9:

```
> print(smoothed_data_by_mean)
[1] 18.9375 18.9375 18.9375 18.9375 18.9375 18.9375 18.9375 18.9375
[9] 18.9375 18.9375 18.9375 18.9375 18.9375 18.9375 18.9375 18.9375
[17] 43.7500 43.7500 43.7500 43.7500 43.7500 72.7500 72.7500 72.7500
> print("Smoothed data by bin median:")
[1] "Smoothed data by bin median:"
> print(smoothed_data_by_median)
[1] 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
[14] 20.0 20.0 20.0 45.0 45.0 45.0 72.5 72.5 72.5 72.5
> print("Smoothed data by bin boundaries:")
[1] "Smoothed data by bin boundaries:"
> print(smoothed_data_by_boundaries)
(10.9,32.3]1 (10.9,32.3]2 (10.9,32.3]1 (10.9,32.3]2 (10.9,32.3]1
11          30          11          30          11
(10.9,32.3]2 (10.9,32.3]1 (10.9,32.3]2 (10.9,32.3]1 (10.9,32.3]2
30          11          30          11          30
(10.9,32.3]1 (10.9,32.3]2 (10.9,32.3]1 (10.9,32.3]2 (10.9,32.3]1
11          30          11          30          11
(10.9,32.3]2 (10.9,32.3]1 (10.9,32.3]2 (10.9,32.3]1 (10.9,32.3]2
30          11          30          11          30
```

Equal-Frequency (Equi-Depth) Partitioning



Q10:



Q11:

R version 4.4.1 (2024-06-14 ucrt) -- "Race for Your Life"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
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.

```
> quantile(age,.75)
Error: object 'age' not found
> #Q1, Q2
> age<-c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,25,30,33,33,35,35,35,35,36,40,45,4
6,52,70)
> quantile(age,.25)
25%
20.5
> quantile(age,.75)
75%
35
> |
```

Values

age	num [1:27]
13	13
15	15
16	16
19	19
20	20
20	20
21	21
22	22
22	22
25	25
25	25
25	25
25	25
30	30
33	33
33	33
35	35
35	35
35	35
36	36
40	40
45	45
46	46
52	52
70	70

Files Plots Packages Help Viewer Presentation

Name	Size	Modified
.Rhistory	76 B	Jul 24, 2024, 8:35 AM
201[1].pdf	573.8 KB	Apr 30, 2024, 4:51 PM
206[1].pptx	1.8 MB	Apr 14, 2024, 5:58 PM
206[1].pdf	676.5 KB	Apr 30, 2024, 4:47 PM
206[1].pptx	805.5 KB	Apr 30, 2024, 4:46 PM
192211201[1].pdf	599.9 KB	May 6, 2024, 10:33 PM
192211201[1].pptx	592 KB	May 6, 2024, 10:32 PM
192211206 - 1.docx	116.9 KB	Mar 17, 2024, 12:52 AM
192211206 - 2.docx	140.9 KB	Mar 7, 2024, 7:00 PM
192211206 - B. Hemanth Chowdary - 1.pptx	270 KB	Apr 16, 2024, 10:41 AM
192211206 - B. Hemanth Chowdary - 3 (1).pptx	397.8 KB	Apr 16, 2024, 1:41 PM
192211206 - B. Hemanth Chowdary - 3.pptx	273.9 KB	Apr 16, 2024, 10:47 AM

09:13 24-07-2024 ENG IN

DAY-2

Q1:

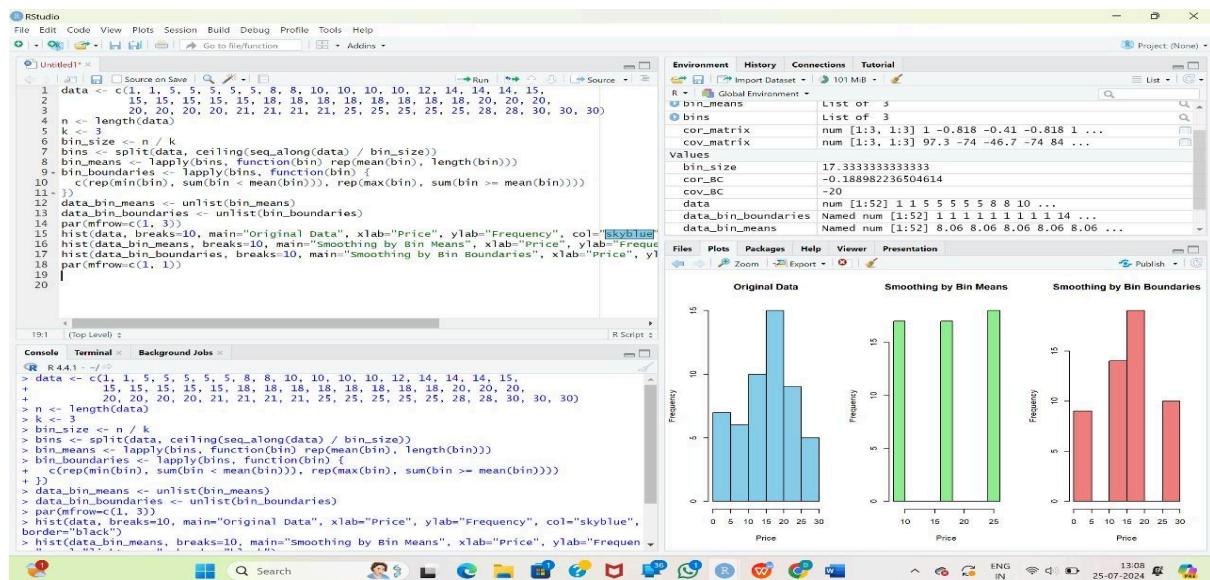
The screenshot shows an RStudio interface with the following details:

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

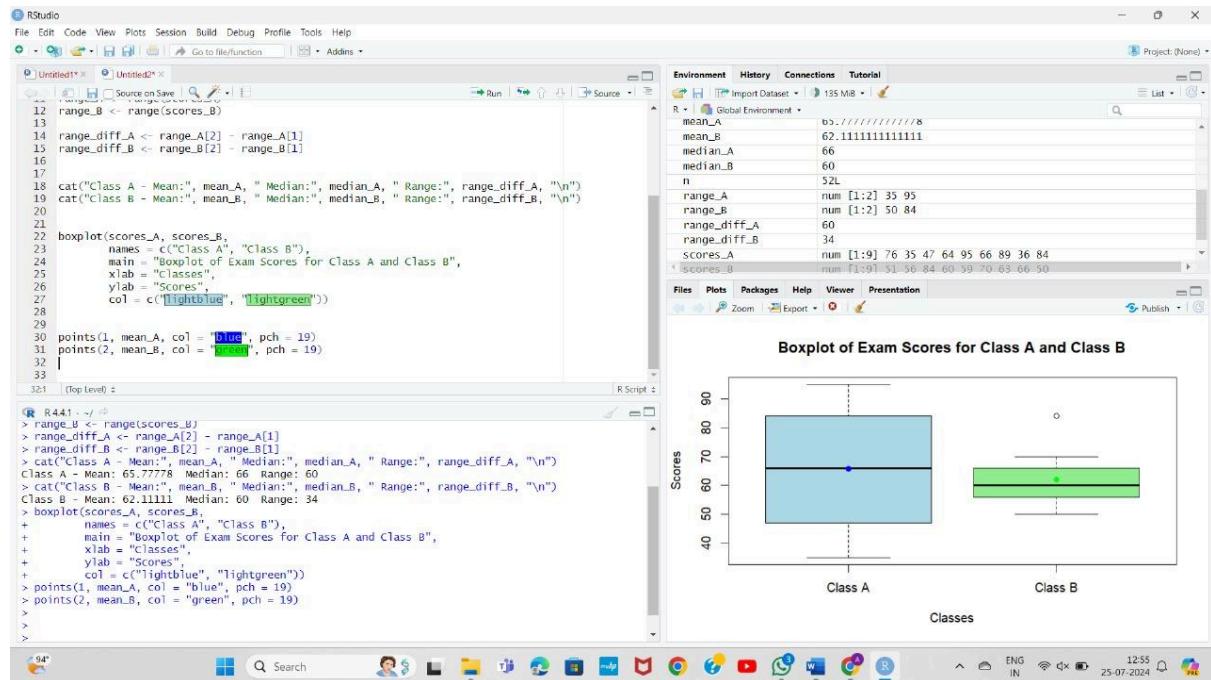
```
1 data <- data.frame(
2   A = c(18, 20),
3   B = c(22, 28, 10),
4   C = c(20, 40, 40)
5 )
6 cov_BC <- cov(data$B, data$C)
7 cov_matrix <- cov(data)
8 cor_BC <- cor(data$B, data$C)
9 cor_matrix <- cor(data)
10 print(paste("Sample Covariance between B and C:", cov_BC))
11 print("Sample Covariance Matrix:")
12 print(cov_matrix)
13 print(paste("Sample Correlation between B and C:", cor_BC))
14 print("Sample Correlation Matrix:")
15 print(cor_matrix)
16
17
```
- Environment View:** Shows variables and their values:

Variable	Value
cor_matrix	num [1:3, 1:3] 1 -0.818 -0.41 -0.818 1 ...
cov_matrix	num [1:3, 1:3] 97.3 -74 -46.7 -74 84 ...
data	3 obs. of 3 variables
cor_BC	-0.188982236504614
cov_BC	-20
- File Explorer:** Shows a list of files in the current directory, including various documents and Python scripts.
- Console:** Displays the output of the R code, including the covariance matrix and correlation coefficient.

Q2:



Q3:



The screenshot shows an RStudio interface with a script editor containing R code for generating a boxplot. The code calculates ranges and differences for two classes, then creates a boxplot with specific styling. The boxplot is titled "Boxplot of Exam Scores for Class A and Class B".

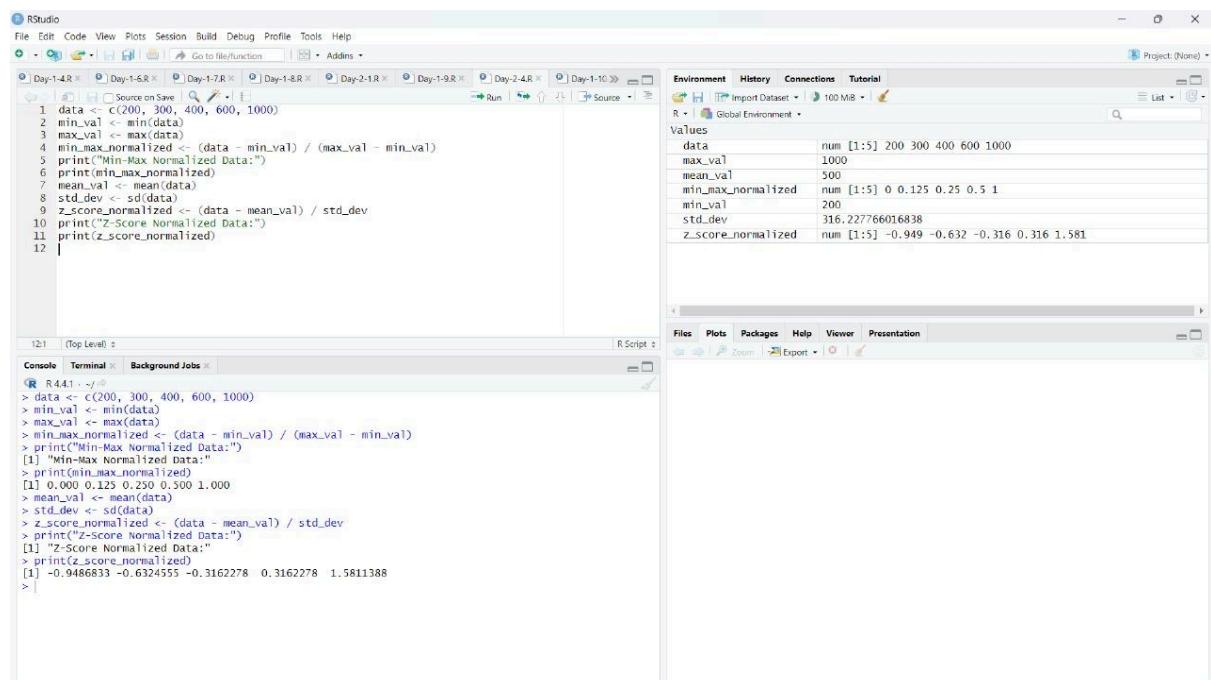
```
R 4.4.1 -->
range_B <- range(scores_B)
range_diff_A <- range_A[2] - range_A[1]
range_diff_B <- range_B[2] - range_B[1]
range_A
range_B
range_diff_A
range_diff_B
cat("Class A - Mean:", mean_A, " Median:", median_A, " Range:", range_A, "\n")
cat("Class B - Mean:", mean_B, " Median:", median_B, " Range:", range_B, "\n")
boxplot(scores_A, scores_B,
names = c("Class A", "Class B"),
main = "Boxplot of Exam Scores for Class A and Class B",
xlab = "Classes",
ylab = "Scores",
col = c("lightblue", "lightgreen"))

points(1, mean_A, col = "blue", pch = 19)
points(2, mean_B, col = "green", pch = 19)

```

The boxplot displays two distributions. Class A (blue) has a median around 65, an IQR from approximately 45 to 70, and whiskers extending from about 35 to 95. Class B (green) has a median around 60, an IQR from approximately 50 to 65, and whiskers extending from about 40 to 80. An outlier is visible for Class B at a score of approximately 90.

Q4:



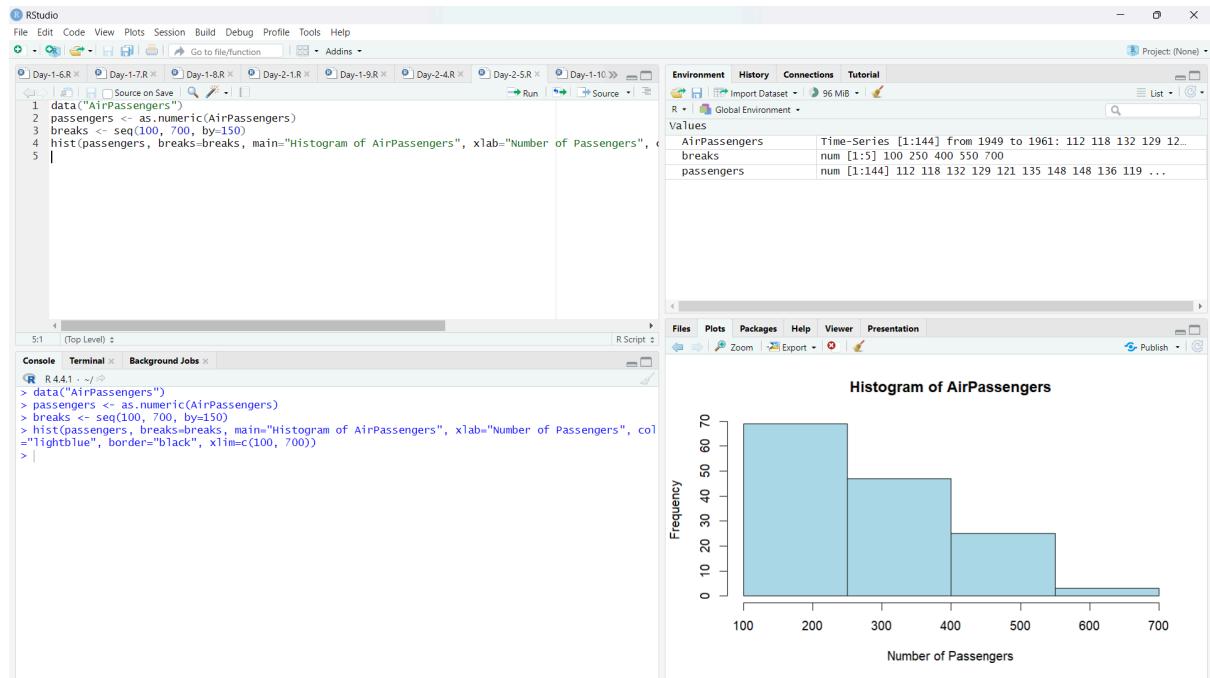
The screenshot shows an RStudio interface with a script editor containing R code for data normalization. The code calculates min, max, and z-scores for a dataset of values [200, 300, 400, 600, 1000].

```
R 4.4.1 -->
data <- c(200, 300, 400, 600, 1000)
min_val <- min(data)
max_val <- max(data)
min_max_normalized <- (data - min_val) / (max_val - min_val)
print("Min-Max Normalized Data:")
[1] "Min-Max Normalized Data:"
[1] 0.000 0.125 0.250 0.500 1.000
mean_val <- mean(data)
std_dev <- sd(data)
z_score_normalized <- (data - mean_val) / std_dev
print("Z-Score Normalized Data:")
[1] "Z-Score Normalized Data:"
[1] -0.9486833 -0.6334555 -0.3162278 0.3162278 1.5811388

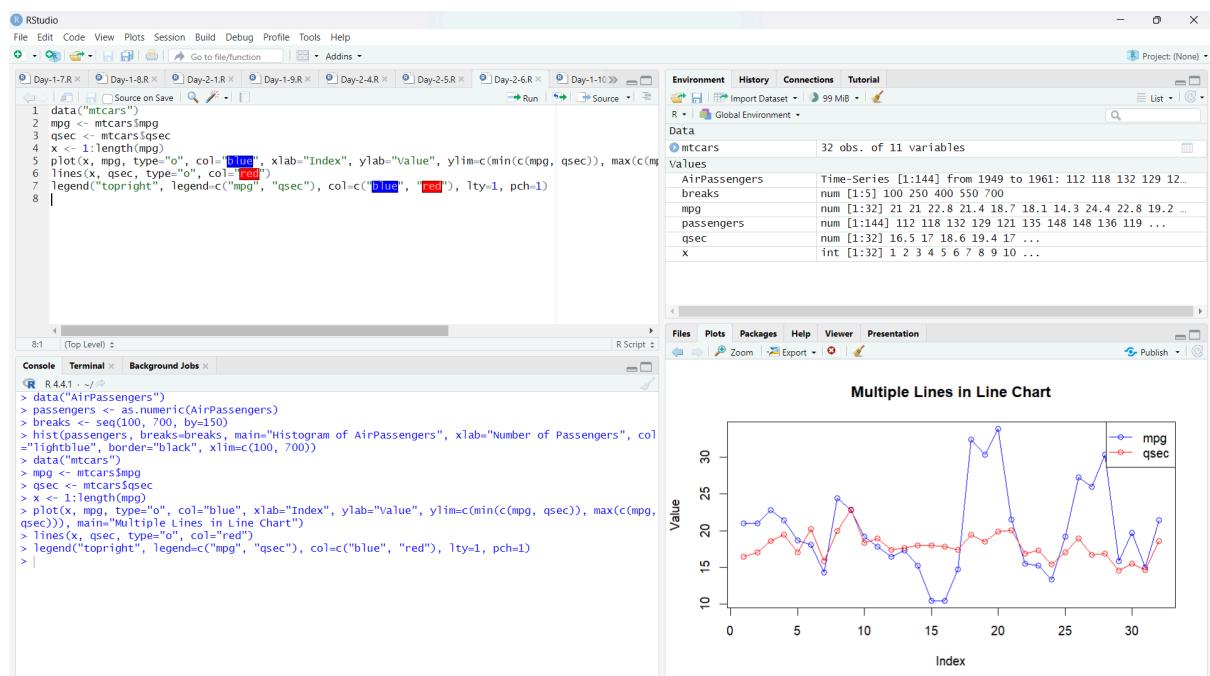
```

The output shows the original data, the calculated min and max values (200 and 1000), the min-max normalized data (ranging from 0.000 to 1.000), the mean and standard deviation (mean ~500, std dev ~316), and the z-score normalized data (ranging from -0.948 to 1.581).

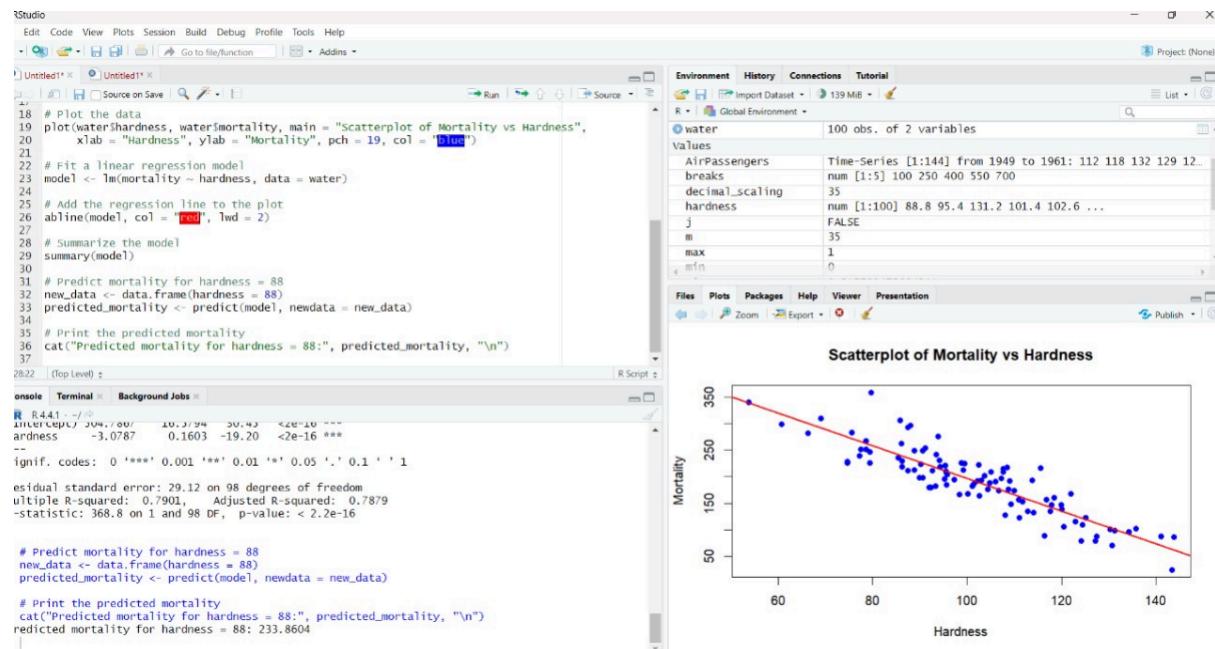
Q5:



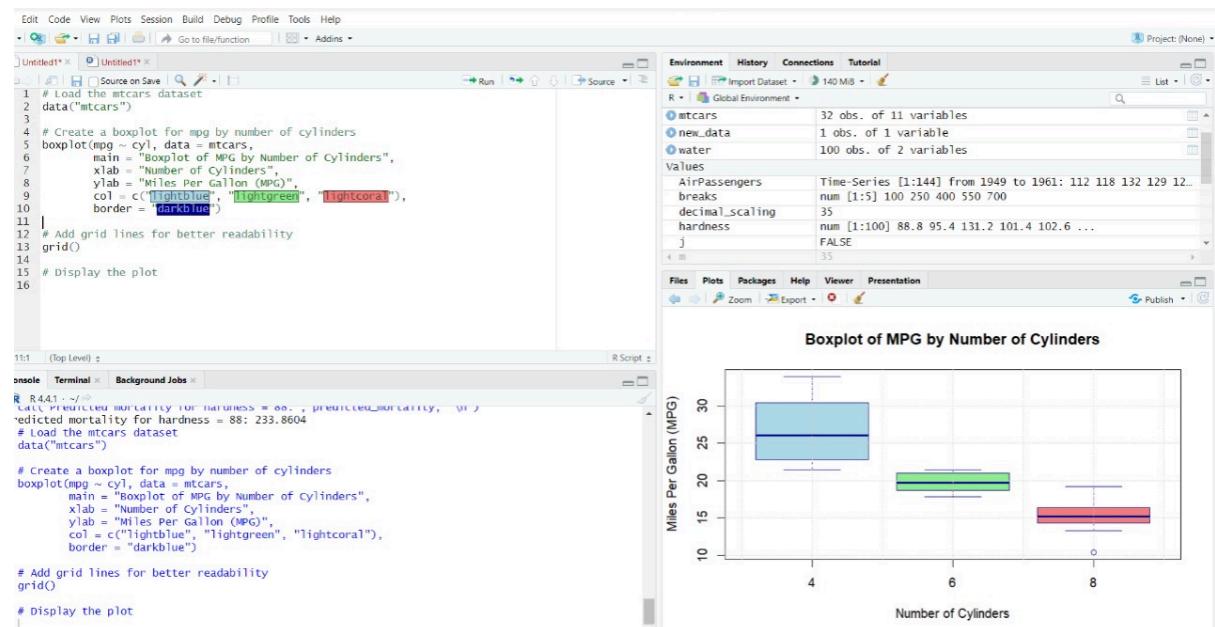
Q6:



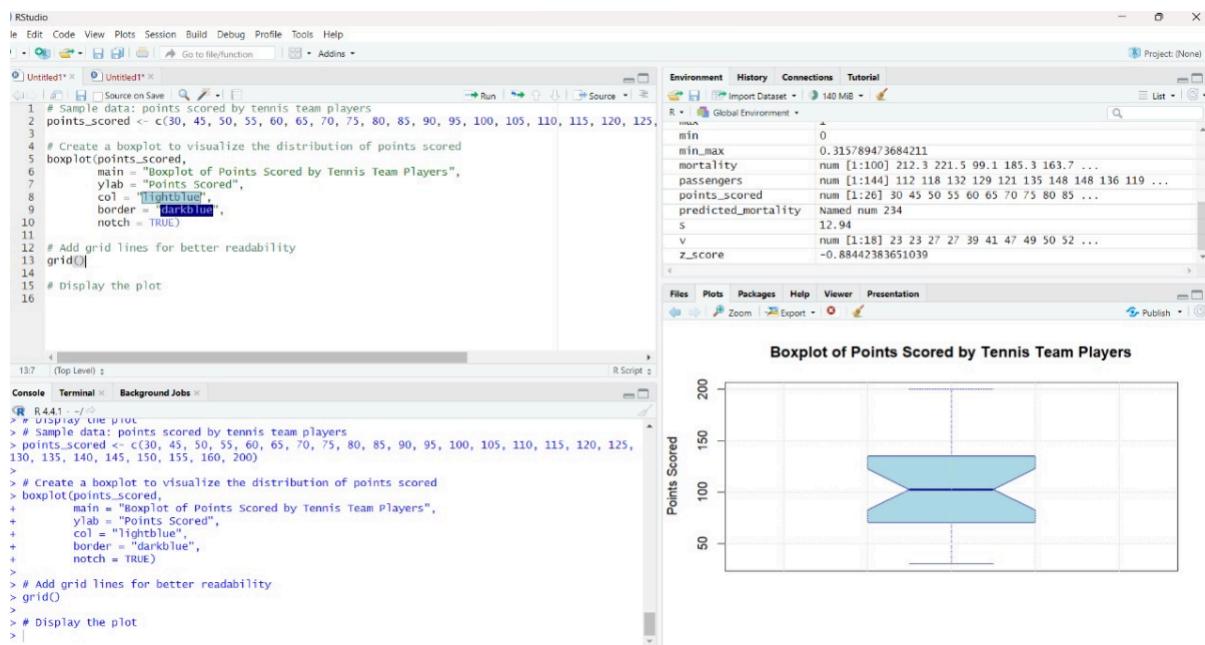
Q7:



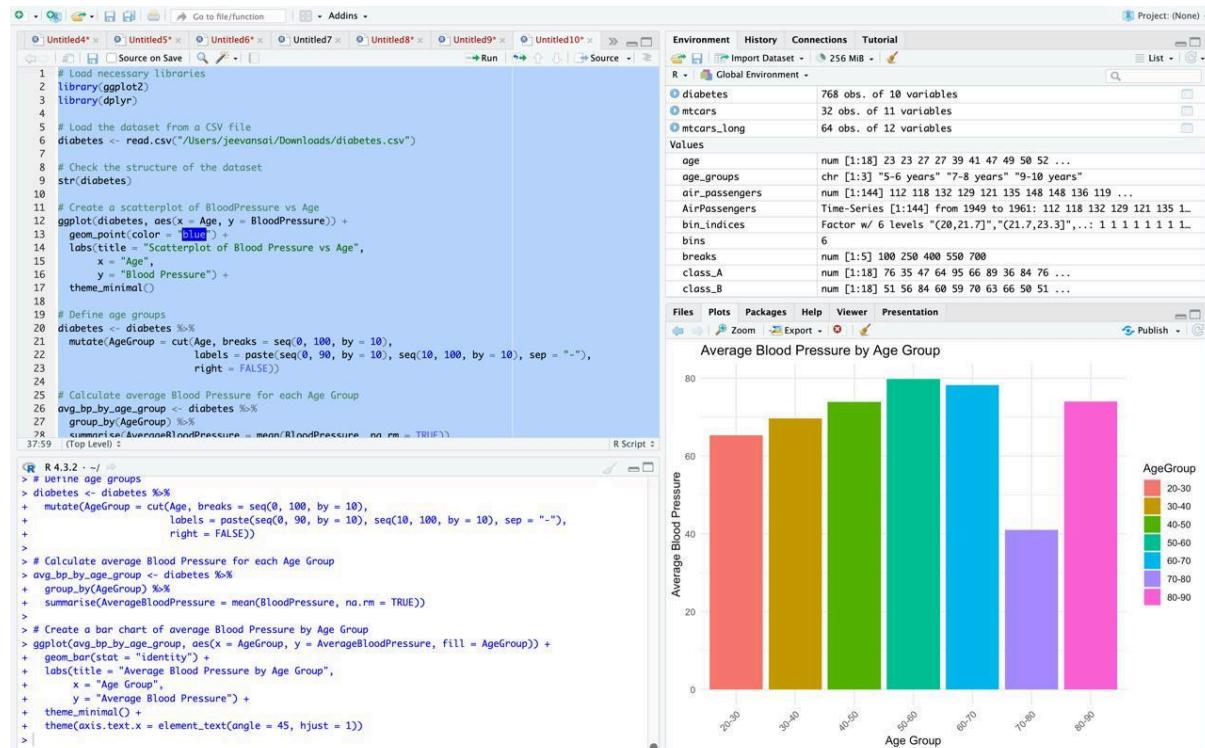
Q8:



Q9:

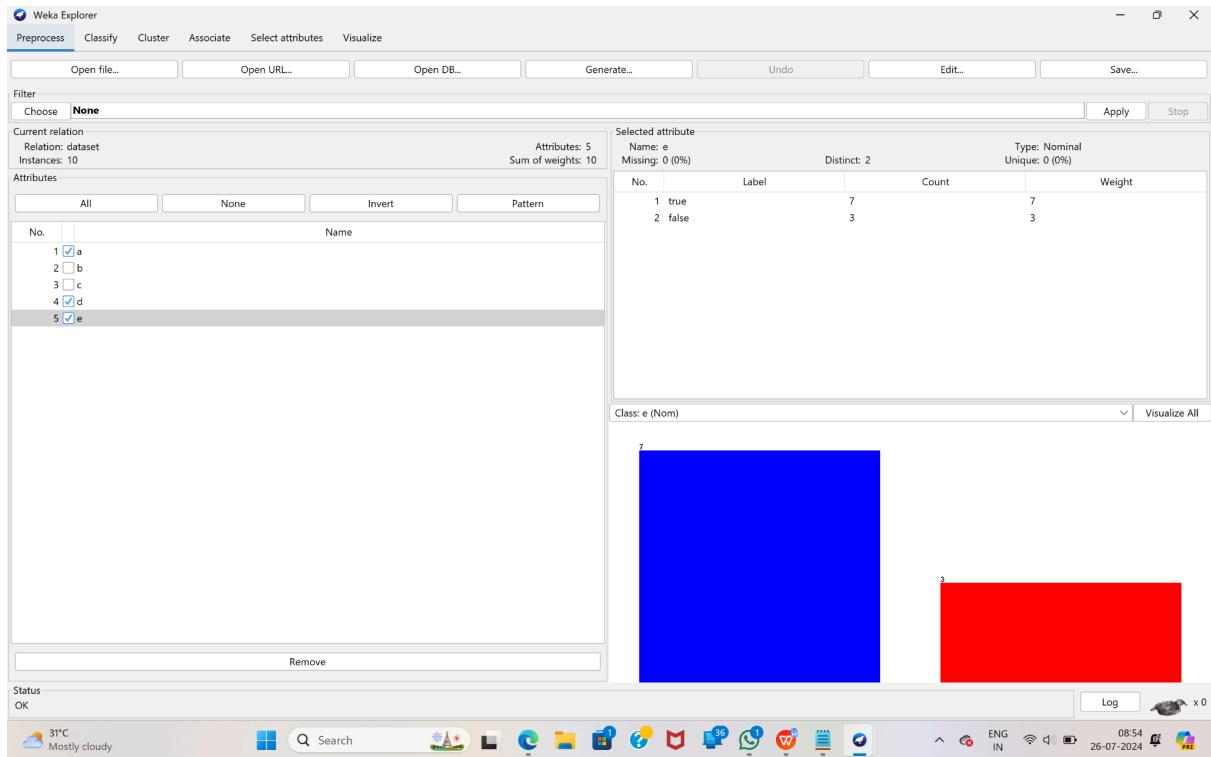


Q10:

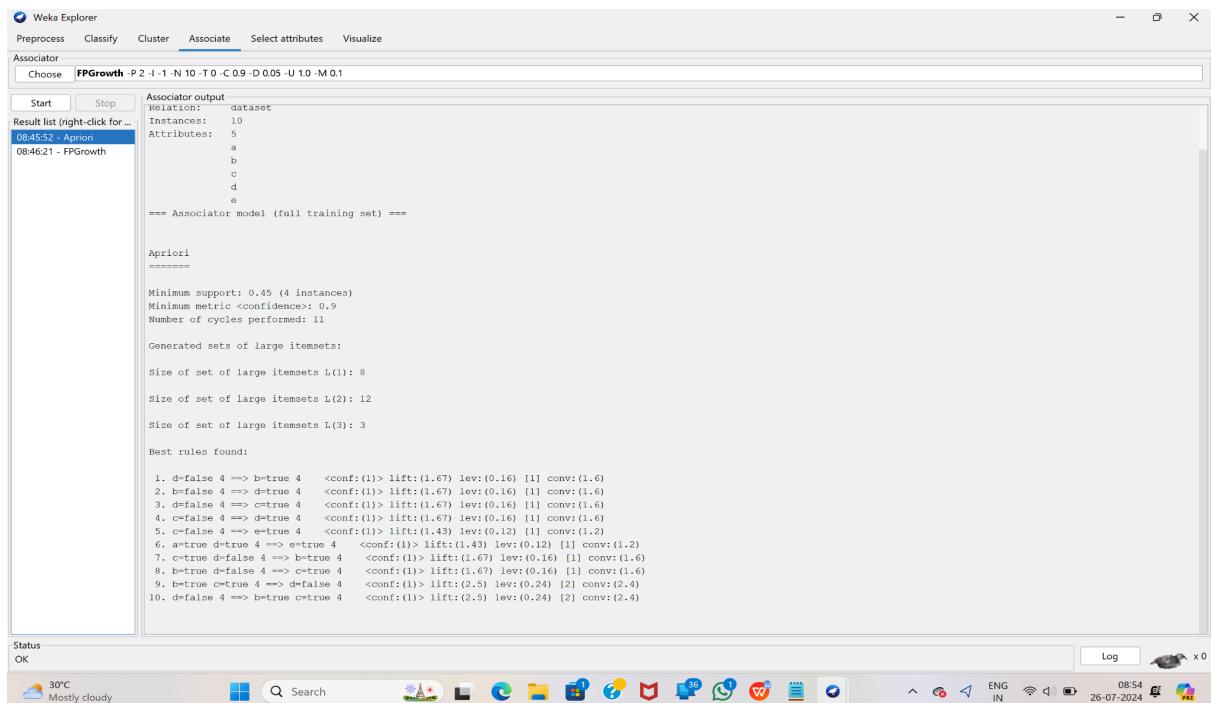


DAY-3

Q1:



Apriori Algorithm:



FP GROWTH:

Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associator

Choose **FPGrowth -P 2 -I 1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1**

Start Stop

Result list (right-click for ...)

08:45:52 - Apriori
08:46:21 - FPGrowth

Associator output

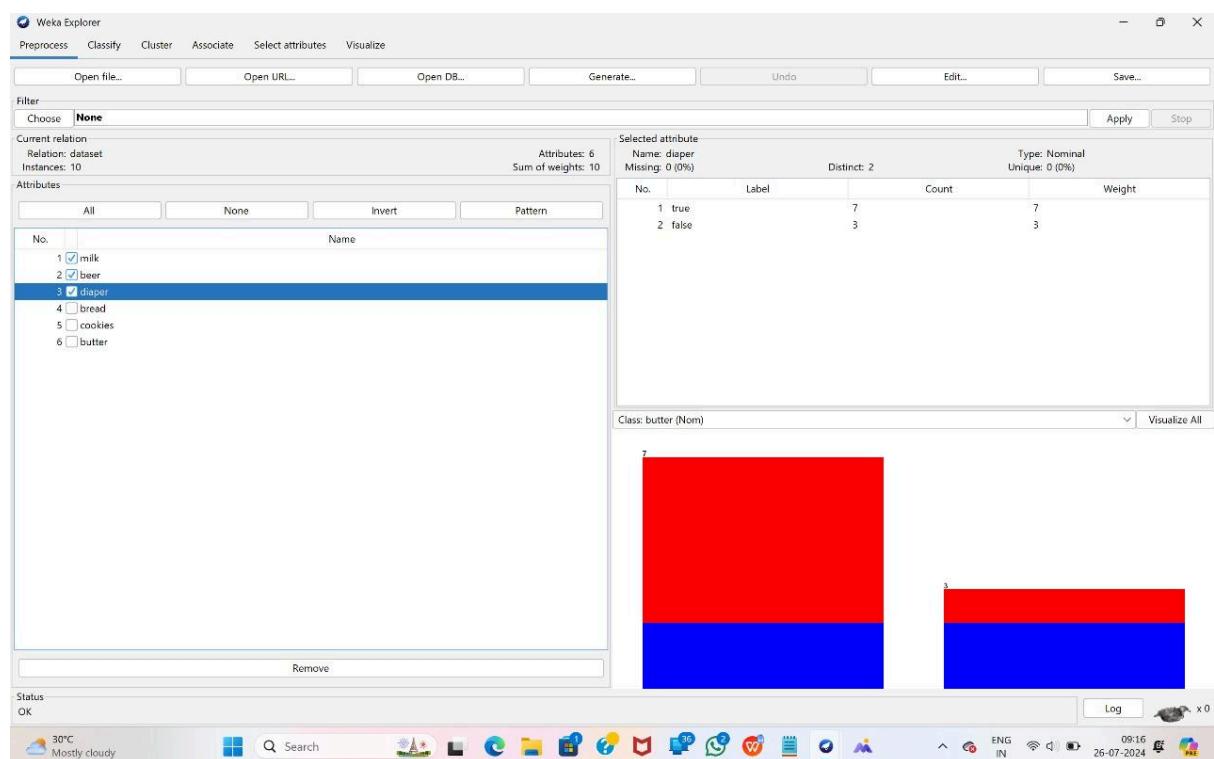
```
==== Run information ====
Scheme: weka.associations.FPGrowth -P 2 -I 1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1
Relation: dataset
Instances: 10
Attributes: 5
a
b
c
d
e
==== Associator model (full training set) ====
FPGrowth found 3 rules (displaying top 3)

1. {b=false, e=false}: l ==> {a=false}: 1 <conf:(1)> lift:(3.33) lev:(0.07) conv:(0.7)
2. {b=false, a=false}: l ==> {e=false}: 1 <conf:(1)> lift:(3.33) lev:(0.07) conv:(0.7)
3. {e=false, a=false}: l ==> {b=false}: 1 <conf:(1)> lift:(2.5) lev:(0.06) conv:(0.6)
```

Status OK Log x 0

30°C Mostly cloudy Search ENG IN 08:54 26-07-2024

Q2:



Apriori Algorithm:

The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Choose" dropdown is set to "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" for a dataset with 10 instances and 6 attributes: milk, beer, diaper, bread, cookies, butter. The output shows the generated sets of large itemsets and the best rules found, such as "bread=true 5 => beer=false 5" and "cookies=true 5 => bread=true 5".

```

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
Result list (right-click for ...)

09:19:22 - Apriori
Associator output
Instances: 10
Attributes: 6
milk
beer
diaper
bread
cookies
butter
==== Associator model (full training set) ====

Apriori
-----
Minimum support: 0.55 (5 instances)
Minimum metric <confidenceno>: 0.9
Number of cycles performed: 9

Generated sets of large itemsets:
Size of set of large itemsets L(1): 9
Size of set of large itemsets L(2): 5
Size of set of large itemsets L(3): 1

Best rules found:
1. bread=true 5 => beer=false 5   <conf:(1)> lift:(1.67) lev:(0.2) [2] conv:(2)
2. cookies=true 5 => beer=false 5   <conf:(1)> lift:(1.67) lev:(0.2) [2] conv:(2)
3. cookies=true 5 => bread=true 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)
4. bread=true 5 => cookies=true 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)
5. cookies=false 5 => bread=false 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)
6. bread=false 5 => cookies=false 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)
7. bread=false cookies=false 5 => beer=false 5   <conf:(1)> lift:(1.67) lev:(0.2) [2] conv:(2)
8. beer=false cookie=false 5 => bread=true 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)
9. beer=false bread=true 5 => cookies=true 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)
10. cookies=true 5 => beer=false bread=true 5   <conf:(1)> lift:(2) lev:(0.25) [2] conv:(2.5)

Status
OK
Log x0

```

FP GROWTH:

The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Choose" dropdown is set to "FPGrowth - P 2 - I 1 - N 10 - T 0 - C 0.9 - D 0.05 - U 1.0 - M 0.1". The main window displays the "Associator output" for a dataset with 10 instances and 6 attributes: milk, beer, diaper, bread, cookies, butter. The output shows the rules found by FP-Growth, such as "[cookies=false]; 5 => [bread=false]; 5" and "[beer=false, bread=false]; 1 => [cookies=false]; 1".

```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Associate
Choose FPGrowth - P 2 - I 1 - N 10 - T 0 - C 0.9 - D 0.05 - U 1.0 - M 0.1

Start Stop
Result list (right-click for ...)

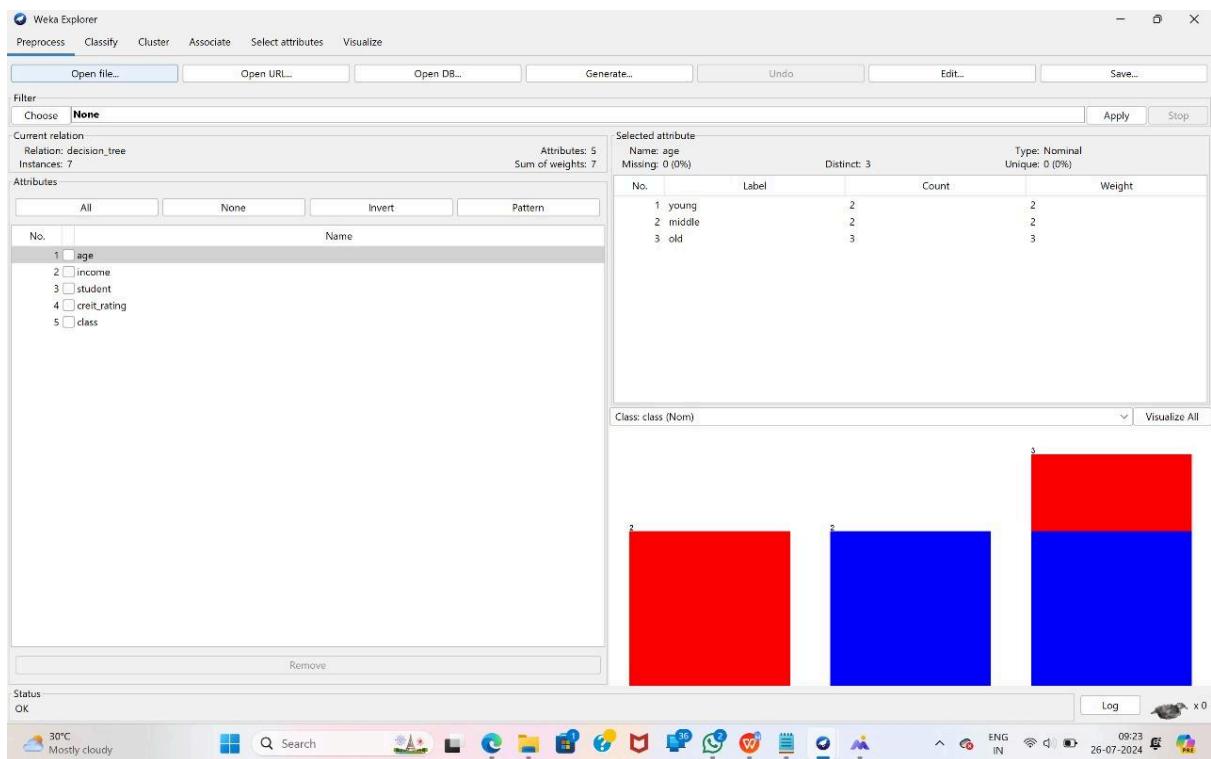
09:19:22 - Apriori
09:20:09 - FPGrowth
Associator output
==== Run information ====
Scheme: weka.associations.FPGrowth P 2 I 1 N 10 T 0 C 0.9 D 0.05 U 1.0 M 0.1
Relation: dataset
Instances: 10
Attributes: 6
milk
beer
diaper
bread
cookies
butter
==== Associator model (full training set) ====
FPGrowth found 20 rules (displaying top 10)

1. [cookies=false]; 5 => [bread=false]; 5   <conf:(1)> lift:(2) lev:(0.25) conv:(2.5)
2. [bread=false]; 5 => [cookies=false]; 5   <conf:(1)> lift:(2) lev:(0.25) conv:(2.5)
3. [beer=false, diaper=false]; 1 => [beer=false]; 1   <conf:(1)> lift:(1.67) lev:(0.04) conv:(0.4)
4. [beer=false, cookie=false]; 2 => [bread=false]; 2   <conf:(1)> lift:(2) lev:(0.1) conv:(1)
5. [bread=false, cookie=false]; 2 => [cookies=false]; 2   <conf:(1)> lift:(2) lev:(0.1) conv:(1)
6. [beer=false, cookies=false]; 1 => [bread=false]; 1   <conf:(1)> lift:(2) lev:(0.05) conv:(0.5)
7. [beer=false, bread=false]; 1 => [cookies=false]; 1   <conf:(1)> lift:(2) lev:(0.05) conv:(0.5)
8. [milk=false, cookies=false]; 3 => [bread=false]; 3   <conf:(1)> lift:(2) lev:(0.15) conv:(1.5)
9. [milk=false, bread=false]; 3 => [cookies=false]; 3   <conf:(1)> lift:(2) lev:(0.15) conv:(1.5)
10. [cookies=false, diaper=false]; 1 => [milk=false]; 1   <conf:(1)> lift:(2) lev:(0.05) conv:(0.5)

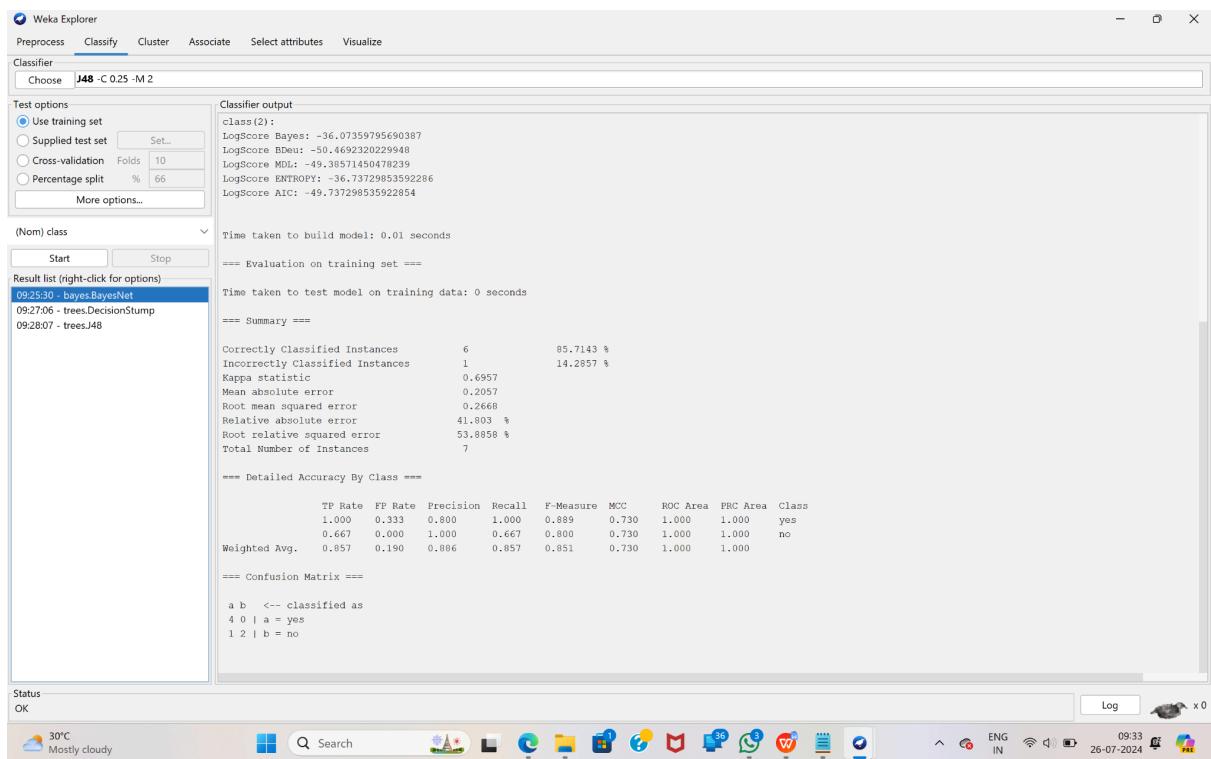
Status
OK
Log x0

```

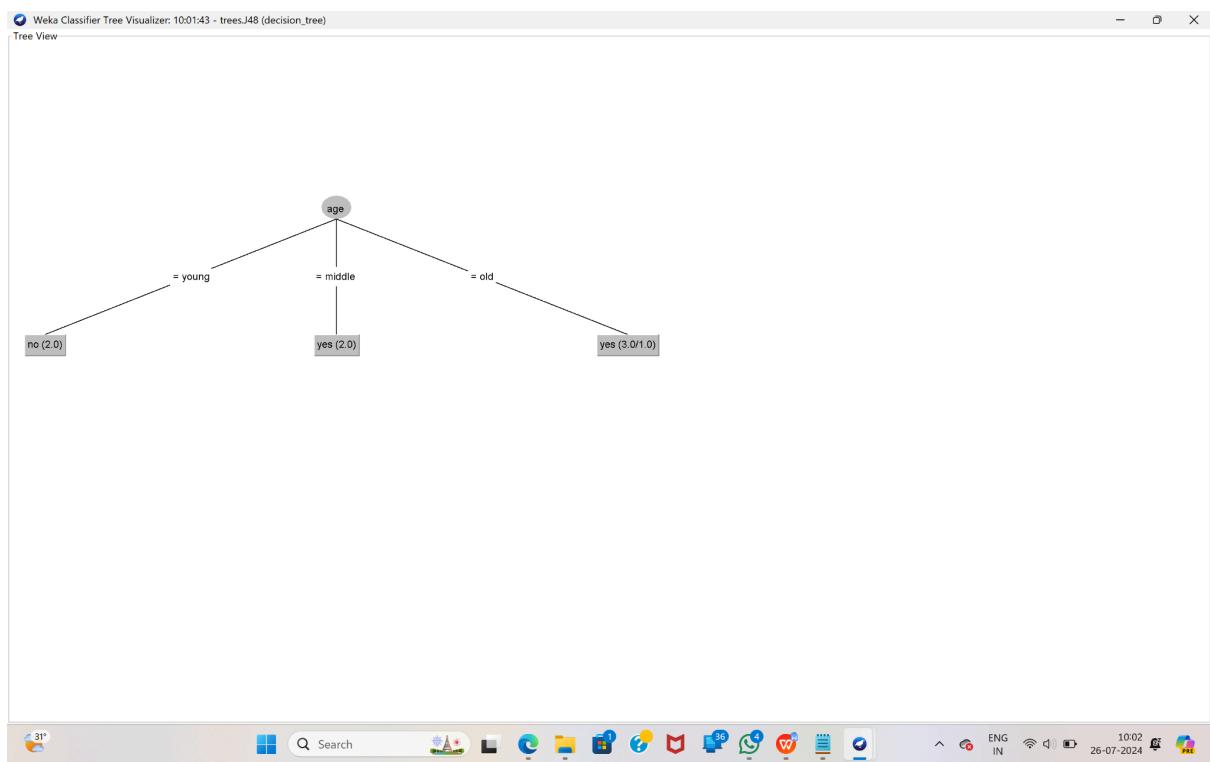
Q3:



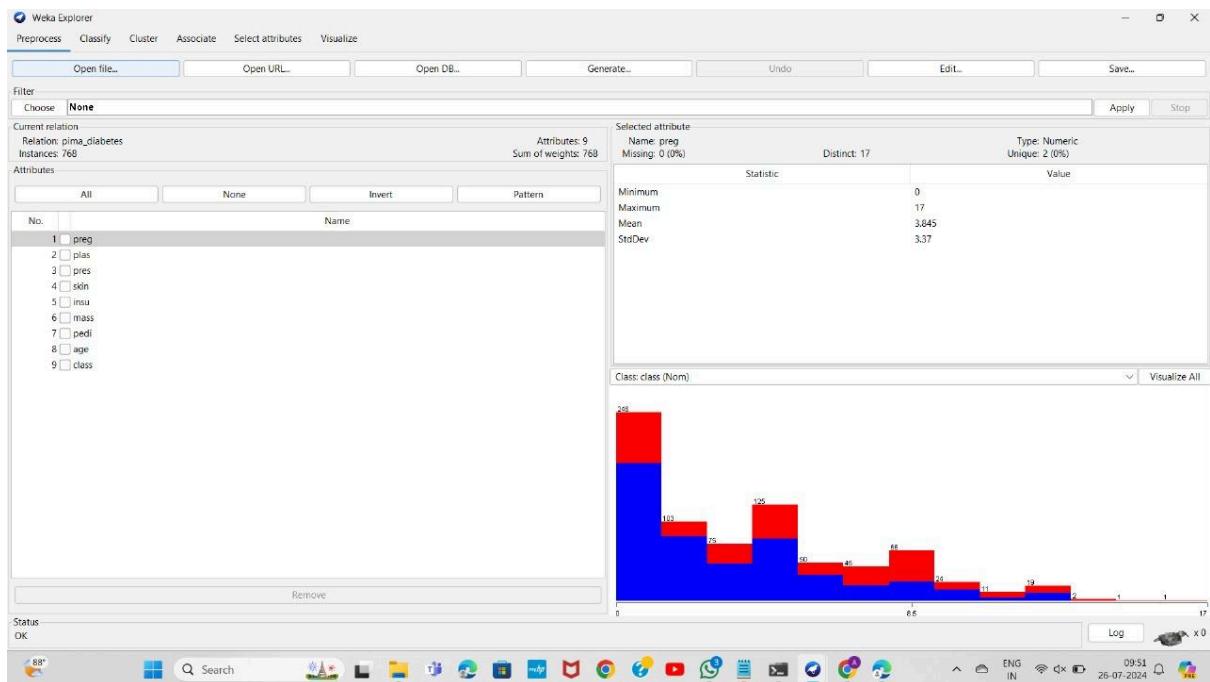
Bayes classification:



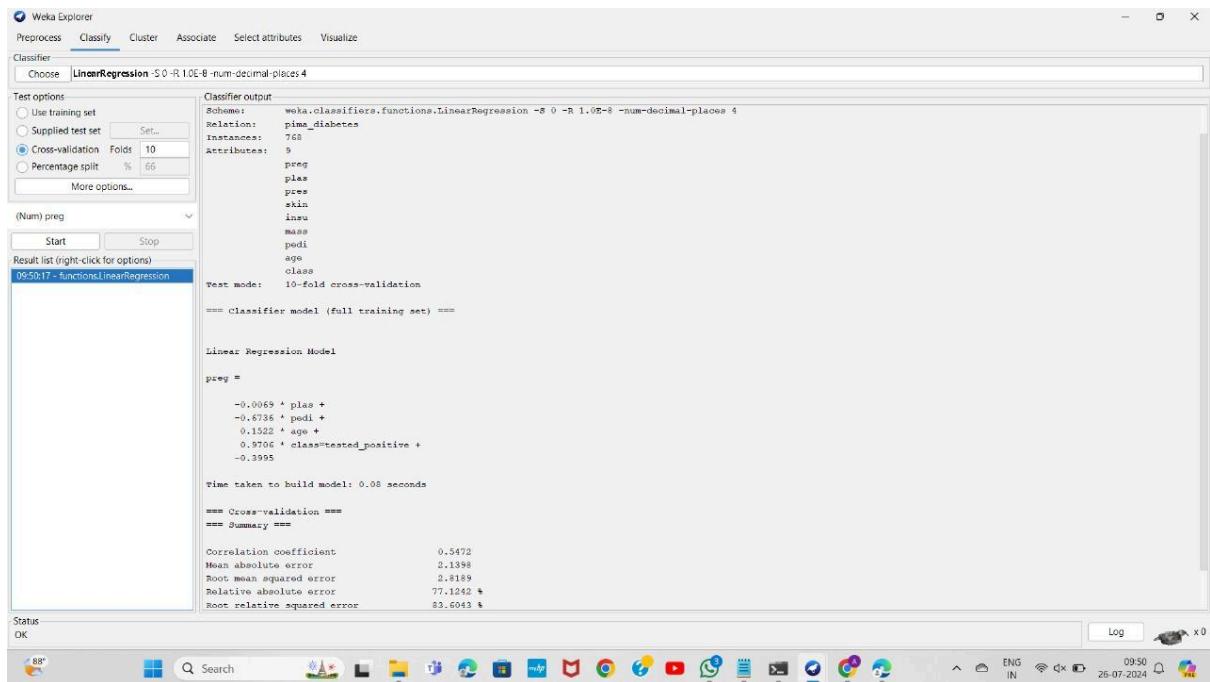
Decision tree:



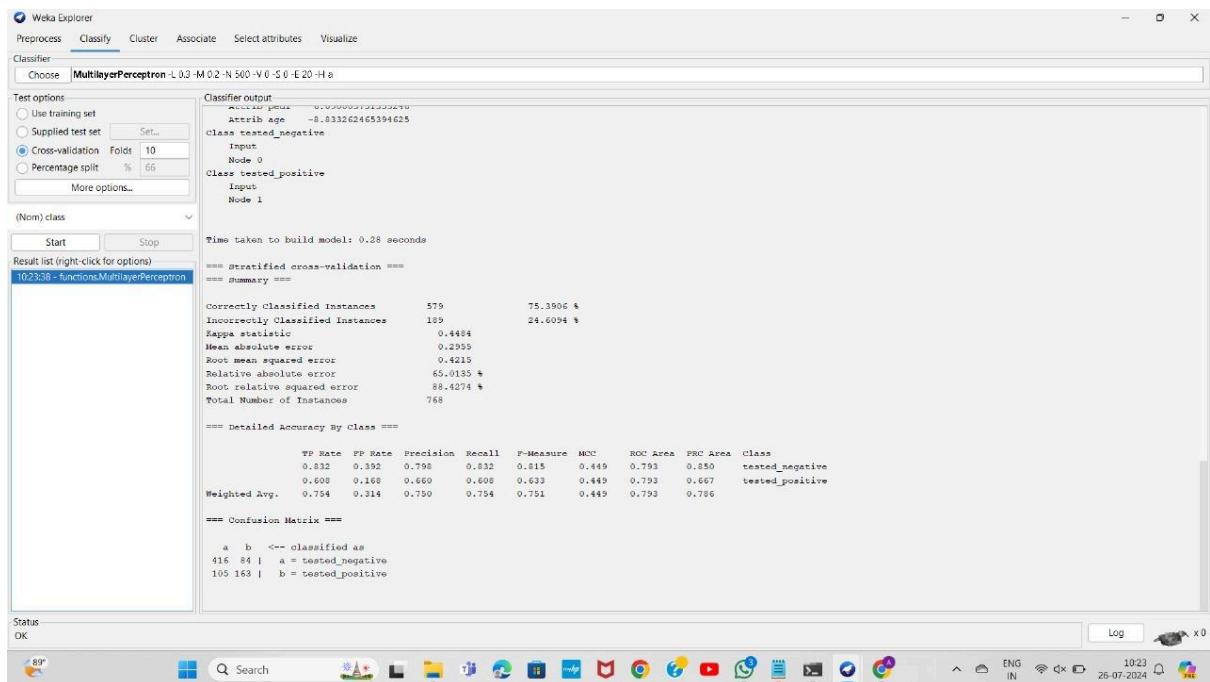
Q4:



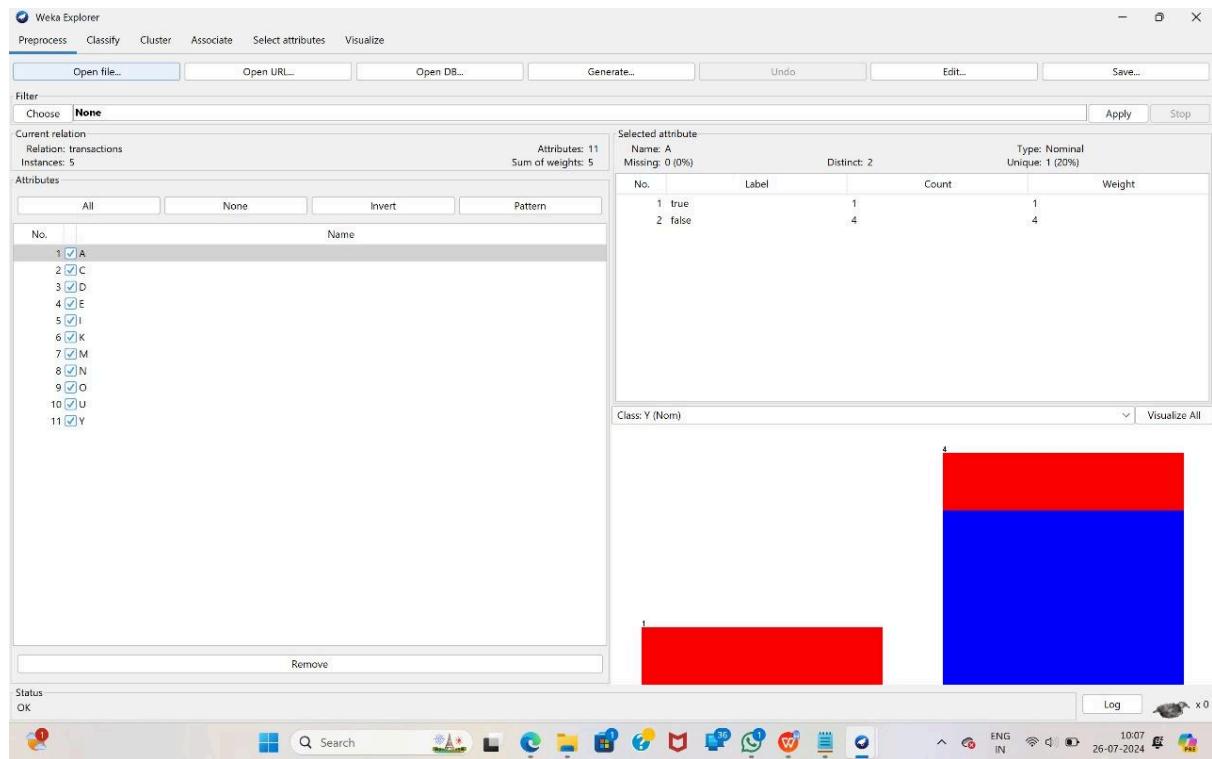
Linear regression:



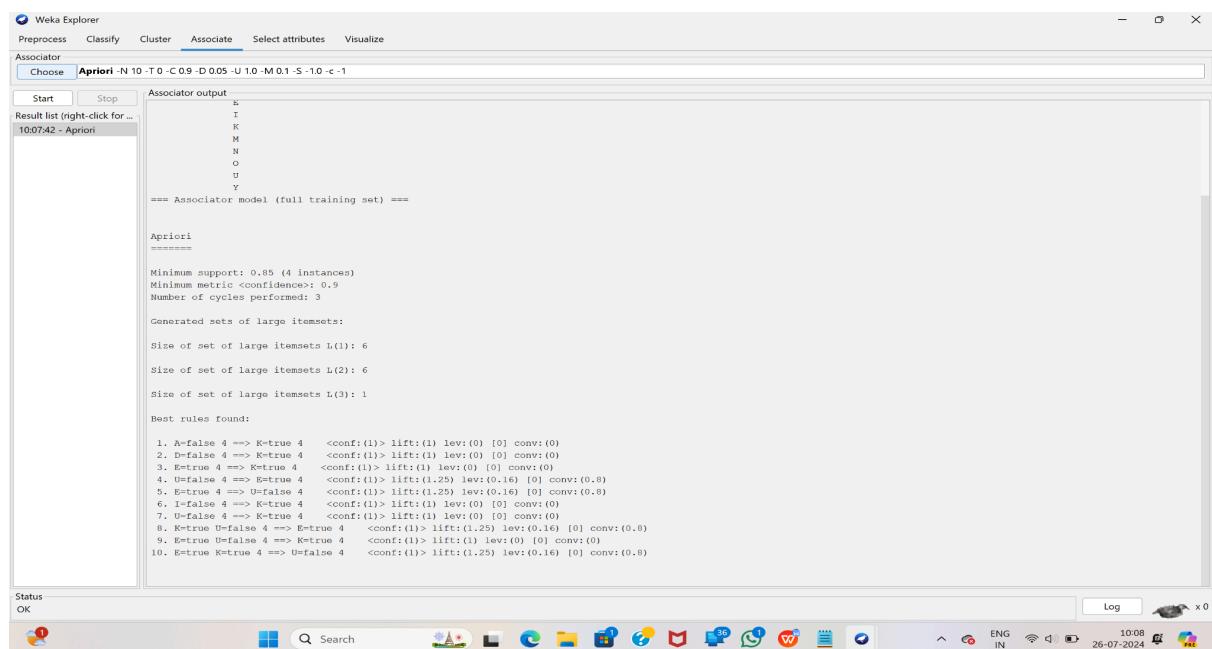
Multiple regression:



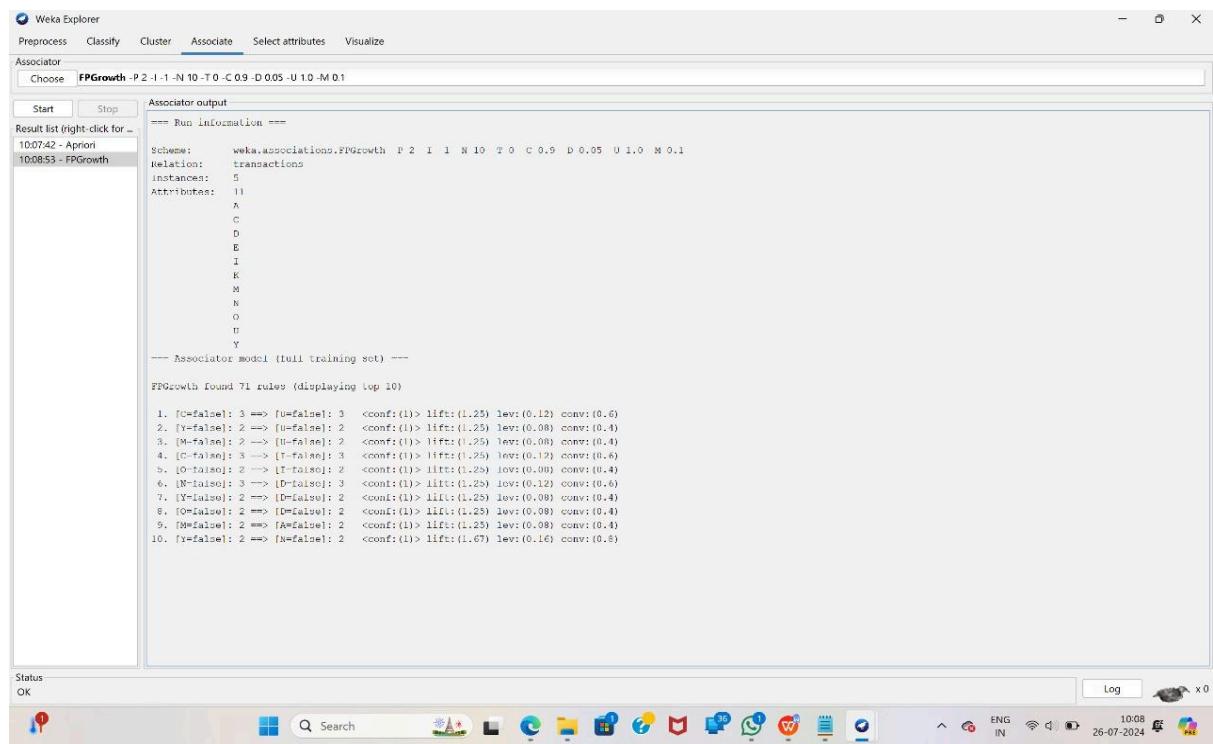
Q5:



Apriori Algorithm:



FP GROWTH:

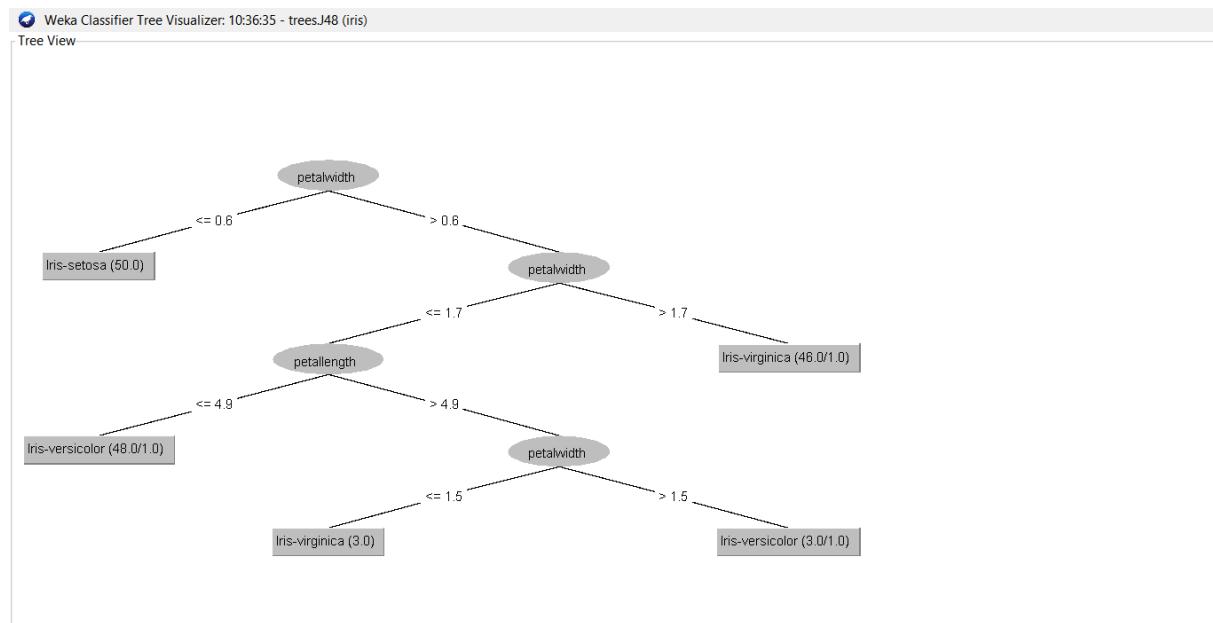


The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Choose" dropdown is set to "FPGrowth - P 2 - I 1 - N 10 - T 0 - C 0.9 - D 0.05 - U 1.0 - M 0.1". The "Result list (right-click for..." section displays the following information:

```
10:07:42 - Aprori  
10:08:53 - FPGrowth  
Start Stop  
Associator output  
==== Run information ====  
Scheme: weka.associations.FPGrowth P 2 I 1 N 10 T 0 C 0.9 D 0.05 U 1.0 M 0.1  
Relation: transactions  
instances: 5  
Attributes: 11  
A  
B  
C  
D  
E  
I  
K  
M  
N  
O  
U  
Y  
==== Associator model (built training set) ====  
FPGrowth found 71 rules (displaying top 10)  
1. [C=false]: 3 ==> [U=false]: 3 <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)  
2. [Y=false]: 2 ==> [U=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)  
3. [M=false]: 2 ==> [U=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)  
4. [C=false]: 3 ==> [T=false]: 3 <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)  
5. [O=false]: 2 ==> [T=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)  
6. [M=false]: 3 ==> [D=false]: 3 <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)  
7. [Y=false]: 2 ==> [D=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)  
8. [O=false]: 2 ==> [P=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)  
9. [M=false]: 2 ==> [A=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)  
10. [Y=false]: 2 ==> [N=false]: 2 <conf:(1)> lift:(1.67) lev:(0.16) conv:(0.8)
```

Q6:

TREE:



PREPROCESS:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'J48 - C 0.25 - M 2'. The 'Test options' panel indicates 'Cross-validation' with 10 folds. The 'Classifier output' panel displays the following information:

- Size of the tree :** 9
- Time taken to build model:** 0 seconds
- Stratified cross-validation**
- Summary**
- Correctly Classified Instances**: 144 (96 %)
- Incorrectly Classified Instances**: 6 (4 %)
- Kappa statistic**: 0.94
- Mean absolute error**: 0.035
- Root mean squared error**: 0.1586
- Relative absolute error**: 7.8705 %
- Root relative squared error**: 33.6353 %
- Total Number of Instances**: 150
- Detailed Accuracy By Class** (Table):

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Iris-setosa	0.980	0.000	1.000	0.980	0.990	0.985	0.990	0.987	Iris-setosa
Iris-versicolor	0.940	0.030	0.940	0.940	0.940	0.910	0.952	0.880	Iris-versicolor
Iris-virginica	0.960	0.030	0.941	0.960	0.950	0.925	0.961	0.905	Iris-virginica
Weighted Avg.	0.960	0.020	0.960	0.960	0.960	0.940	0.968	0.924	
- Confusion Matrix** (Table):

	a	b	c	--> classified as
a	49	1	0	a = Iris-setosa
b	0	47	3	b = Iris-versicolor
c	0	2	48	c = Iris-virginica

Logistic:

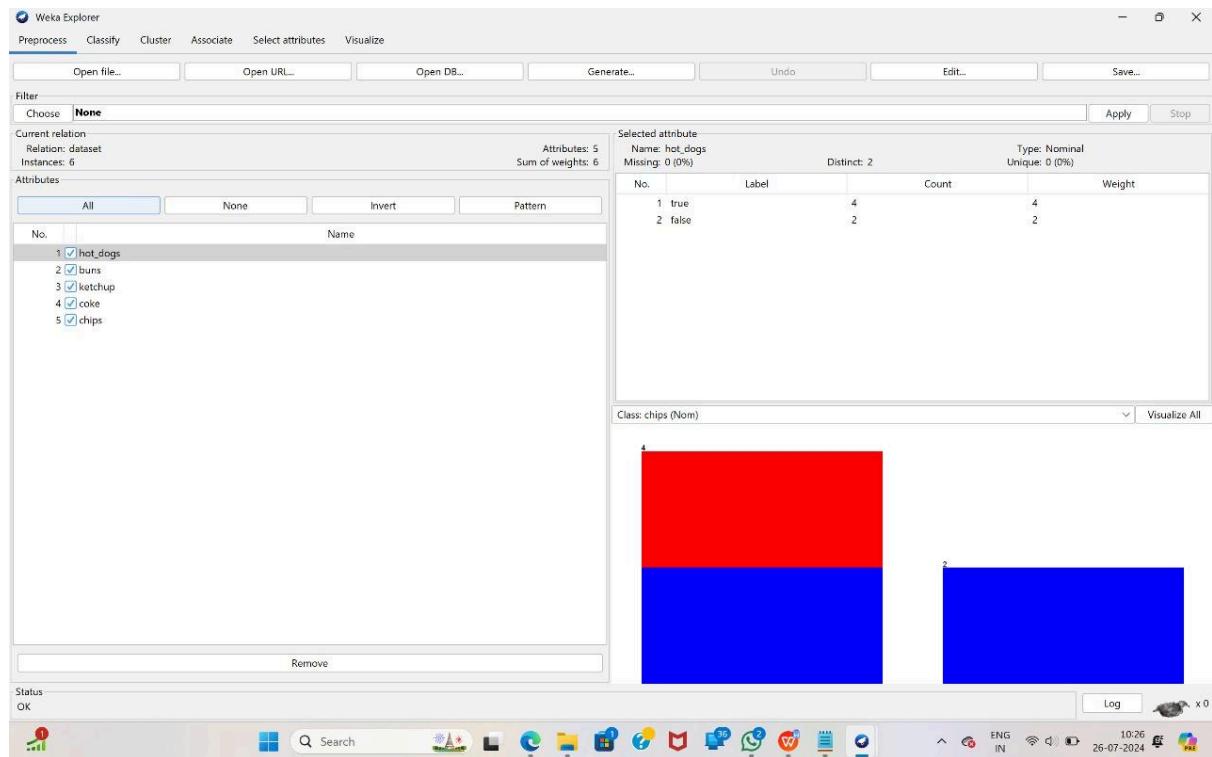
The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'Logistic - R 1.0E-8 - M 1 - num-decimal-places 4'. The 'Test options' panel indicates 'Cross-validation' with 10 folds. The 'Classifier output' panel displays the following information:

- petalwidth**: 0 0
- Time taken to build model:** 0.03 seconds
- Stratified cross-validation**
- Summary**
- Correctly Classified Instances**: 144 (96 %)
- Incorrectly Classified Instances**: 6 (4 %)
- Kappa statistic**: 0.94
- Mean absolute error**: 0.0287
- Root mean squared error**: 0.1424
- Relative absolute error**: 6.456 %
- Root relative squared error**: 30.2139 %
- Total Number of Instances**: 150
- Detailed Accuracy By Class** (Table):

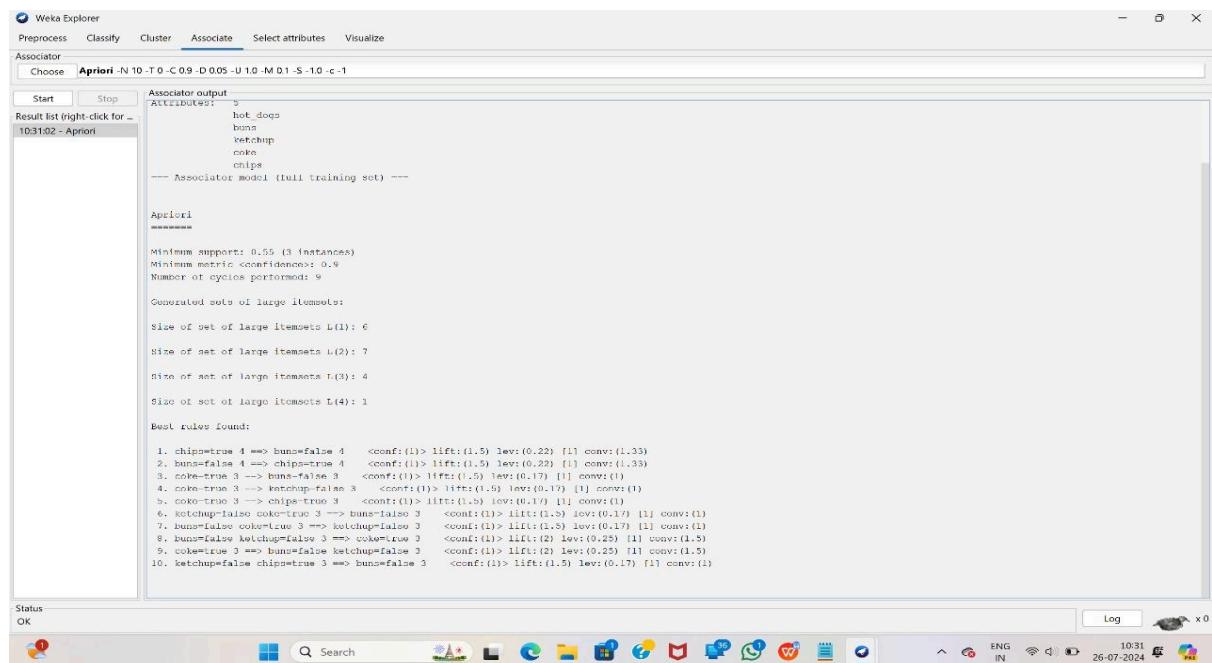
	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Iris-setosa	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	Iris-setosa
Iris-versicolor	0.920	0.020	0.958	0.920	0.939	0.910	0.972	0.934	Iris-versicolor
Iris-virginica	0.960	0.040	0.923	0.960	0.941	0.911	0.972	0.934	Iris-virginica
Weighted Avg.	0.960	0.020	0.960	0.960	0.960	0.940	0.981	0.956	
- Confusion Matrix** (Table):

	a	b	c	--> classified as
a	50	0	0	a = Iris-setosa
b	0	46	4	b = Iris-versicolor
c	0	2	48	c = Iris-virginica

Q7:



Apriori Algorithm:



FP GROWTH:

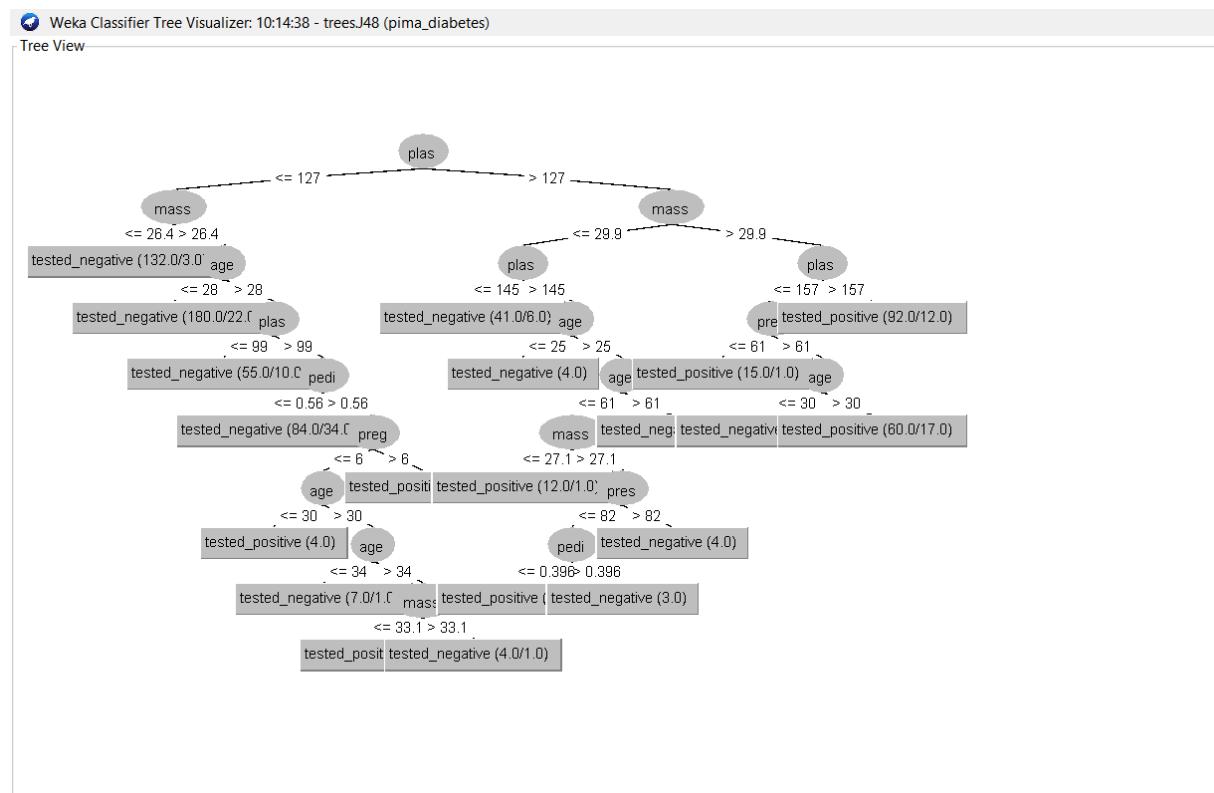
```

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Result list (right-click for ...)
1031:02 - Apriori
1031:33 - FPGrowth
Choose FPGrowth - P 2 - I 1 - N 10 - T 0 - C 0.9 - D 0.05 - U 1.0 - M 0.1
Associator
Start Stop
Result list (right-click for ...)
Run information ...
Scheme: weka.associations.FPGrowth P 2 I 1 N 10 T 0 C 0.9 D 0.05 U 1.0 M 0.1
Relation: dataset
instances: 6
Attributes: 5
hot_dogs
buns
ketchup
coke
chips
Associator model (full training set) ...
FPGrowth found 7 rules (displaying top 7)

1. [hot_dogs=false]: 2 -> [buns=false]: 2 <conf:(1)> lift:(1.5) lev:(0.11) conv:(0.67)
2. [chips=false]: 2 -> [coke=false]: 2 <conf:(1)> lift:(2) lev:(0.17) conv:(1)
3. [ketchup=false, hot_dogs=false]: 1 -> [buns=false]: 1 <conf:(1)> lift:(1.5) lev:(0.06) conv:(0.33)
4. [ketchup=false, coke=false]: 1 ==> [chips=false]: 1 <conf:(1)> lift:(3) lev:(0.11) conv:(0.67)
5. [ketchup=false, chips=false]: 1 ==> [coke=false]: 1 <conf:(1)> lift:(2) lev:(0.08) conv:(0.5)
6. [buns=false, coke=false]: 1 ==> [hot_dogs=false]: 1 <conf:(1)> lift:(3) lev:(0.11) conv:(0.67)
7. [coke=false, hot_dogs=false]: 1 ==> [buns=false]: 1 <conf:(1)> lift:(1.5) lev:(0.06) conv:(0.33)
    
```

Q8:

Tree:



Decision Tree:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. A classifier named 'DecisionTable-X 1-5 "weka.attributeSelection.BestFirst-D 1-N 5"' is chosen. The 'Test options' section shows 'Cross-validation' with 'Folds' set to 10. The 'Classifier output' pane displays the following text:

```

Classifier output
=====
Search: sev. NO attributes
Search direction: forward
Stale search after 5 node expansions
Total number of subsets evaluated: 12
Merit of best subset found:  96
Evaluation (for feature selection): CV (leave one out)
Feature set: 4,5

Time taken to build model: 0.02 seconds

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

Correctly Classified Instances      139      92.6667 %
Incorrectly Classified Instances    11       7.3333 %
Kappa statistic                      0.89
Mean absolute error                  0.092
Root mean squared error              0.2087
Relative absolute error              20.6978 %
Root relative squared error         44.2707 %
Total Number of Instances           150

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

      TP Rate   FP Rate   Precision   Recall   F-Measure   MCC   ROC Area   PRC Area   Class
1.000     0.000     1.000     1.000     1.000     1.000     1.000     1.000   Iris-setosa
0.880     0.050     0.898     0.880     0.889     0.834     0.946     0.861   Iris-versicolor
0.900     0.060     0.882     0.900     0.891     0.836     0.947     0.869   Iris-virginica
Weighted Avg.                     0.927     0.037     0.927     0.927     0.890     0.964     0.910

==== Confusion Matrix ====

      a   b   c   <-- classified as
50  0  0 |  a = Iris-setosa
0  44  6 |  b = Iris-versicolor
0  5  45 |  c = Iris-virginica

```

Logistic:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. A classifier named 'Logistic-R 1.0E-8-M -1-num-decimal-places 4' is chosen. The 'Test options' section shows 'Cross-validation' with 'Folds' set to 10. The 'Classifier output' pane displays the following text:

```

Classifier output
=====
Time taken to build model: 0.13 seconds

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

Correctly Classified Instances      2948      63.713 %
Incorrectly Classified Instances    1679      36.287 %
Kappa statistic                      0
Mean absolute error                  0.4624
Root mean squared error              0.4908
Relative absolute error              99.9961 %
Root relative squared error         100      %
Total Number of Instances           4627

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

      TP Rate   FP Rate   Precision   Recall   F-Measure   MCC   ROC Area   PRC Area   Class
1.000     0.000     0.637     1.000     0.778     ?     0.499     0.637   low
0.000     0.000     ?         0.000     ?         ?     0.499     0.362   high
Weighted Avg.                     0.637     0.637     ?         0.637     ?         ?     0.499     0.537

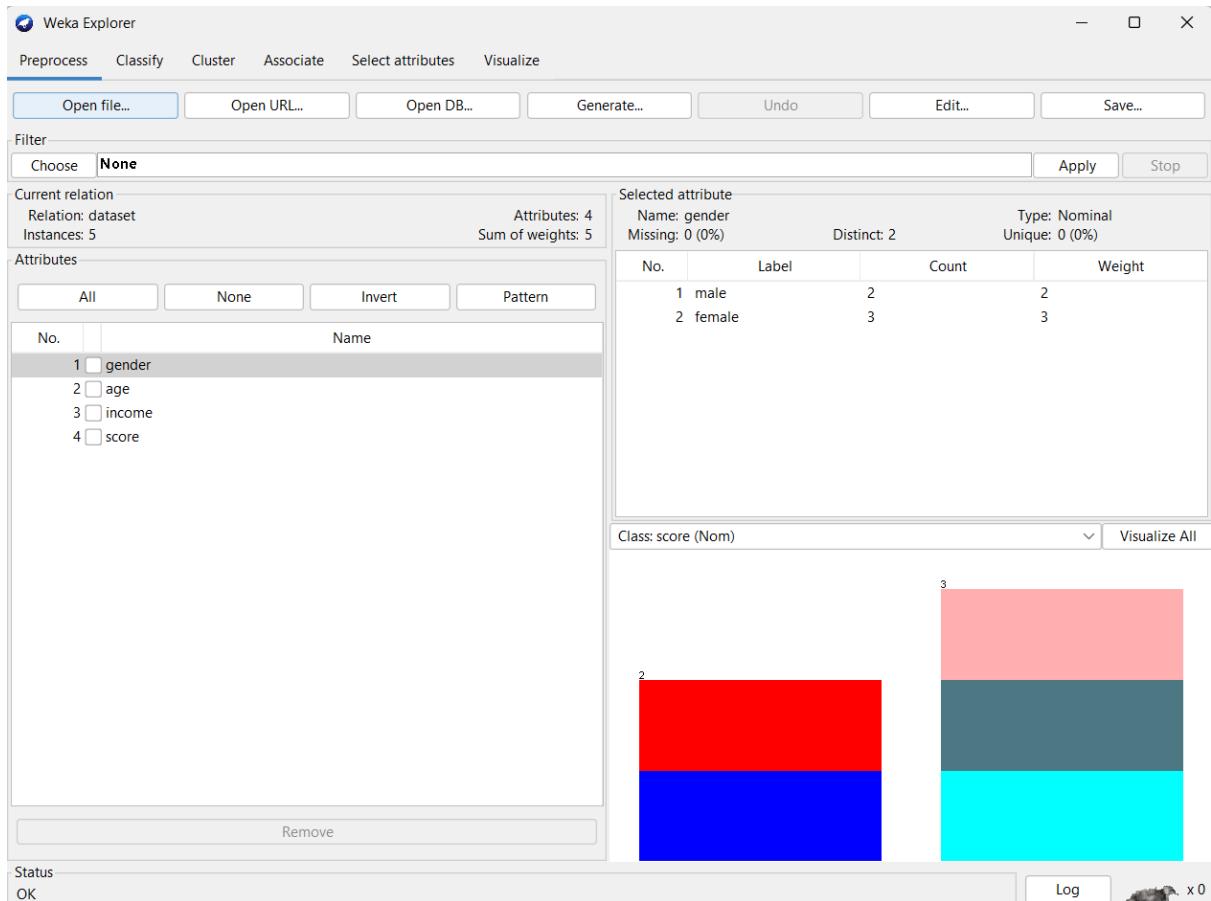
==== Confusion Matrix ====

      a   b   <-- classified as
2948  0 |  a = low
1679  0 |  b = high

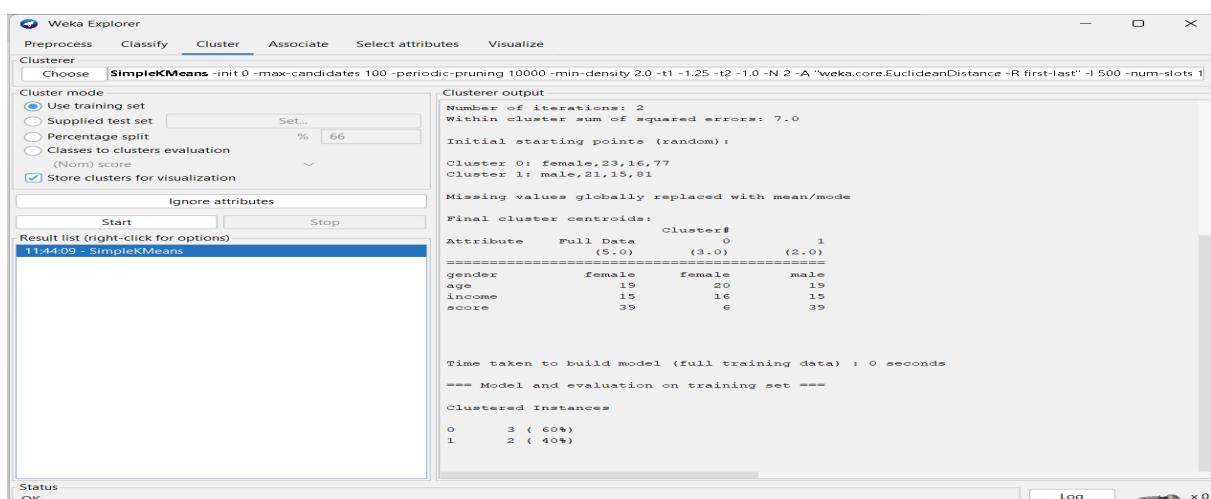
```

DAY-4

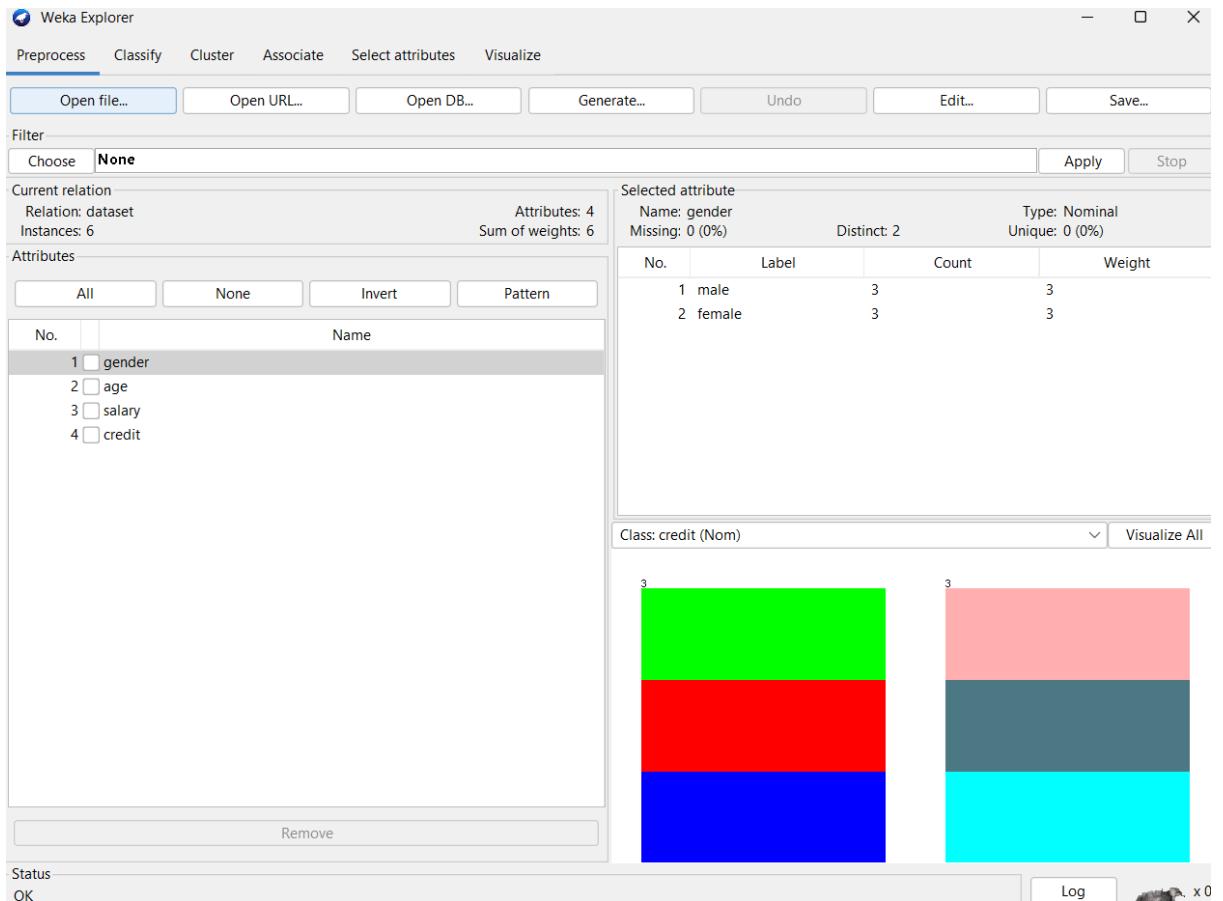
Q1:



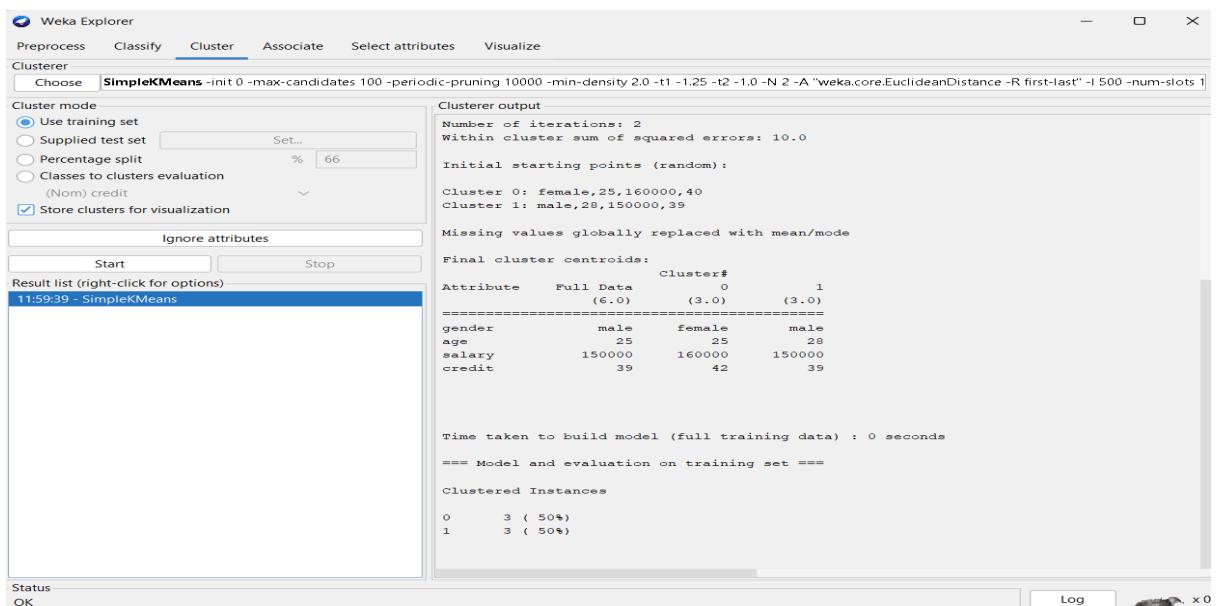
Clustering Analysis:



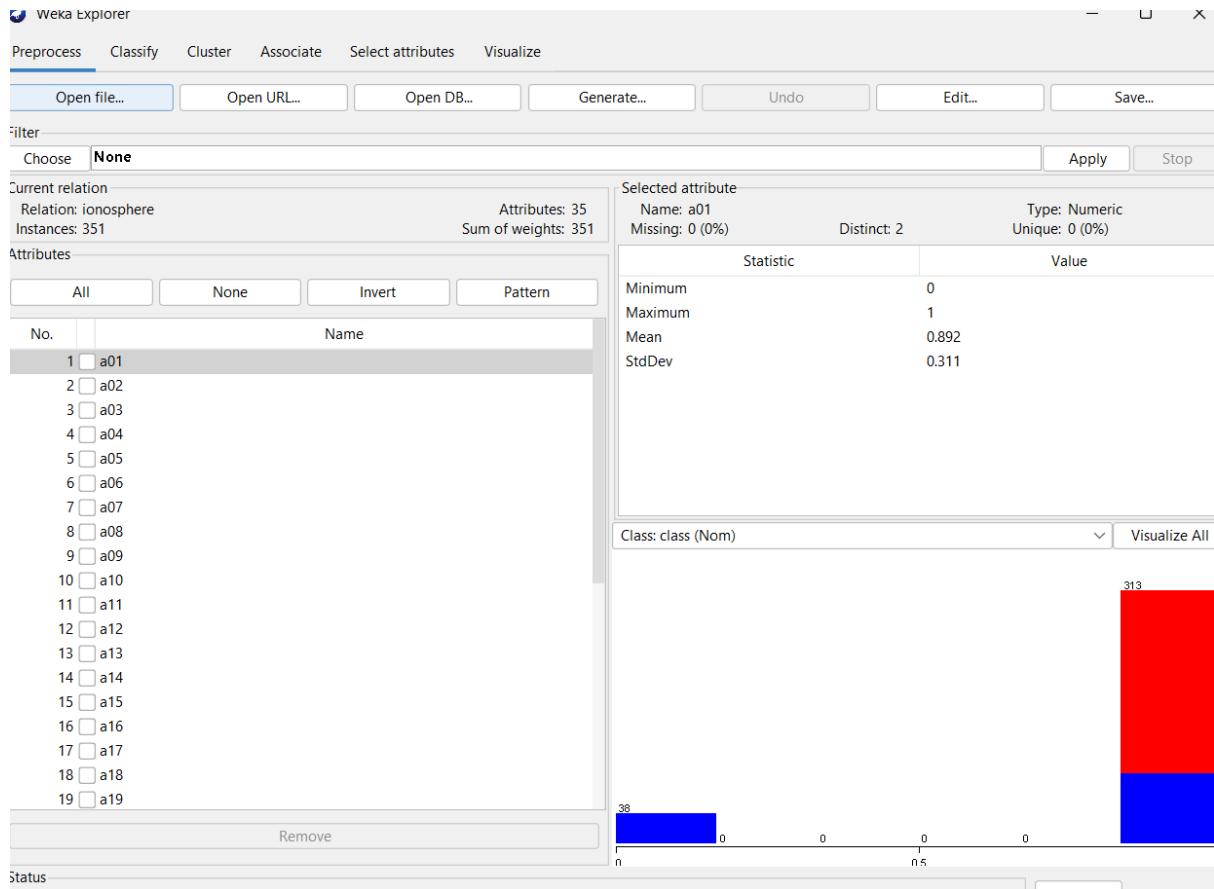
Q2:



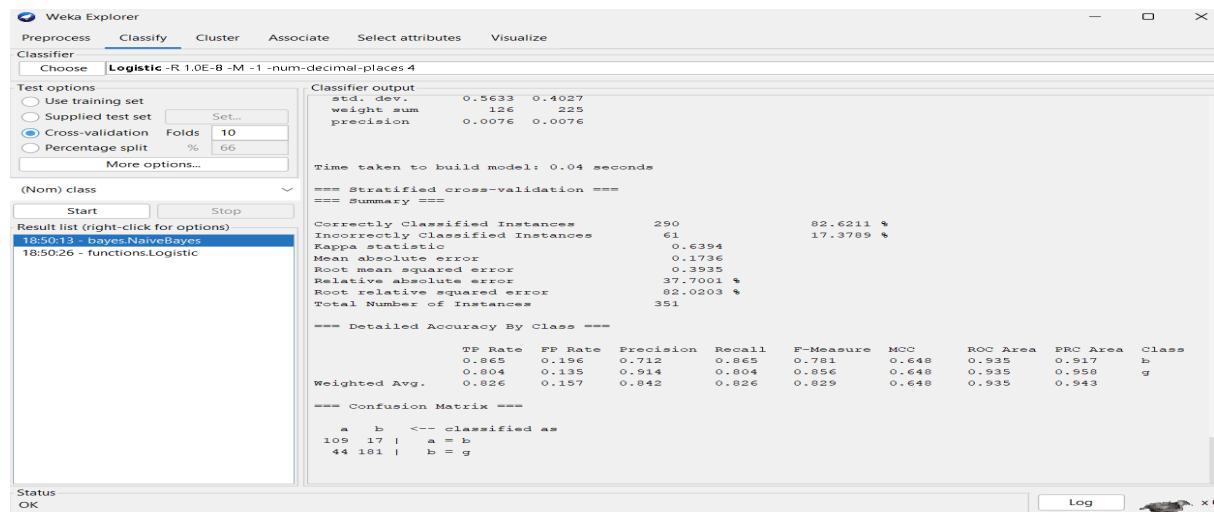
K-Means:



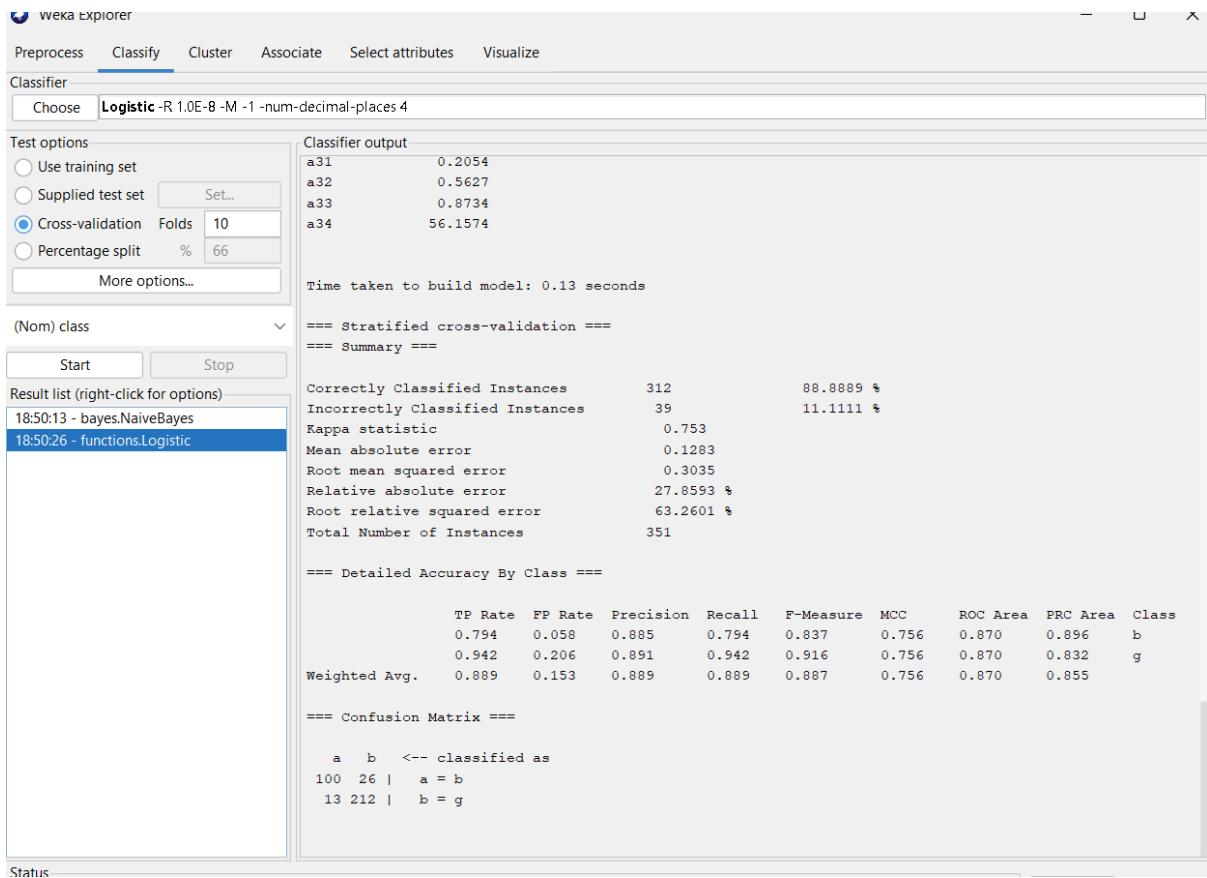
Q3:



Naïve Bayes classification:



SVM:

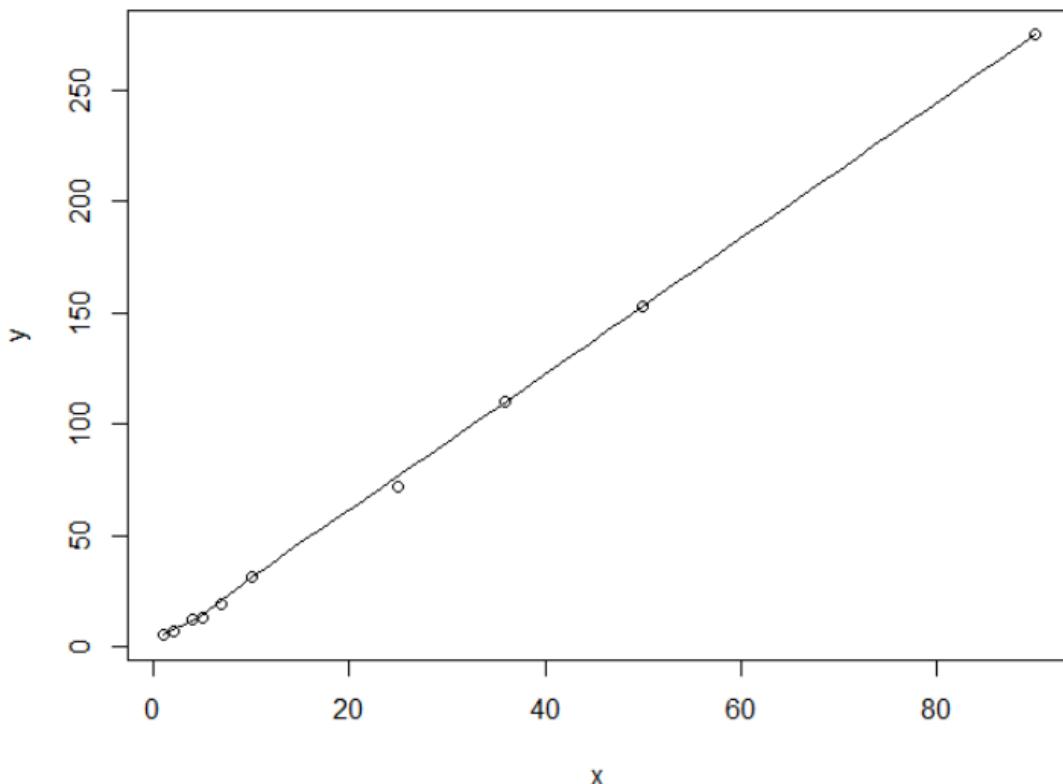


Q4:

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
+ - Go to file/function Addins -
7.R* Source on Save Run Source
1 # Data
2 persons <- c("Gopu", "Babu", "Baby", "Gopal", "Krishna", "Jai", "Dev", "Malini", "Hema", "Anu")
3 vegetarian_status <- c("yes", "yes", "yes", "no", "yes", "no", "no", "yes", "yes", "yes")
4
5 # Create a data frame
6 data <- data.frame(persons, vegetarian_status)
7
8 # Count the number of vegetarians and non-vegetarians
9 veg_count <- sum(vegetarian_status == "yes")
10
10 (Top Level) ▾
R 4.4.1 . ~/ ▾
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.

> # Data
> persons <- c("Gopu", "Babu", "Baby", "Gopal", "Krishna", "Jai", "Dev", "Malini", "Hema", "Anu")
> vegetarian_status <- c("yes", "yes", "yes", "no", "yes", "no", "no", "yes", "yes", "yes")
> # Create a data frame
> data <- data.frame(persons, vegetarian_status)
> # Count the number of vegetarians and non-vegetarians
> veg_count <- sum(vegetarian_status == "yes")
> non_veg_count <- sum(vegetarian_status == "no")
> # Determine which type has the greater count
> if (veg_count > non_veg_count) {
+   greater_count_type <- "Vegetarians"
+ } else if (non_veg_count > veg_count) {
+   greater_count_type <- "Non-Vegetarians"
+ } else {
+   greater_count_type <- "Both are equal"
+ }
> # Results
> cat("Number of Vegetarians:", veg_count, "\n")
Number of Vegetarians: 7
> cat("Number of Non-Vegetarians:", non_veg_count, "\n")
Number of Non-Vegetarians: 3
> cat("Type with Greater Count:", greater_count_type, "\n")
Type with Greater Count: Vegetarians
>
> |
```

Q5:



Q6:

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter Choose None Apply Stop

Current relation Relation: dataset Attributes: 6 Instances: 5 Sum of weights: 5

Attributes

No.	Name	Count	Weight
1	true	4	4
2	false	1	1

Selected attribute Name: bread Type: Nominal Missing: 0 (0%) Distinct: 2 Unique: 1 (20%)

Class: Yogurt (Nom) Visualize All

Remove

Status OK Log x 0

4

1

FP GROWTH:

Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associate

Choose **FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1**

Start Stop

Result list (right-click for ...)

17:42:10 - FPGrowth

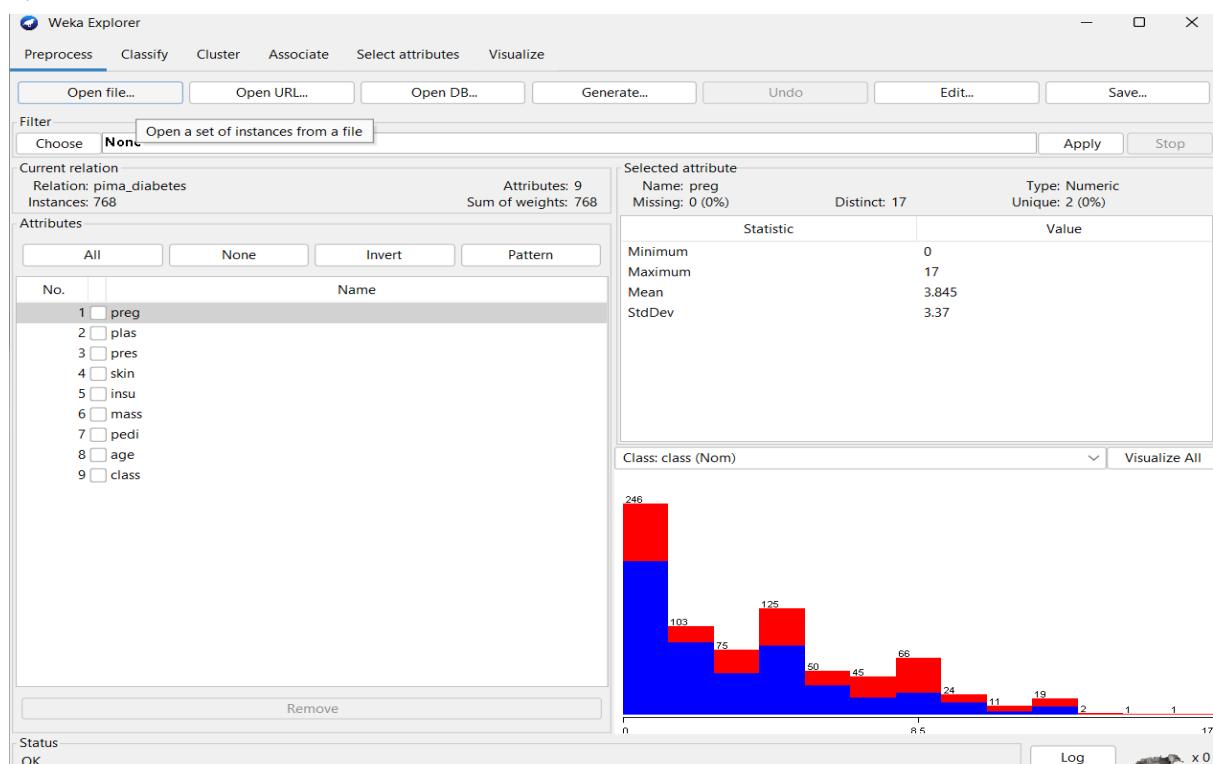
Associate output

```
== Run information ==
Scheme:      weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1
Relation:     dataset
Instances:    5
Attributes:   6
              bread
              cheese
              Egg
              juice
              milk
              Yogurt
== Associate model (full training set) ==
FPGrowth found 14 rules (displaying top 10)

1. [milk=false]: 2 ==> [Yogurt=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
2. [bread=false]: 1 ==> [Yogurt=false]: 1 <conf:(1)> lift:(1.25) lev:(0.04) conv:(0.4)
3. [cheese=false]: 2 ==> [Egg=false]: 2 <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
4. [juice=false]: 1 ==> [Egg=false]: 1 <conf:(1)> lift:(1.25) lev:(0.04) conv:(0.2)
5. [bread=false]: 1 ==> [Egg=false]: 1 <conf:(1)> lift:(1.25) lev:(0.04) conv:(0.2)
6. [juice=false]: 1 ==> [cheese=false]: 1 <conf:(1)> lift:(2.5) lev:(0.12) conv:(0.6)
7. [Egg=false, milk=false]: 1 ==> [Yogurt=false]: 1 <conf:(1)> lift:(1.25) lev:(0.04) conv:(0.2)
8. [Yogurt=false, cheese=false]: 1 ==> [Egg=false]: 1 <conf:(1)> lift:(1.25) lev:(0.04) conv:(0.2)
9. [bread=false]: 1 ==> [Yogurt=false, Egg=false]: 1 <conf:(1)> lift:(1.67) lev:(0.08) conv:(0.4)
10. [Yogurt=false, bread=false]: 1 ==> [Egg=false]: 1 <conf:(1)> lift:(1.25) lev:(0.04) conv:(0.2)
```

Status OK Log x 0

Q7:



Decision Tree Classifier:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. Under 'Classifier', 'Logistic -R 1.0E-8 -M 1 -num-decimal-places 4' is chosen. The 'Test options' panel shows 'Cross-validation' with 'Folds 10'. The 'Classifier output' pane displays the following text:

```
Number of Leaves : 20
Size of the tree : 39
Time taken to build model: 0.19 seconds
== Stratified cross-validation ==
== Summary ==
Correctly Classified Instances      567      73.8281 %
Incorrectly Classified Instances    201      26.1719 %
Kappa statistic                      0.4164
Mean absolute error                  0.3158
Root mean squared error              0.4463
Relative absolute error              69.4841 %
Root relative squared error         93.6293 %
Total Number of Instances            768

== Detailed Accuracy By Class ==

```

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.814	0.403	0.790	0.814	0.802	0.417	0.751	0.811	tested_n
0.597	0.186	0.632	0.597	0.614	0.417	0.751	0.572	tested_F
Weighted Avg.	0.738	0.327	0.735	0.738	0.736	0.417	0.751	0.727

```
== Confusion Matrix ==

```

		a	b	-- classified as
a	407	93		a = tested_negative
	108	160		b = tested_positive

Status: OK Log: x 0

Support Vector Machine Classifier:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. Under 'Classifier', 'Logistic -R 1.0E-8 -M 1 -num-decimal-places 4' is chosen. The 'Test options' panel shows 'Cross-validation' with 'Folds 10'. The 'Classifier output' pane displays the following text:

```
mass           0.9142
pedi          0.3886
age            0.9852

Time taken to build model: 0.17 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	PRC Area	Class
0.880	0.429	0.793	0.880	0.834	0.480	0.832	0.892	tested_n
0.571	0.120	0.718	0.571	0.636	0.480	0.832	0.715	tested_F
Weighted Avg.	0.772	0.321	0.767	0.772	0.765	0.480	0.832	0.831

```
== Confusion Matrix ==

```

		a	b	-- classified as
a	440	60		a = tested_negative
	115	153		b = tested_positive

Q8:

INPUT: marks <- c(55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65, 67, 71, 72, 75)

```
bins_a <- cut(marks, breaks = 3, labels = c("Low", "Medium", "High"))
```

```
bins_b <- cut(marks, breaks = seq(min(marks), max(marks), length.out = 4), labels = c("Low", "Medium", "High"))
```

```
k <- 3
```

```
clusters <- kmeans(matrix(marks), centers = k)
```

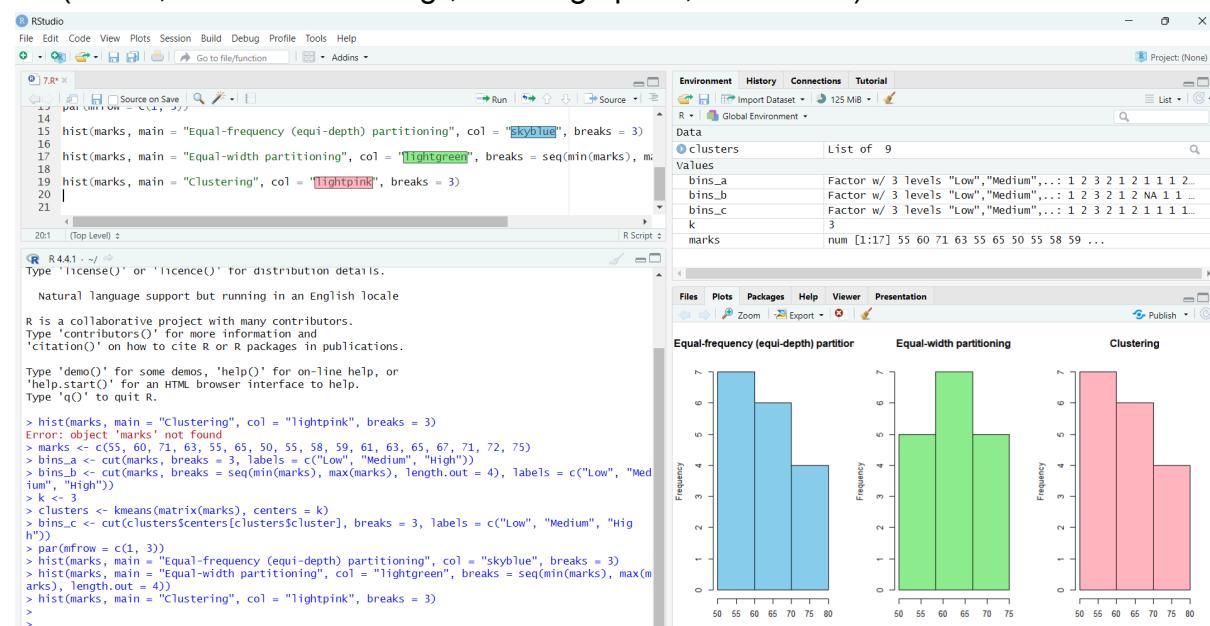
```
bins_c <- cut(clusters$centers[clusters$cluster], breaks = 3, labels = c("Low", "Medium", "High"))
```

```
par(mfrow = c(1, 3))
```

```
hist(marks, main = "Equal-frequency (equi-depth) partitioning", col = "skyblue", breaks = 3)
```

```
hist(marks, main = "Equal-width partitioning", col = "lightgreen", breaks = seq(min(marks), max(marks), length.out = 4))
```

```
hist(marks, main = "Clustering", col = "lightpink", breaks = 3)
```



Q9:

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 -C 0.25 -M 2

Test options

Use training set

Supplied test set Set...

Cross-validation Folds 10

Percentage split % 66

More options...

(Nom) gender

Start Stop

Result list (right-click for options)

12:29:34 - treesJ48

J48 pruned tree

height <= 170: female (4.0)
height > 170: male (6.0)

Number of Leaves : 2

Size of the Tree : 3

Time taken to build model: 0 seconds

--- Stratified cross-validation ---

--- Summary ---

Correctly Classified Instances	80	80	8
Incorrectly Classified Instances	2	20	8
Kappa statistic	0.5833		
Mean absolute error	0.2		
Root mean squared error	0.4472		
Relative absolute error	37.931 %		
Root relative squared error	83.6315 %		
Total Number of Instances	100		

--- Detailed Accuracy By Class ---

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0 b	0.833	0.250	0.833	0.833	0.833	0.583	0.792	0.794	male
0 a	0.750	0.167	0.750	0.750	0.750	0.583	0.792	0.663	female
weighted Avg.	0.800	0.217	0.800	0.800	0.800	0.583	0.792	0.742	

--- Confusion Matrix ---

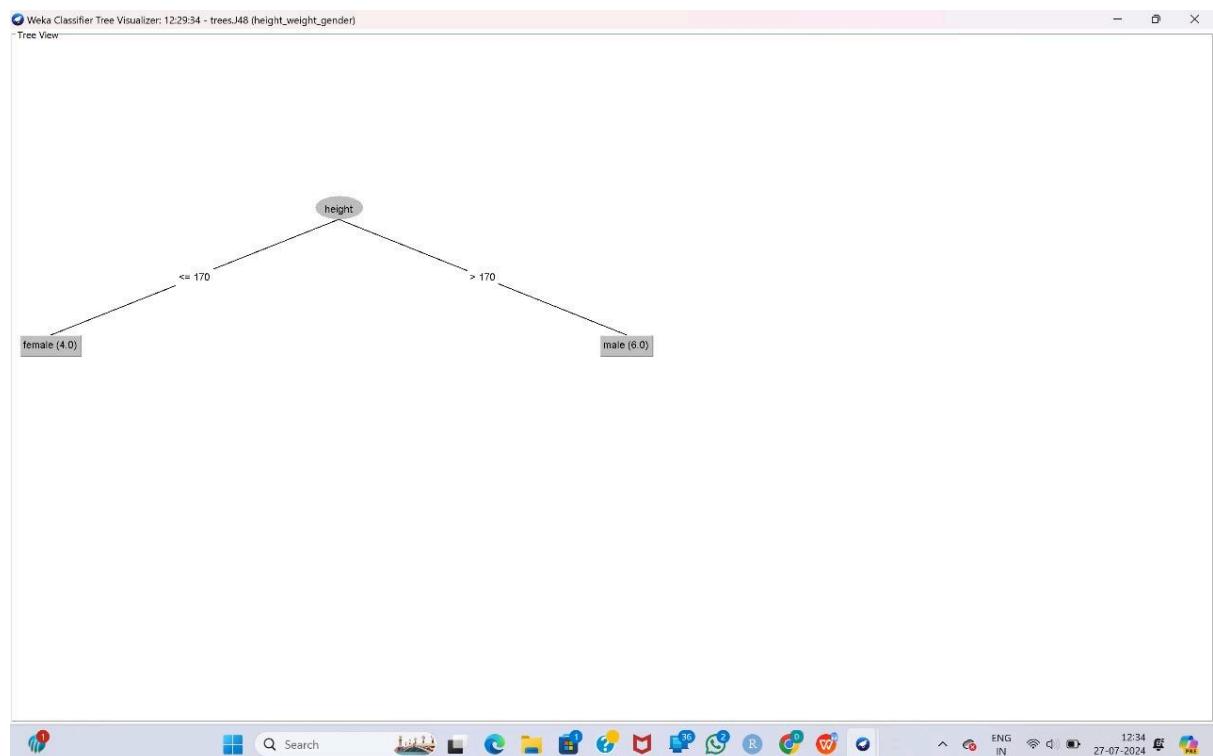
a | b <- classified as
b | a = male
1 | 0 = female

Status OK

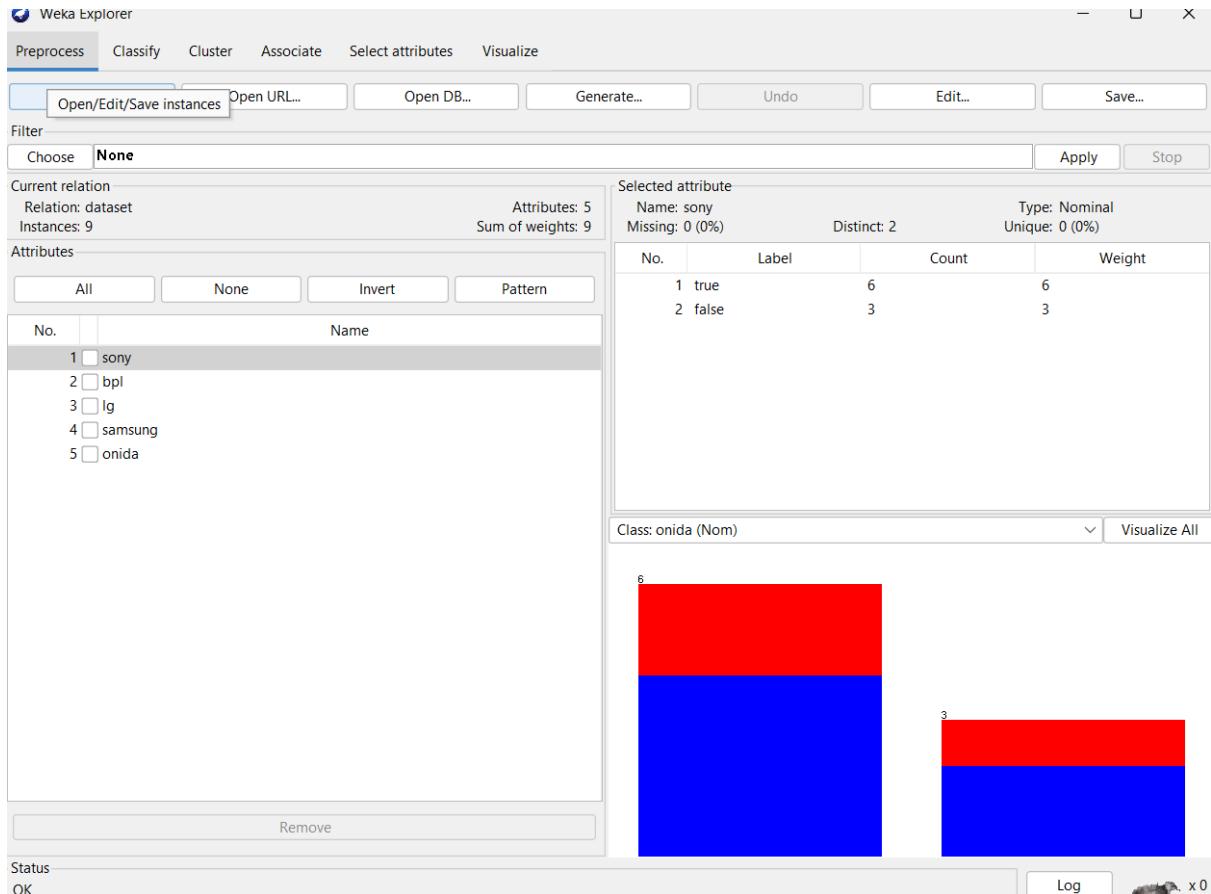
Log x 0

12:29:34 27-07-2024

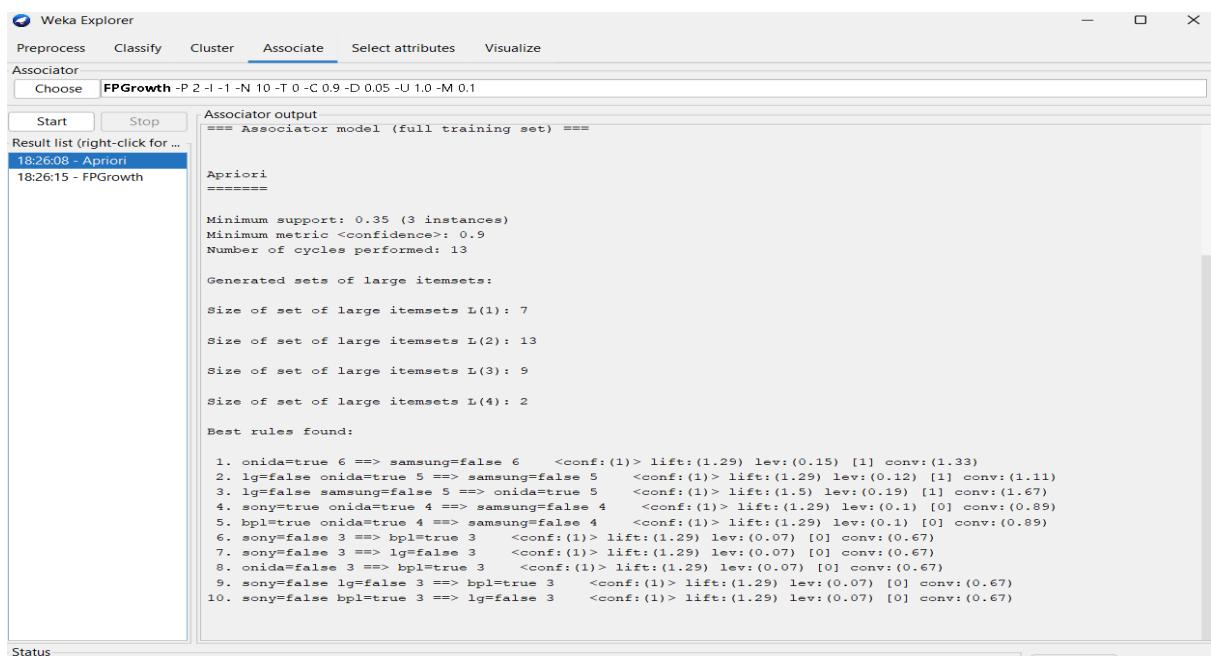
Decision Tree :



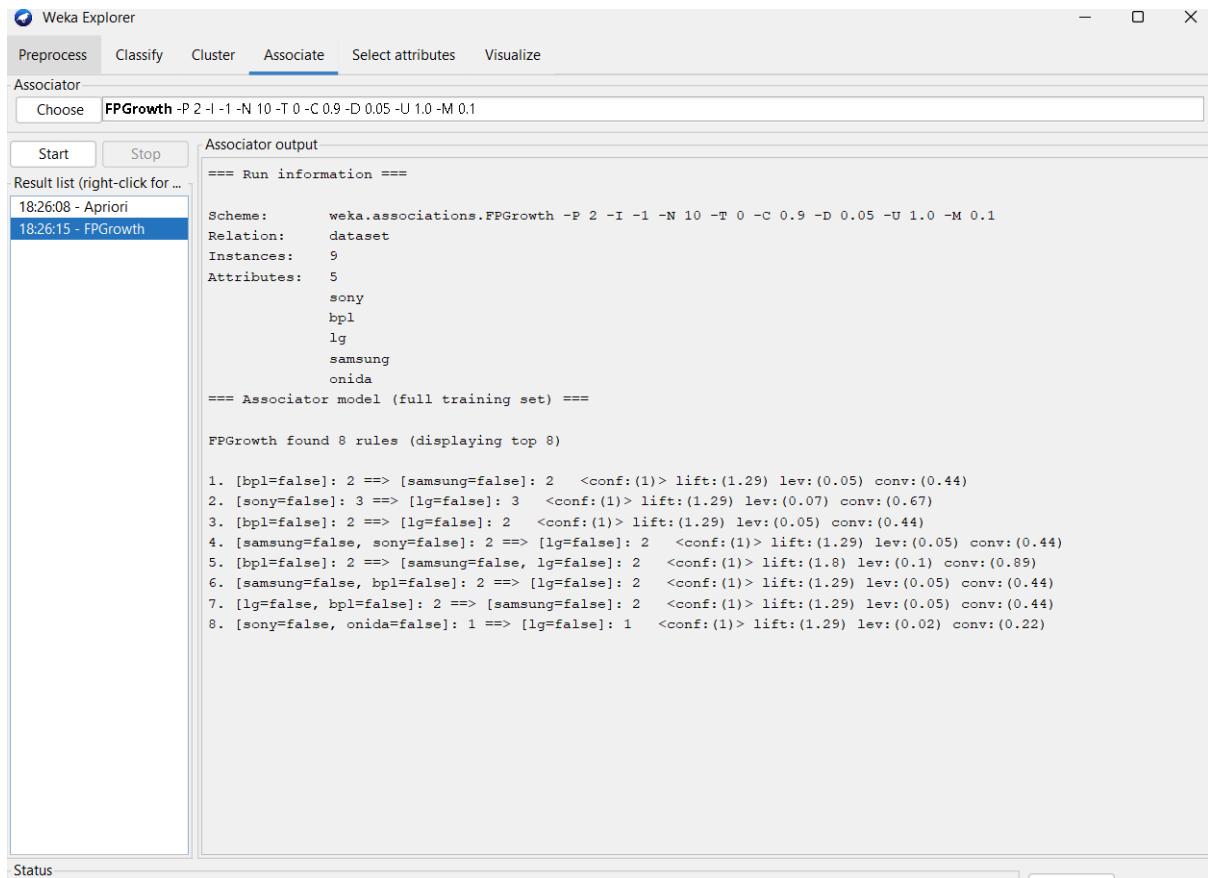
Q10:



Apriori Algorithm:



FP GROWTH:



Q11:

```

# Data
strike_rates <- c(100, 70, 60, 90, 90)

# (a) Min-Max Normalization
min_value <- min(strike_rates)
max_value <- max(strike_rates)
min_max_normalization <- (strike_rates - min_value) / (max_value - min_value)

# (b) Z-Score Normalization
mean_sr <- mean(strike_rates)
sd_sr <- sd(strike_rates)
z_score_normalization <- (strike_rates - mean_sr) / sd_sr

# (c) Z-Score Normalization using Mean Absolute Deviation (MAD)
mad_sr <- mean(abs(strike_rates - mean_sr))
z_score_mad_normalization <- (strike_rates - mean_sr) / mad_sr

# (d) Normalization by Decimal Scaling
j <- ceiling(log10(max(abs(strike_rates))))
decimal_scaling_normalization <- strike_rates / (10^j)

# Results
cat("Min-Max Normalization:\n", min_max_normalization, "\n")
Min-Max Normalization:
1 0.25 0 0.75 0.75
cat("Z-Score Normalization:\n", z_score_normalization, "\n")
Z-Score Normalization:
1.095445 -0.7302967 -1.338877 0.4868645 0.4868645
cat("Z-Score Normalization using MAD:\n", z_score_mad_normalization, "\n")
Z-Score Normalization using MAD:
1.323529 -0.8823529 -1.617647 0.5882353 0.5882353
cat("Normalization by Decimal Scaling:\n", decimal_scaling_normalization, "\n")
Normalization by Decimal Scaling:
1 0.7 0.6 0.9 0.9

```

Q12:

KStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Source on Save Go to file/function Addins

```
ency.R x age.R* x smoothing.R* x Untitled1* x Untitled2* x Untitled3* x Untitled4* x Untitled5* x Run Source
```

```
10 var_AvgSpeed <- var(AvgSpeed)
11 var_TotalTime <- var(TotalTime)
12
13 # Results
14 cat("Standard Deviation of AvgSpeed:", sd_AvgSpeed, "\n")
15 cat("Standard Deviation of TotalTime:", sd_TotalTime, "\n")
16 cat("Variance of AvgSpeed:", var_AvgSpeed, "\n")
17 cat("Variance of TotalTime:", var_TotalTime, "\n")
18
19
```

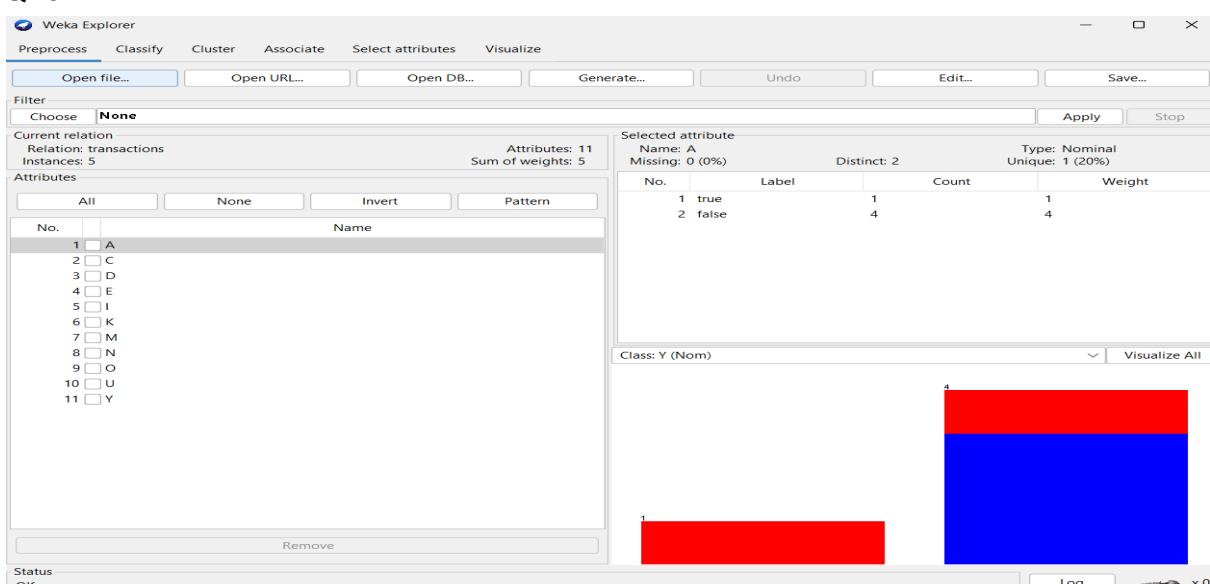
18:1 (Top Level) R Script

```
R 4.4.1 · ~/ 
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.

>
> # Data
> AvgSpeed <- c(78, 81, 82, 74, 83, 82, 77, 80, 70)
> TotalTime <- c(39, 37, 36, 42, 35, 36, 40, 38, 46)
> # Standard Deviation
> sd_AvgSpeed <- sd(AvgSpeed)
> sd_TotalTime <- sd(TotalTime)
> # Variance
> var_AvgSpeed <- var(AvgSpeed)
> var_TotalTime <- var(TotalTime)
> # Results
> cat("Standard Deviation of AvgSpeed:", sd_AvgSpeed, "\n")
Standard Deviation of AvgSpeed: 4.304391
> cat("Standard Deviation of TotalTime:", sd_TotalTime, "\n")
Standard Deviation of TotalTime: 3.492054
> cat("Variance of AvgSpeed:", var_AvgSpeed, "\n")
Variance of AvgSpeed: 18.52778
> cat("Variance of TotalTime:", var_TotalTime, "\n")
Variance of TotalTime: 12.19444
>
>
> |
```

Q13:



Apriori Algorithm:

The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Choose" dropdown is set to "FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1". The "Result list" pane shows "10:04:05 - Apriori" and "10:04:11 - FPGrowth". The "Associator output" pane displays the following log:

```
U
Y
==== Associator model (full training set) ====
Apriori
=====
Minimum support: 0.85 (4 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 3

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): 1

Best rules found:

1. A=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
2. D=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
3. E=true 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
4. U=false 4 ==> E=true 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
5. E=true 4 ==> U=false 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
6. I=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
7. U=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
8. K=true U=false 4 ==> E=true 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
9. E=true U=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
10. E=true K=true 4 ==> U=false 4   <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
```

FP Growth:

The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Choose" dropdown is set to "FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1". The "Result list" pane shows "10:04:05 - Apriori" and "10:04:11 - FPGrowth". The "Associator output" pane displays the following log:

```
=====
Run information ===
Scheme:      weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1
Relation:     transactions
Instances:   5
Attributes:  11
A
C
D
E
I
K
M
N
O
U
Y
==== Associator model (full training set) ====
FPGrowth found 71 rules (displaying top 10)

1. [C=false]: 3 ==> [U=false]: 3    <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
2. [Y=false]: 2 ==> [U=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
3. [M=false]: 2 ==> [U=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
4. [O=false]: 3 ==> [I=false]: 3    <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
5. [O=false]: 2 ==> [I=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
6. [N=false]: 3 ==> [D=false]: 3    <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
7. [Y=false]: 2 ==> [D=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
8. [O=false]: 2 ==> [D=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
9. [M=false]: 2 ==> [A=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
10. [Y=false]: 2 ==> [N=false]: 2   <conf:(1)> lift:(1.67) lev:(0.16) conv:(0.8)
```

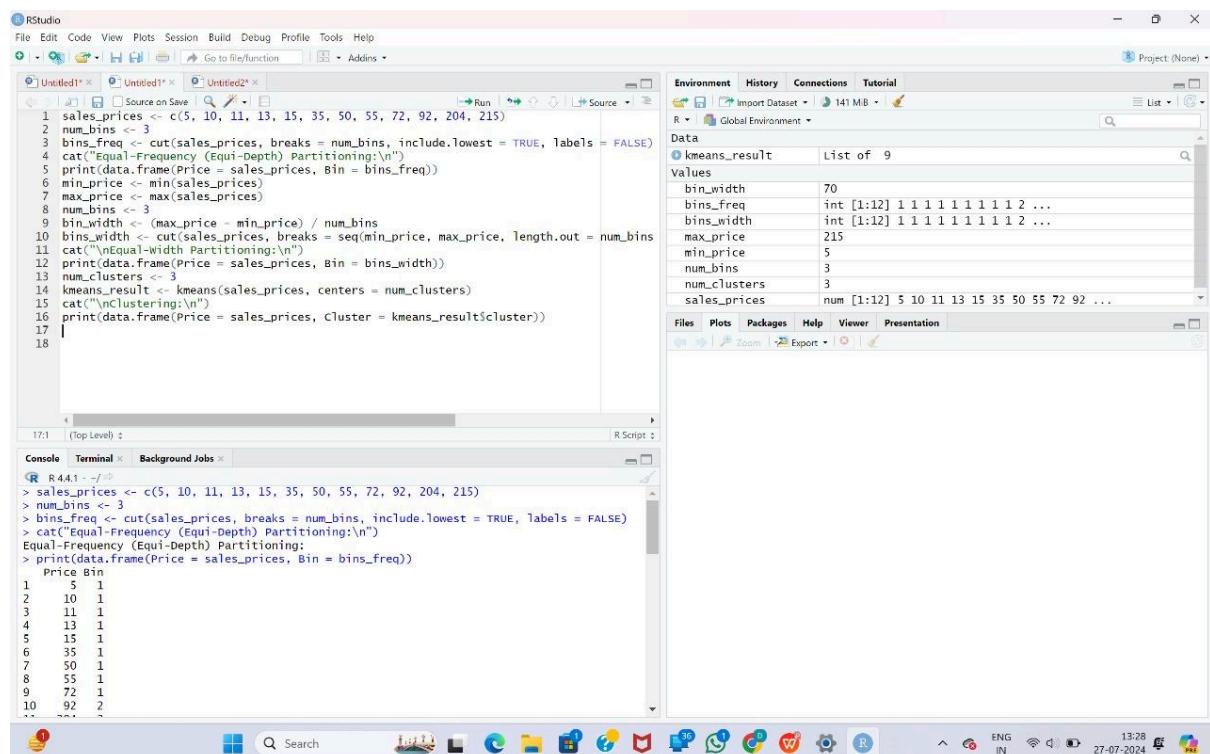
QUESTION BANK

Q1:

INPUT:

```
sales_prices <- c(5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215)
num_bins <- 3
bins_freq <- cut(sales_prices, breaks = num_bins, include.lowest = TRUE, labels = FALSE)
cat("Equal-Frequency (Equi-Depth) Partitioning:\n")
print(data.frame(Price = sales_prices, Bin = bins_freq))
min_price <- min(sales_prices)
max_price <- max(sales_prices)
num_bins <- 3
bin_width <- (max_price - min_price) / num_bins
bins_width <- cut(sales_prices, breaks = seq(min_price, max_price, length.out = num_bins + 1), include.lowest = TRUE, labels = FALSE)
cat("\nEqual-Width Partitioning:\n")
print(data.frame(Price = sales_prices, Bin = bins_width))
num_clusters <- 3
kmeans_result <- kmeans(sales_prices, centers = num_clusters)
cat("\nClustering:\n")
print(data.frame(Price = sales_prices, Cluster = kmeans_result$cluster))
```

OUTPUT:

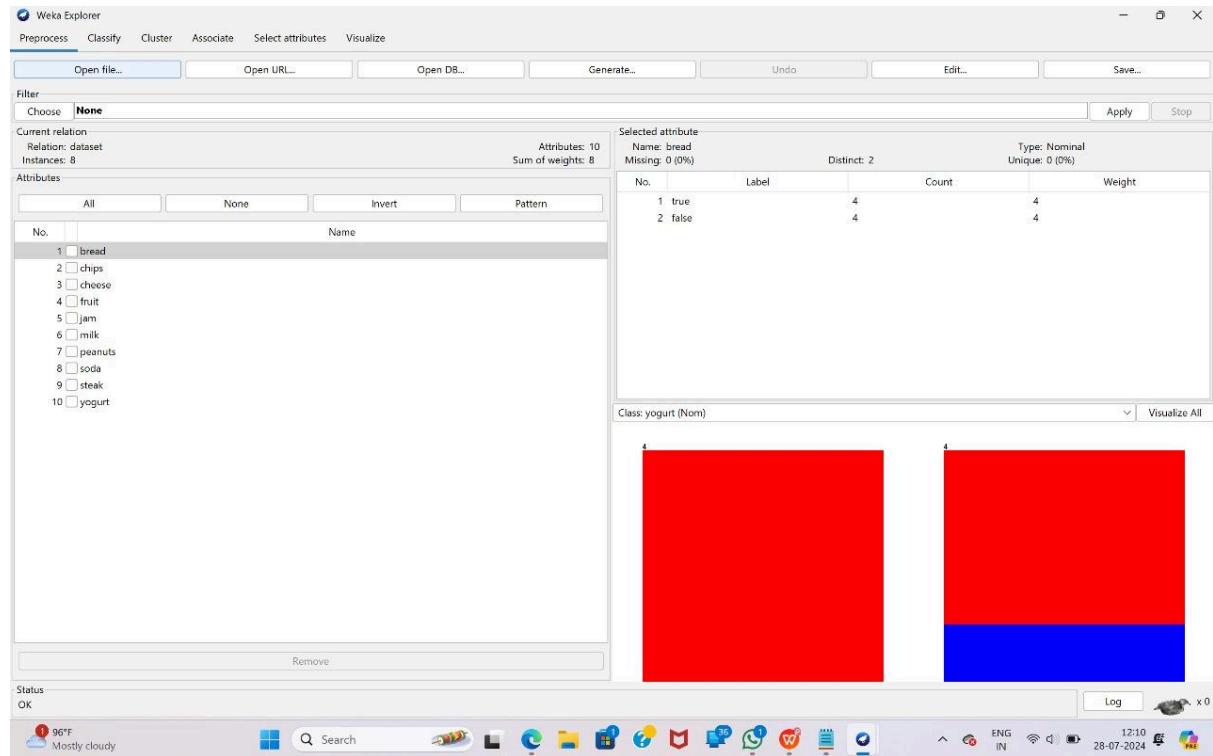


The screenshot shows the RStudio interface with the following details:

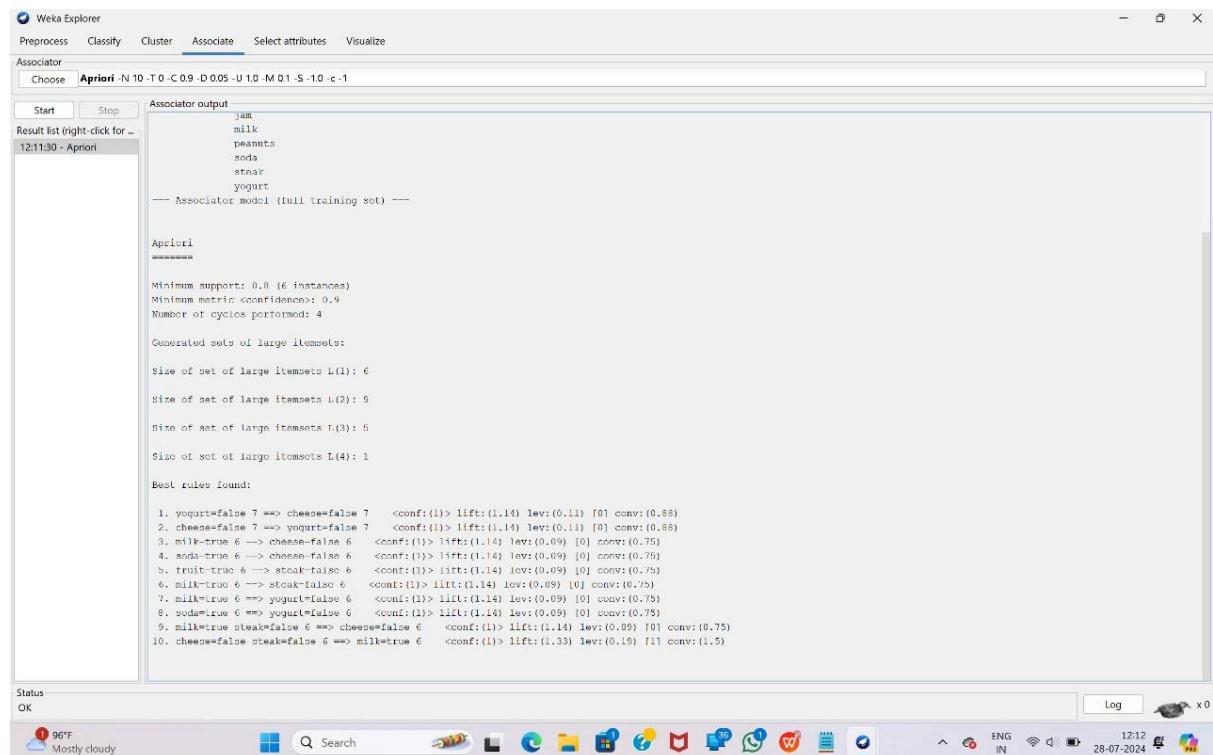
- Script Editor:** Displays the R script provided in the question.
- Data View:** Shows the data frame `kmeans_result` with columns: bin_width, bins_freq, max_price, min_price, num_bins, num_clusters, and sales_prices.
- Console:** Displays the R session output, showing the execution of the script and the resulting data frame.
- Bottom Taskbar:** Shows the Windows taskbar with various application icons.

```
R 4.4.1 -- / 
> sales_prices <- c(5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215)
> num_bins <- 3
> bins_freq <- cut(sales_prices, breaks = num_bins, include.lowest = TRUE, labels = FALSE)
> cat("Equal-Frequency (Equi-Depth) Partitioning:\n")
> print(data.frame(Price = sales_prices, Bin = bins_freq))
  Price Bin
1      5    1
2     10    1
3     11    1
4     13    1
5     35    1
6     50    1
8     55    1
9     72    1
10    92    2
11    204   2
12    215   2
```

Q3:



Apriori Algorithm :

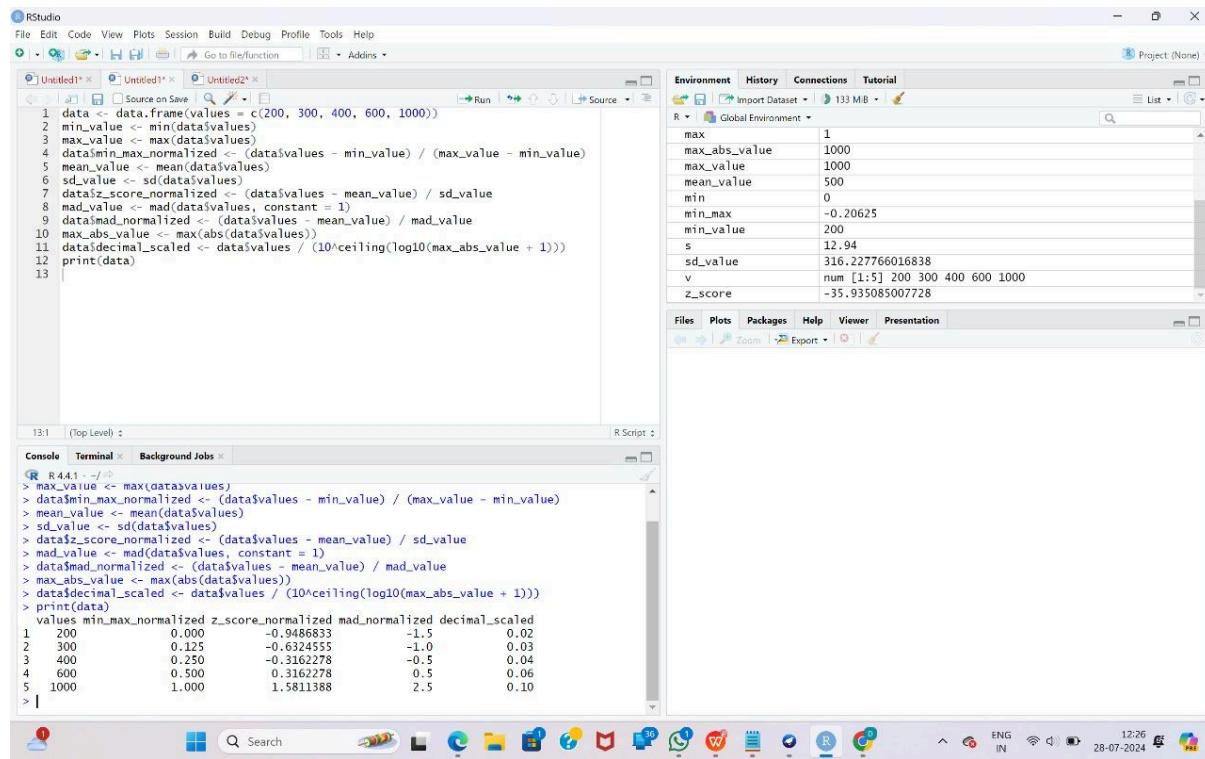


Q4:

INPUT:

```
data <- data.frame(values = c(200, 300, 400, 600, 1000))
min_value <- min(data$values)
max_value <- max(data$values)
data$min_max_normalized <- (data$values - min_value) / (max_value - min_value)
mean_value <- mean(data$values)
sd_value <- sd(data$values)
data$z_score_normalized <- (data$values - mean_value) / sd_value
mad_value <- mad(data$values, constant = 1)
data$mad_normalized <- (data$values - mean_value) / mad_value
max_abs_value <- max(abs(data$values))
data
```

Output:



The screenshot shows the RStudio interface with the following details:

- Code Editor:** Contains the R script provided in the question.
- Environment View:** Shows the following variables and their values:

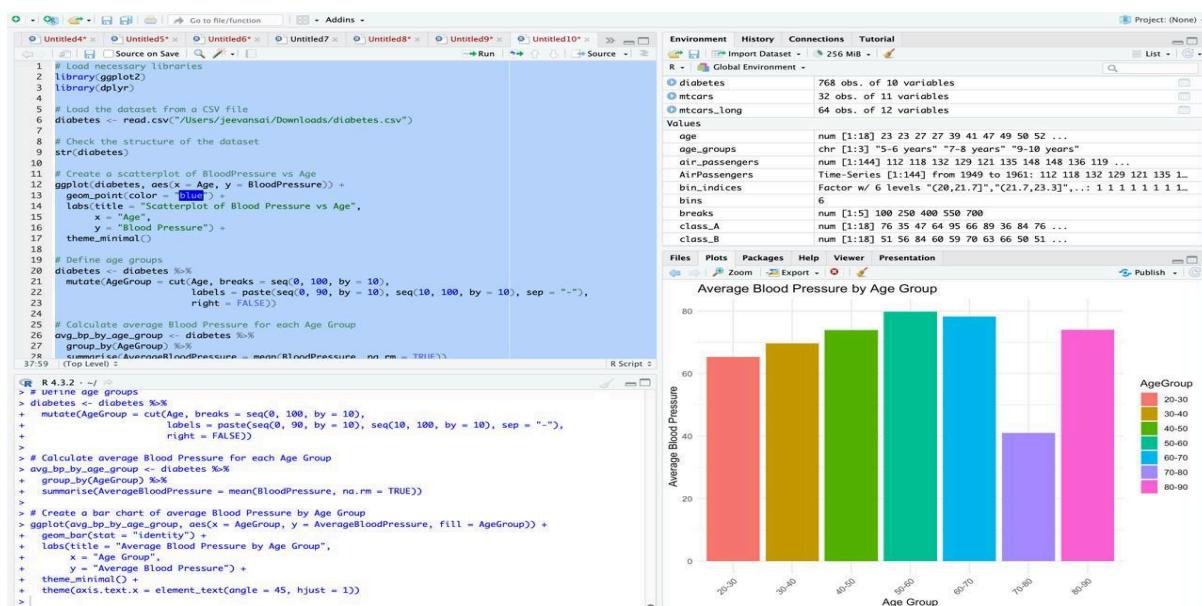
max	1
max_abs_value	1000
max_value	1000
mean_value	500
min	0
min_max	-0.20625
min_value	200
s	12.94
sd_value	316.227766016838
v	num [1:5] 200 300 400 600 1000
Z_score	-35.935085007728
- Console View:** Displays the R session output, showing the execution of the script and the resulting data frame.
- System Tray:** Shows the date and time as 28-07-2024 12:26.

Q5:

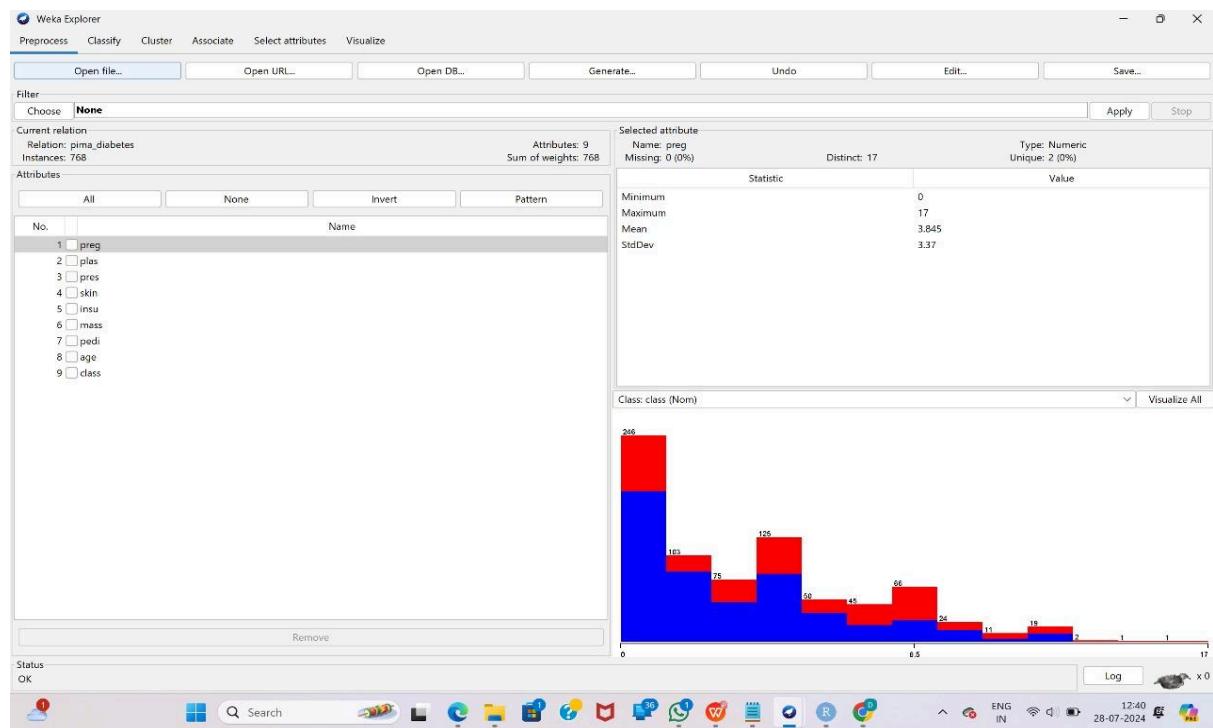
Input:

```
library(ggplot2)
library(dplyr)
diabetes <- read.csv("D:/DWDM/diabetes.csv")
str(diabetes)
ggplot(diabetes, aes(x = Age, y = BloodPressure)) +
  geom_point(color = "blue") +
  labs(title = "Scatterplot of Blood Pressure vs Age",
       x = "Age",
       y = "Blood Pressure") +
  theme_minimal()
diabetes <- diabetes %>%
  mutate(AgeGroup = cut(Age, breaks = seq(0, 100, by = 10),
                        labels = paste(seq(0, 90, by = 10), seq(10, 100, by = 10), sep = "-"),
                        right = FALSE))
avg_bp_by_age_group <- diabetes %>%
  group_by(AgeGroup) %>%
  summarise(AverageBloodPressure = mean(BloodPressure, na.rm = TRUE))
ggplot(avg_bp_by_age_group, aes(x = AgeGroup, y = AverageBloodPressure, fill =
AgeGroup)) +
  geom_bar(stat = "identity") +
  labs(title = "Average Blood Pressure by Age Group",
       x = "Age Group",
       y = "Average Blood Pressure") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

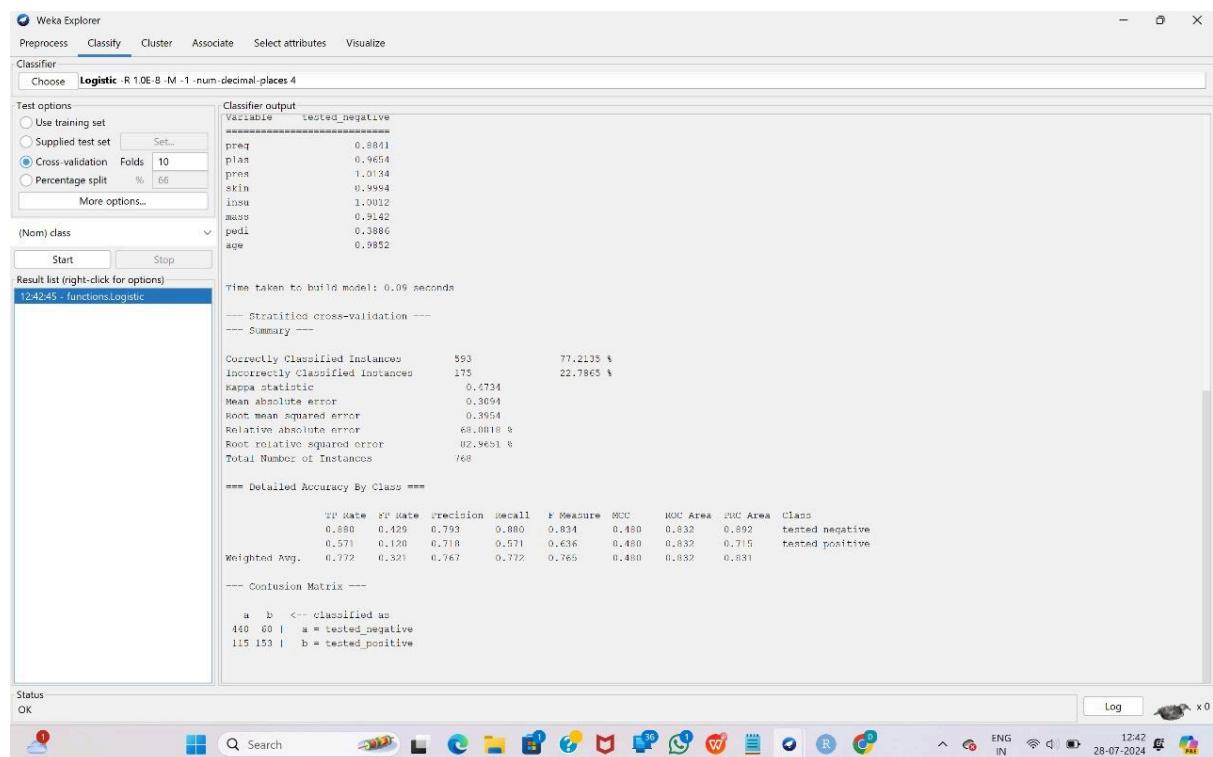
OUTPUT:



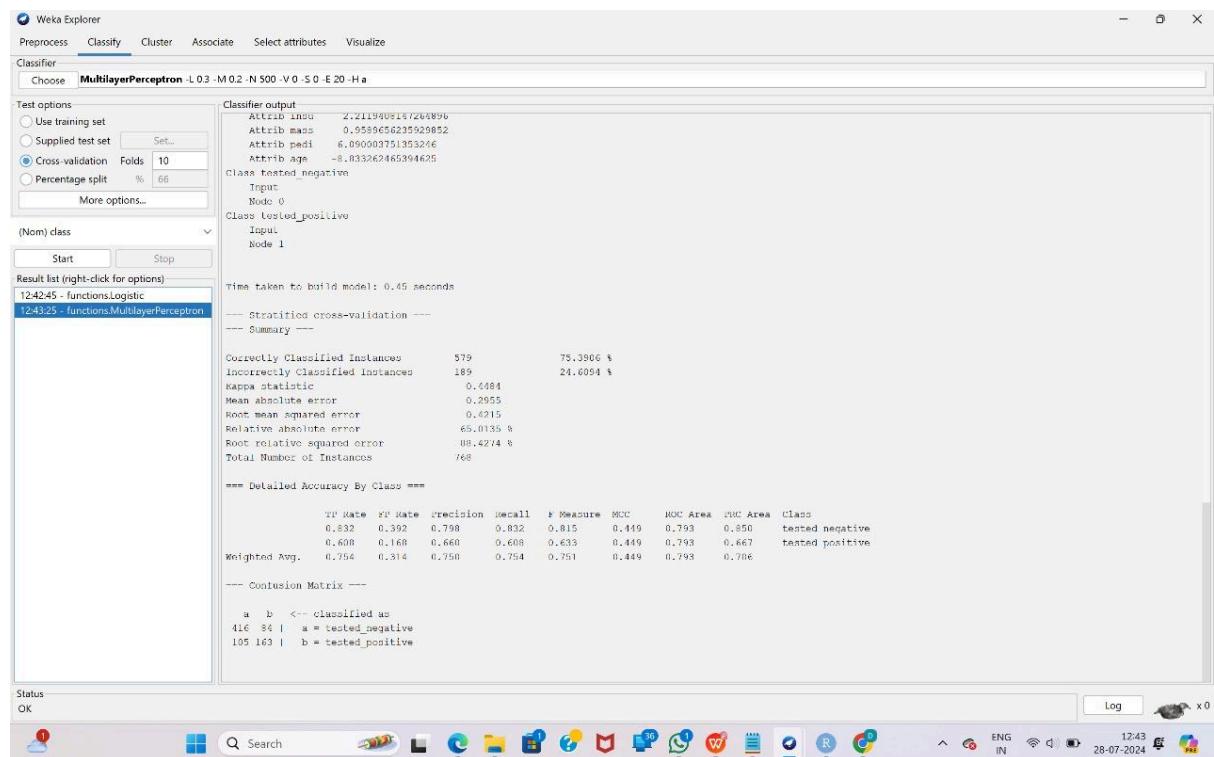
Q6:



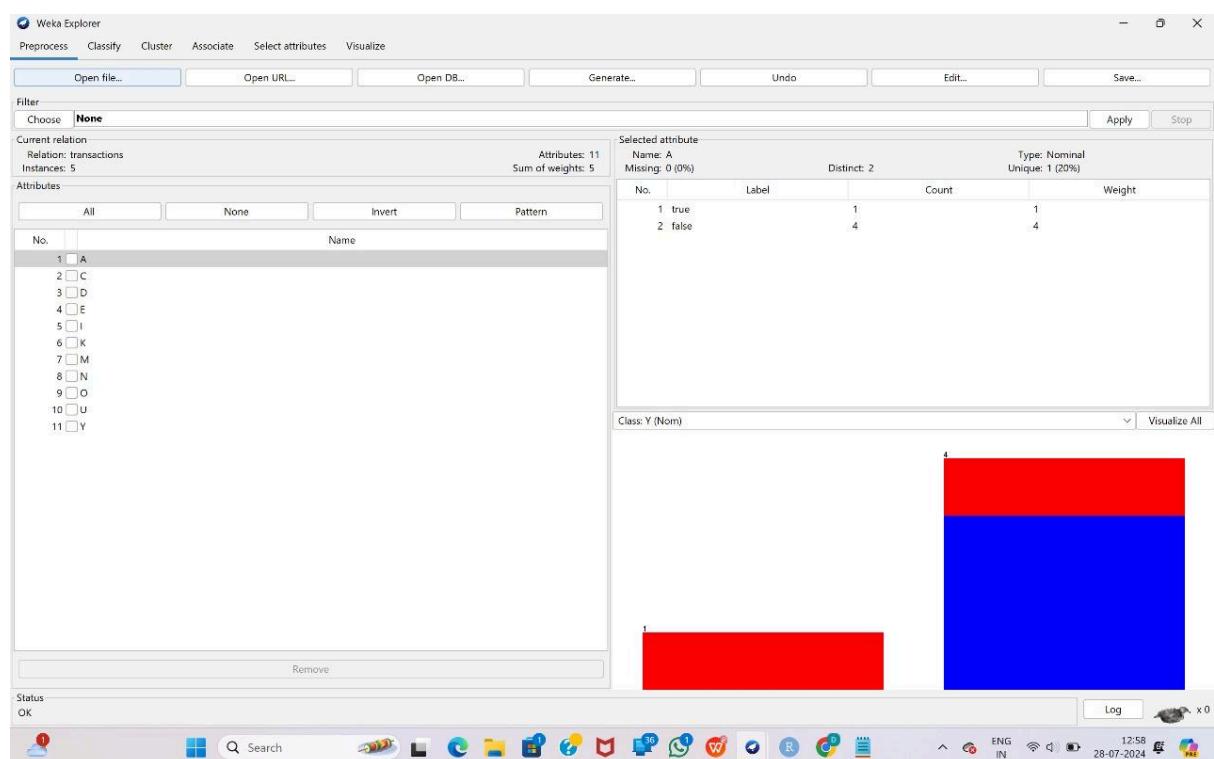
Linear regression:



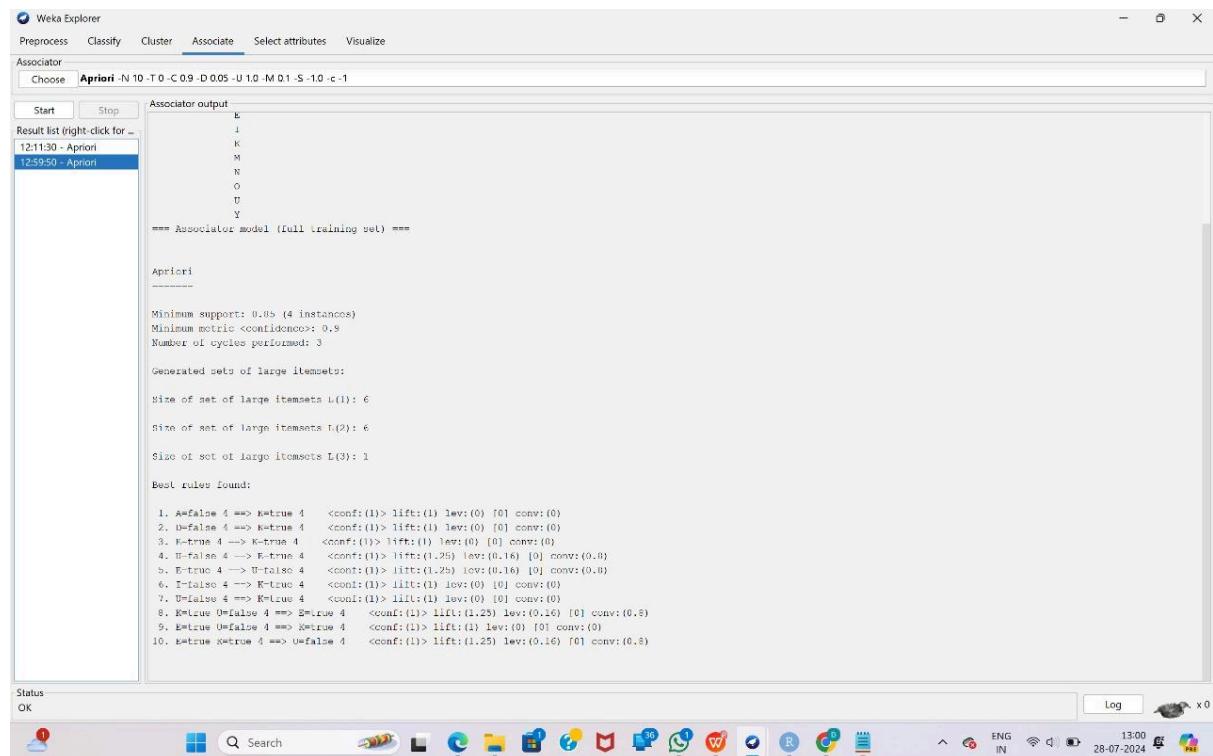
Multiple regression:



Q7:



Apriori algorithm:



The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Result list" dropdown shows "12:11:30 - Apriori" and "12:59:50 - Apriori". The "Associator output" pane displays the following text:

```

Start Stop
Result list (right-click for ...)
12:11:30 - Apriori
12:59:50 - Apriori

Associator output
E
I
K
M
N
O
U
Y
*** Associator model (full training set) ***

Apriori
-----
Minimum support: 0.05 (4 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 3

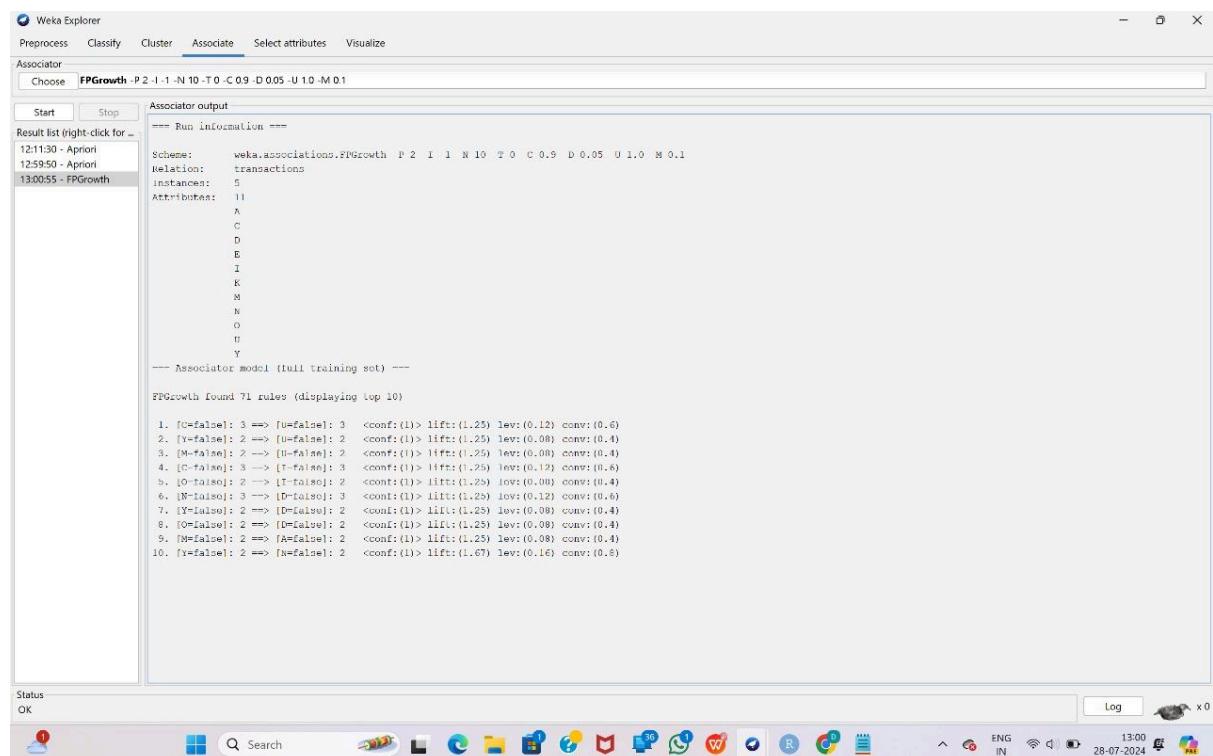
Generated sets of large itemsets:
Size of set of large itemsets L(1): 6
Size of set of large itemsets L(2): 6
Size of set of large itemsets L(3): 1

Best rules found:
1. A=false 4 ==> B=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
2. D=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
3. F=true 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
4. H=false 4 ==> F=true 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
5. F=true 4 ==> U=false 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
6. I=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
7. D=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
8. K=true O=false 4 ==> B=true 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
9. I=false O=false 4 ==> K=true 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
10. I=false K=true 4 ==> O=false 4   <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)

Status OK

```

FP Growth:



The screenshot shows the Weka Explorer interface with the "Associate" tab selected. The "Result list" dropdown shows "12:11:30 - Apriori", "12:59:50 - Apriori", and "13:00:55 - FPGrowth". The "Associator output" pane displays the following text:

```

Start Stop
Result list (right-click for ...)
12:11:30 - Apriori
12:59:50 - Apriori
13:00:55 - FPGrowth

Associator output
--- Run information ---
Scheme: weka.associations.FPGrowth P 2 I 1 N 10 T 0 C 0.9 D 0.05 U 1.0 M 0.1
Relation: transactions
instances: 5
Attributes: 11
A
B
C
D
E
F
I
K
M
N
O
U
Y
--- Associator model (full training set) ---

FPGrowth found 71 rules (displaying top 10)

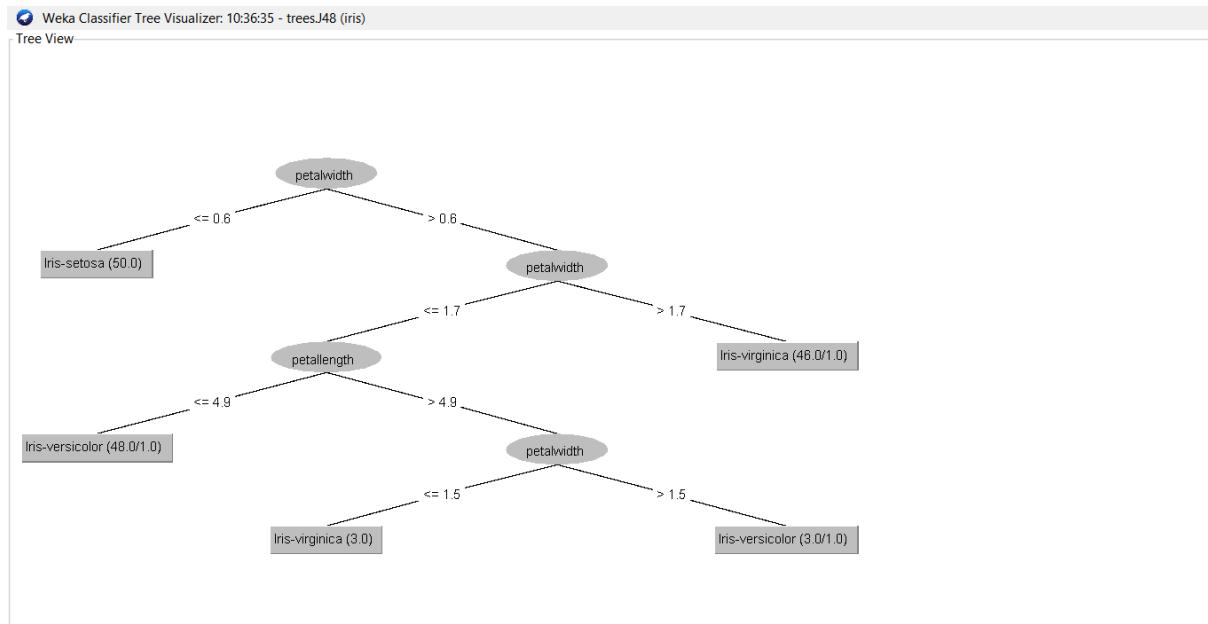
1. [C=false]: 3 ==> [U=false]: 2    <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
2. [Y=false]: 2 ==> [U=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
3. [M=false]: 2 ==> [U=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
4. [C=false]: 3 ==> [T=false]: 3    <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
5. [O=false]: 2 ==> [T=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
6. [N=false]: 3 ==> [D=false]: 3    <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
7. [Y=false]: 2 ==> [D=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
8. [O=false]: 2 ==> [D=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
9. [M=false]: 2 ==> [A=false]: 2    <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
10. [Y=false]: 2 ==> [N=false]: 2   <conf:(1)> lift:(1.67) lev:(0.16) conv:(0.8)

Status OK

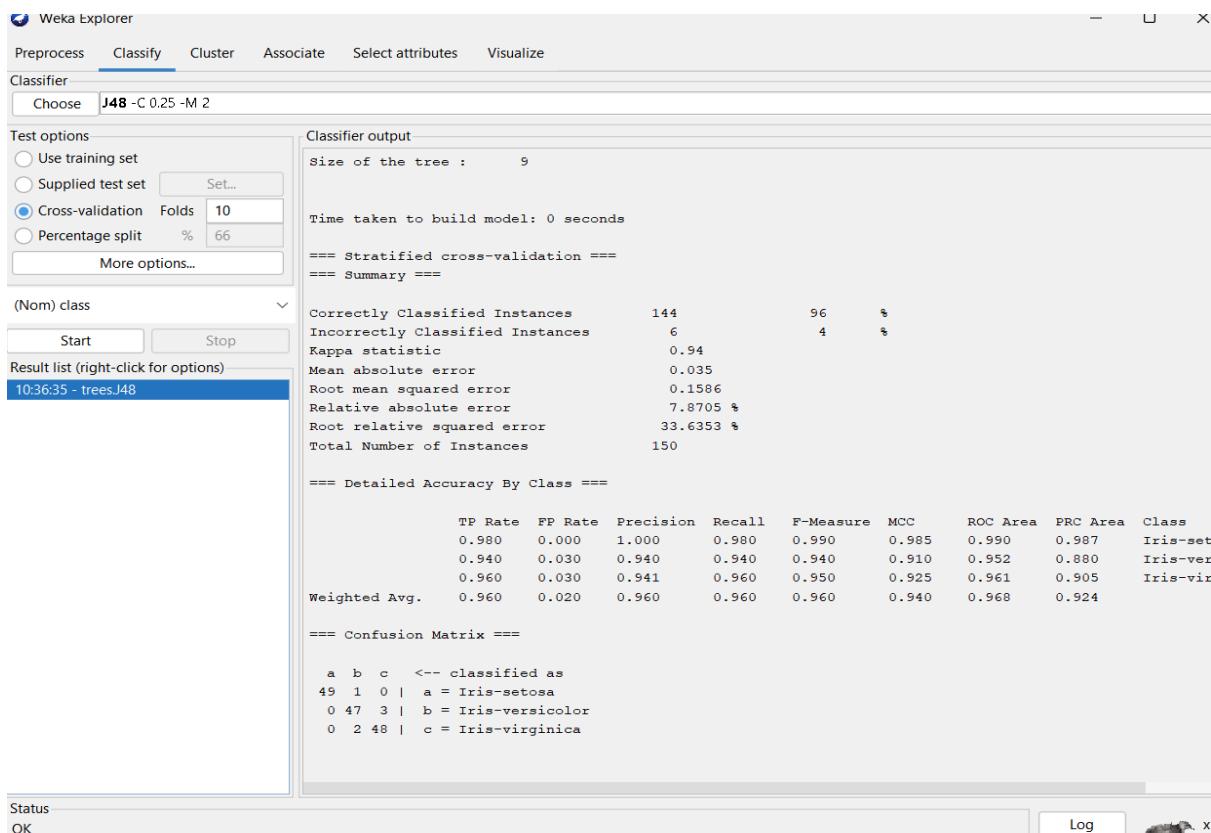
```

Q8:

TREE:



PREPROCESS:



Logistic:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'Logistic' with parameters: -R 1.0E-8 -M -1 -num-decimal-places 4. The test option used is 'Cross-validation' with 10 folds. The classifier output section displays the following details:

- Classifier output:** petalwidth 0 0
- Time taken to build model:** 0.03 seconds
- Stratified cross-validation Summary:**

	Correctly Classified Instances	96 %
Incorrectly Classified Instances	6	4 %
Kappa statistic	0.94	
Mean absolute error	0.0287	
Root mean squared error	0.1424	
Relative absolute error	6.456 %	
Root relative squared error	30.2139 %	
Total Number of Instances	150	
- Detailed Accuracy By Class:**

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	Iris-setosa
0.920	0.020	0.958	0.920	0.939	0.910	0.972	0.934	0.934	Iris-versicolor
0.960	0.040	0.923	0.960	0.941	0.911	0.972	0.934	0.934	Iris-virginica
Weighted Avg.	0.960	0.020	0.960	0.960	0.940	0.981	0.956		
- Confusion Matrix:**

	a	b	c	<-- classified as
50	0	0	1	a = Iris-setosa
0	46	4	0	b = Iris-versicolor
0	2	48	1	c = Iris-virginica

Q9:

INPUT:

#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)

quantile(age,.75)

Output:

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Contains the R code:

```
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)
quantile(age,.75)
```
- Environment View:** Shows the following variables and their values:

Variable	Value
age	num [1:27] 13 15 16 16 19 20 21 22 22 ...
age_values	num [1:27] 13 15 16 16 19 20 21 22 22 ...
decimal_scaling	0
j	FALSE
m	35
max_value	70
max_abs_value	1
mean_value	1000
mean_abs_value	1000
mean_value	500
- Terminal View:** Shows the command history and the output of the R code:

```
[1] > #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)
[2] > 20.5
> quantile(age,.75)
[3] > 35
```

Q10:

INPUT:

```
data(water, package="datasets")

plot(water$hardness, water$mortality, main="Scatterplot of Mortality vs Hardness",
     xlab="Hardness", ylab="Mortality")

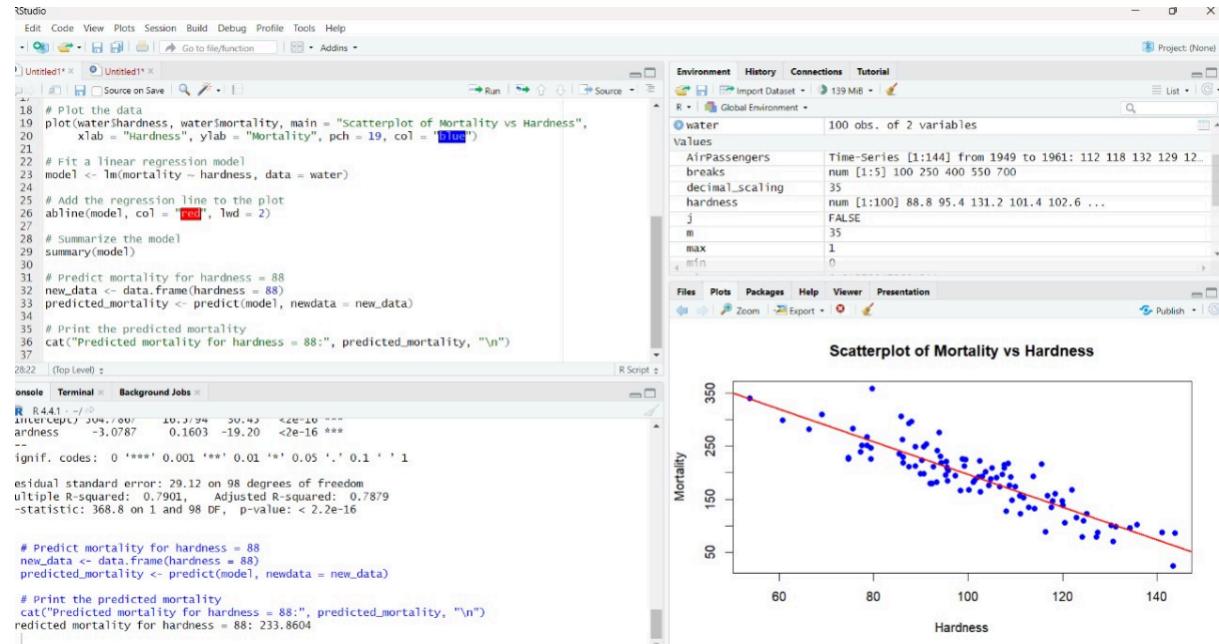
model <- lm(mortality ~ hardness, data=water)

abline(model, col="red")

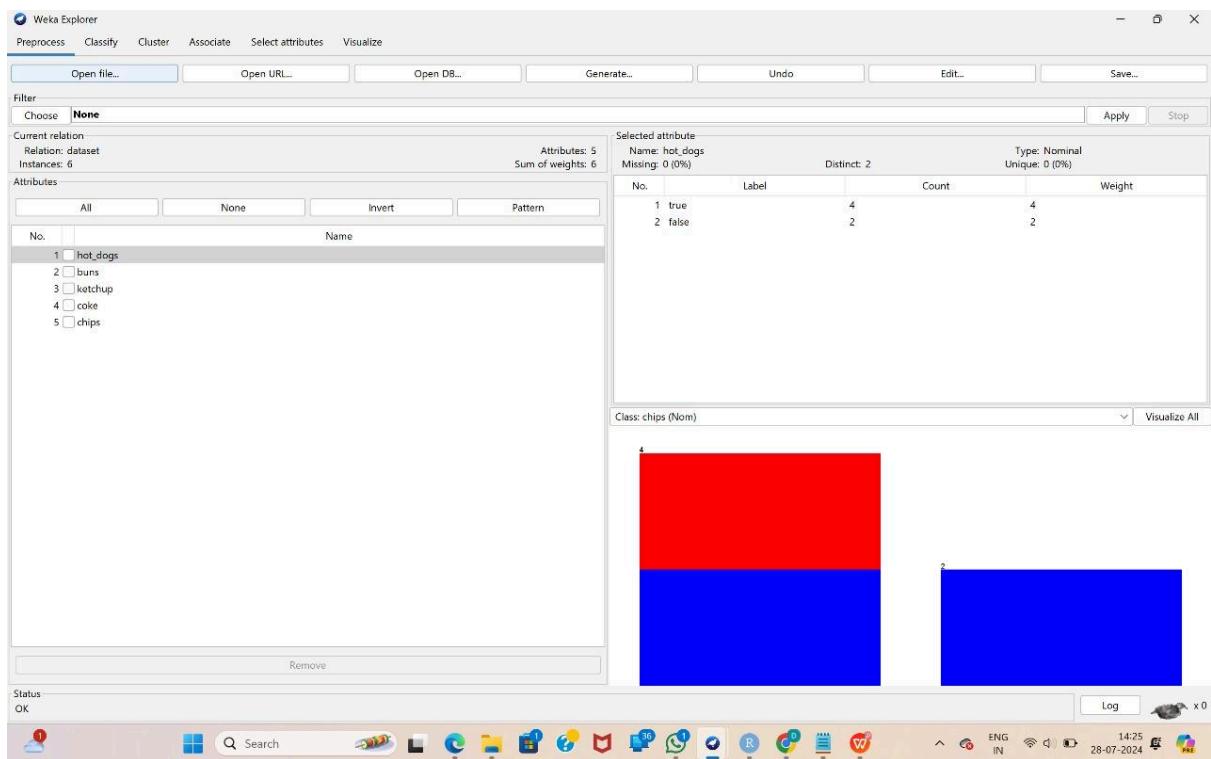
predicted_mortality <- predict(model, newdata=data.frame(hardness=88))

predicted_mortality
```

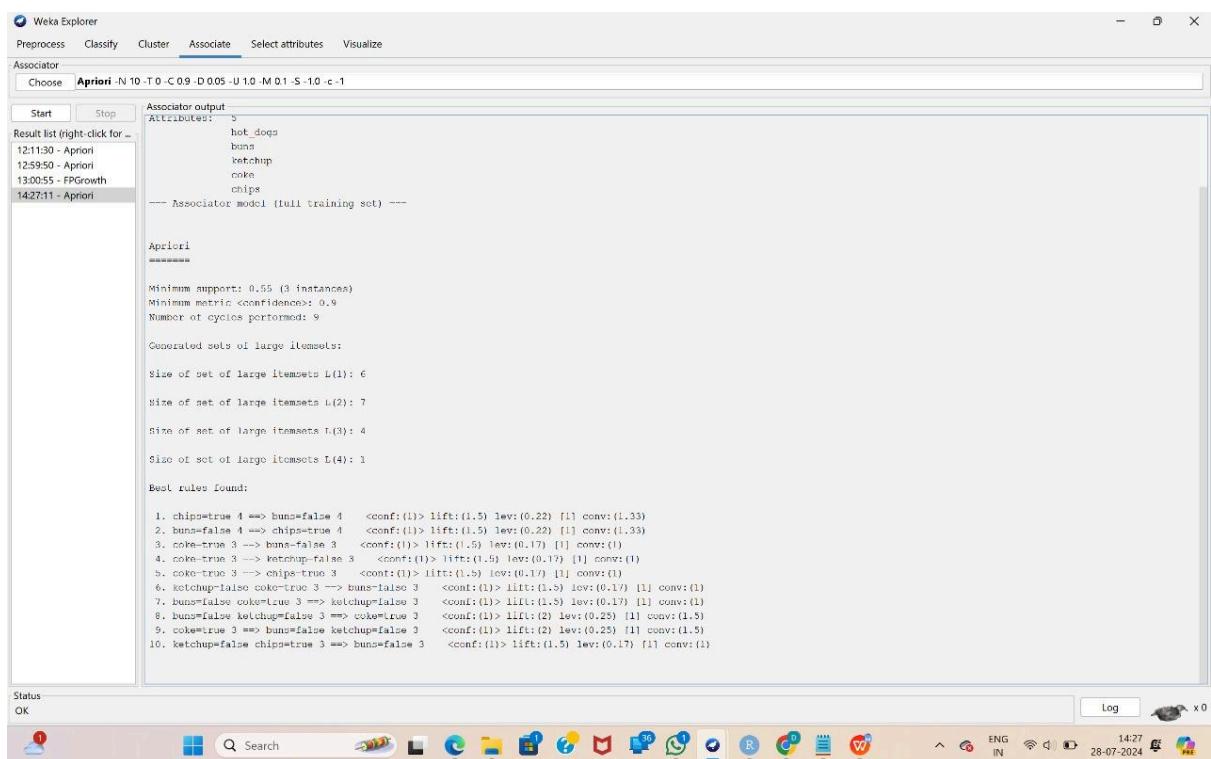
OUTPUT:



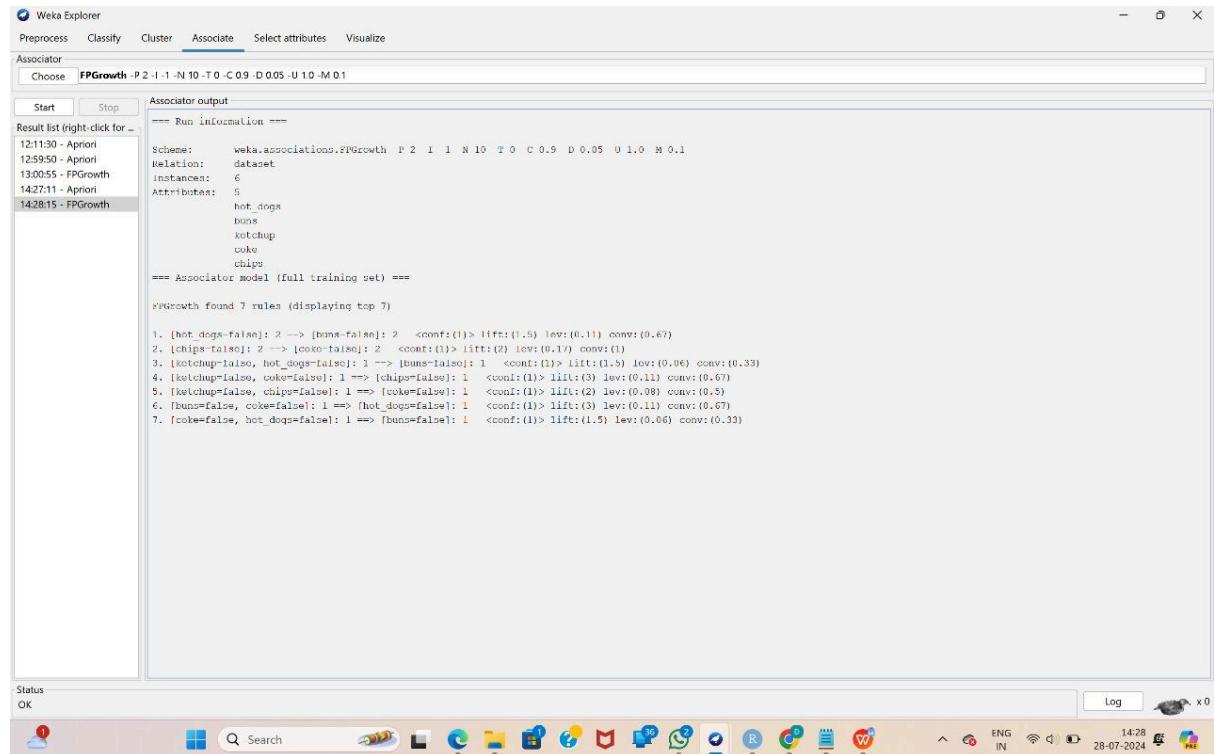
Q11:



Apriori Algorithm:



FP Growth:



The screenshot shows the Weka Explorer interface with the 'Associate' tab selected. The 'Associate output' pane displays the following information:

```

Result list (right-click for ...)
12:11:30 - Apriori
12:59:50 - Apriori
13:00:55 - FPGrowth
14:27:11 - Apriori
14:28:15 - FPGrowth

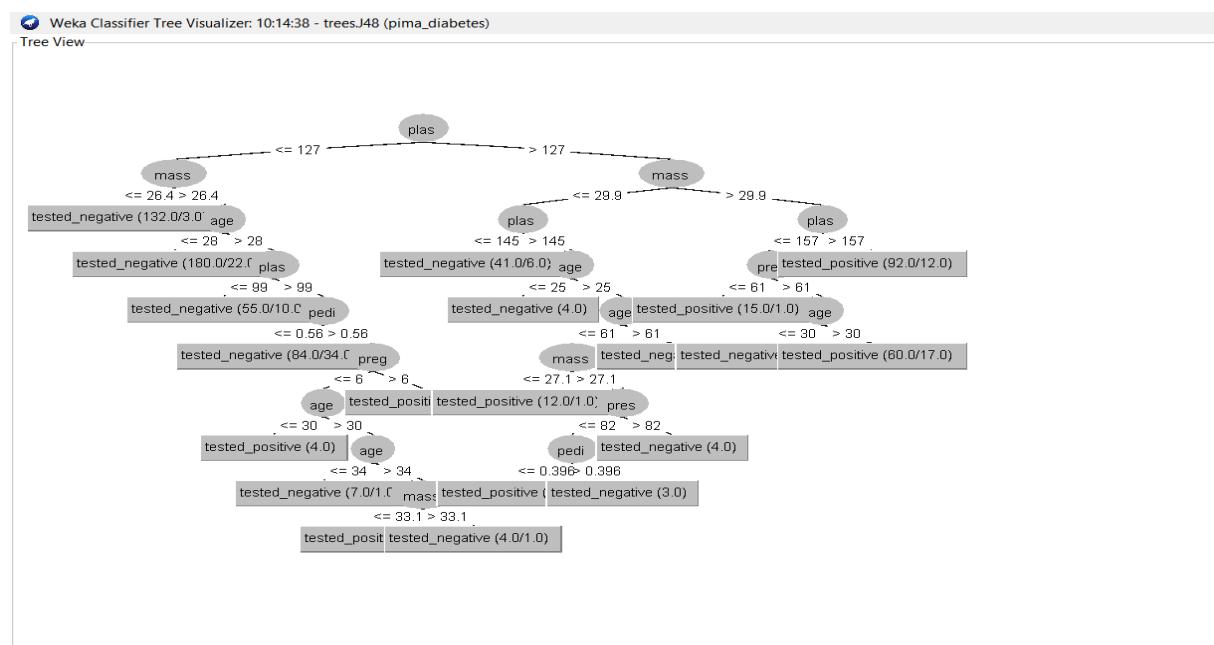
Associator output
Start Stop
==== Run information ====
Scheme: weka.associations.FPGrowth P 2 I 1 N 10 T 0 C 0.9 D 0.005 U 1.0 M 0.1
Relation: dataset
Instances: 6
Attributes: 5
Attributes:
    hot_dogs
    buns
    ketchup
    coke
    chips
==== Associator model (full training set) ====
FPGrowth found 7 rules (displaying top 7)

1. [hot_dogs=false]: 2 --> [buns=false]: 2 <conf:(1)> lift:(1.5) lev:(0.11) conv:(0.67)
2. [chips=false]: 2 --> [coke=false]: 2 <conf:(1)> lift:(2) lev:(0.17) conv:(1)
3. [ketchup=false, hot_dogs=false]: 1 --> [buns=false]: 1 <conf:(1)> lift:(1.5) lev:(0.06) conv:(0.33)
4. [ketchup=false, coke=false]: 1 --> [chips=false]: 1 <conf:(1)> lift:(3) lev:(0.11) conv:(0.67)
5. [ketchup=false, chips=false]: 1 --> [coke=false]: 1 <conf:(1)> lift:(2) lev:(0.08) conv:(0.5)
6. [buns=false, coke=false]: 1 --> [hot_dogs=false]: 1 <conf:(1)> lift:(3) lev:(0.11) conv:(0.67)
7. [coke=false, hot_dogs=false]: 1 --> [buns=false]: 1 <conf:(1)> lift:(1.5) lev:(0.06) conv:(0.33)

```

Q12:

Tree:



Decision Tree:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. A classifier named 'DecisionTable-X 1-5 "weka.attributeSelection.BestFirst-D 1-N 5"' is chosen. The 'Test options' section shows 'Cross-validation Folds 10' selected. The 'Classifier output' pane displays the results of a 10-fold cross-validation on the Iris dataset. It includes statistics like correctly classified instances (139), incorrectly classified instances (11), Kappa statistic (0.89), and various error metrics. The detailed accuracy by class table shows high performance across all classes. A confusion matrix is also provided.

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	Iris-setosa
0.880	0.050	0.898	0.880	0.889	0.834	0.946	0.861	Iris-versicolor
0.900	0.060	0.882	0.900	0.891	0.836	0.947	0.869	Iris-virginica
Weighted Avg.	0.927	0.037	0.927	0.927	0.890	0.964	0.910	

Logistic:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. A classifier named 'Logistic-R 1.0E-8 -M 1 -num-decimal-places 4' is chosen. The 'Test options' section shows 'Cross-validation Folds 10' selected. The 'Classifier output' pane displays the results of a 10-fold cross-validation on the CSIRO Land Surface dataset. It includes statistics like correctly classified instances (2948), incorrectly classified instances (1679), Kappa statistic (0), and various error metrics. The detailed accuracy by class table shows high performance across all classes. A confusion matrix is also provided.

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class	
1.000	1.000	0.637	1.000	0.778	?	0.499	0.637	low	
0.000	0.000	?	0.000	?	?	0.499	0.362	high	
Weighted Avg.	0.637	0.637	?	0.637	?	?	0.499	0.537	

Q13:

INPUT:

```
prices <- c(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, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30)
```

```
library(dplyr)

num_bins <- 3

bin_size <- length(prices) / num_bins

bins <- split(prices, ceiling(seq_along(prices) / bin_size))

print("Equal-frequency bins:")

print(bins)

bin_means <- lapply(bins, mean)

print("Smoothing by bin means:")

print(bin_means)

bin_boundaries <- lapply(bins, function(bin) {

  min_val <- min(bin)

  max_val <- max(bin)

  sapply(bin, function(x) {

    if (x - min_val < max_val - x) min_val else max_val

  })

})

print("Smoothing by bin boundaries:")

print(bin_boundaries)

bins_unlisted <- unlist(bins)
```

```

hist(bins_unlisted, breaks=num_bins, main="Histogram of Prices with
Equal-Frequency Bins", xlab="Price", ylab="Frequency", col="lightblue",
border="black")

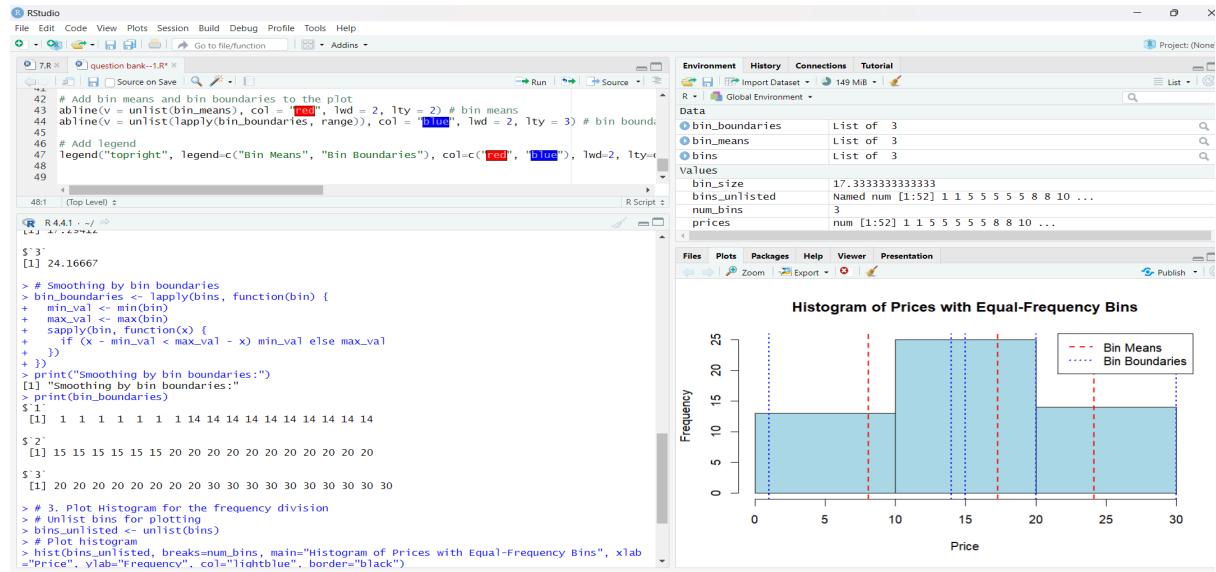
abline(v = unlist(bin_means), col = "red", lwd = 2, lty = 2) # bin means

abline(v = unlist(lapply(bin_boundaries, range)), col = "blue", lwd = 2, lty = 3) # bin
boundaries

legend("topright", legend=c("Bin Means", "Bin Boundaries"), col=c("red", "blue"),
lwd=2, lty=c(2, 3))

```

OUTPUT:



Q14:

INPUT:

```
class_A <- c(76, 35, 47, 64, 95, 66, 89, 36, 84)
```

```
class_B <- c(51, 56, 84, 60, 59, 70, 63, 66, 50)
```

```
mean_A <- mean(class_A)
```

```
mean_B <- mean(class_B)
```

```
median_A <- median(class_A)
```

```
median_B <- median(class_B)
```

```
range_A <- range(class_A)
```

```
range_B <- range(class_B)
```

```
range_A_diff <- diff(range_A)
```

```
range_B_diff <- diff(range_B)
```

```
cat("Class A - Mean:", mean_A, " Median:", median_A, " Range:", range_A_diff,  
"\n")
```

```
cat("Class B - Mean:", mean_B, " Median:", median_B, " Range:", range_B_diff,  
"\n")
```

```
boxplot(class_A, class_B, names=c("Class A", "Class B"), main="Comparison of Class  
A and Class B Performance", ylab="Scores", col=c("lightblue", "lightgreen"))
```

```
abline(h=mean_A, col="blue", lty=2)
```

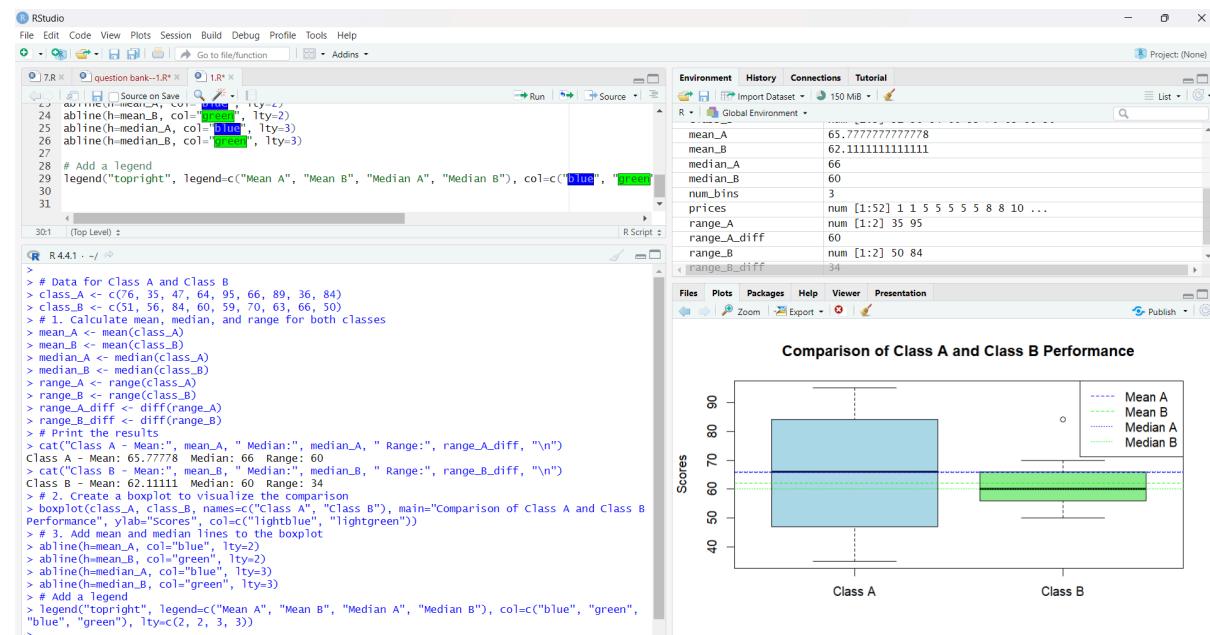
```
abline(h=mean_B, col="green", lty=2)
```

```
abline(h=median_A, col="blue", lty=3)
```

```
abline(h=median_B, col="green", lty=3)
```

```
legend("topright", legend=c("Mean A", "Mean B", "Median A", "Median B"),  
col=c("blue", "green", "blue", "green"), lty=c(2, 2, 3, 3))
```

OUTPUT:



Q16:

```
library(cluster)    # For clustering algorithms  
library(factoextra) # For clustering visualization  
library(datasets)   # For sample datasets  
  
set.seed(123)  
  
customer_data <- data.frame(  
  percentage_opened = runif(100, 0, 100), # Percentage of emails opened  
  clicks_per_email = runif(100, 0, 10),  # Number of clicks per email  
  time_spent_viewing = runif(100, 0, 5)  # Time spent viewing email (in minutes)  
)  
  
customer_data_scaled <- scale(customer_data)  
  
set.seed(123)  
  
kmeans_result <- kmeans(customer_data_scaled, centers = 3, nstart = 25)  
  
distances <- dist(customer_data_scaled)  
  
hclust_result <- hclust(distances, method = "ward.D2")  
  
cutree_result <- cutree(hclust_result, k = 3)  
  
silhouette_kmeans <- silhouette(kmeans_result$cluster, distances)  
  
silhouette_hclust <- silhouette(cutree_result, distances)  
  
avg_silhouette_kmeans <- mean(silhouette_kmeans[, 3])  
  
avg_silhouette_hclust <- mean(silhouette_hclust[, 3])  
  
cat("Average Silhouette Score for k-means: ", avg_silhouette_kmeans, "\n")  
  
cat("Average Silhouette Score for Hierarchical Clustering: ", avg_silhouette_hclust,  
"\n")  
  
fviz_cluster(kmeans_result, data = customer_data_scaled, geom = "point", stand =  
FALSE, main = "K-means Clustering")
```

```

fviz_dend(hclust_result, k = 3, rect = TRUE, main = "Hierarchical Clustering
Dendrogram")

if (avg_silhouette_kmeans > avg_silhouette_hclust) {

  cat("K-means clustering performed better based on the silhouette score.\n")

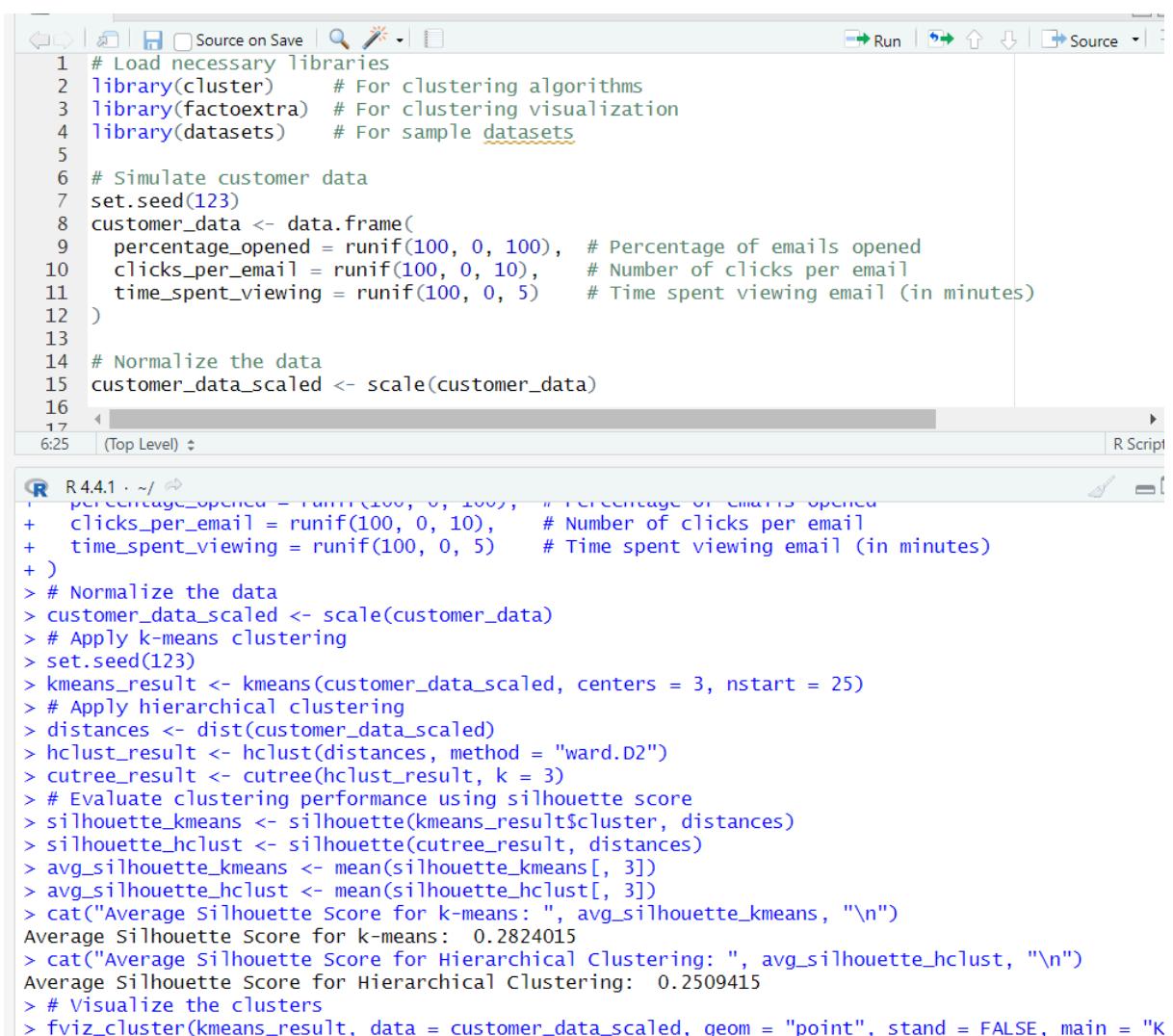
} else {

  cat("Hierarchical clustering performed better based on the silhouette score.\n")

}

```

OUTPUT:



The screenshot shows the RStudio interface with an R script editor and a console window.

R Script Content:

```

1 # Load necessary libraries
2 library(cluster)      # For clustering algorithms
3 library(factoextra)    # For clustering visualization
4 library(datasets)      # For sample datasets
5
6 # Simulate customer data
7 set.seed(123)
8 customer_data <- data.frame(
9   percentage_opened = runif(100, 0, 100), # Percentage of emails opened
10  clicks_per_email = runif(100, 0, 10),   # Number of clicks per email
11  time_spent_viewing = runif(100, 0, 5)    # Time spent viewing email (in minutes)
12 )
13
14 # Normalize the data
15 customer_data_scaled <- scale(customer_data)
16
17
6:25 (Top Level) ◊

```

Console Output:

```

R 4.4.1 · ~/ ◊
percentage_opened = runif(100, 0, 100), "Percentage of emails opened"
+ clicks_per_email = runif(100, 0, 10), # Number of clicks per email
+ time_spent_viewing = runif(100, 0, 5) # Time spent viewing email (in minutes)
+
> # Normalize the data
> customer_data_scaled <- scale(customer_data)
> # Apply k-means clustering
> set.seed(123)
> kmeans_result <- kmeans(customer_data_scaled, centers = 3, nstart = 25)
> # Apply hierarchical clustering
> distances <- dist(customer_data_scaled)
> hclust_result <- hclust(distances, method = "ward.D2")
> cutree_result <- cutree(hclust_result, k = 3)
> # Evaluate clustering performance using silhouette score
> silhouette_kmeans <- silhouette(kmeans_result$cluster, distances)
> silhouette_hclust <- silhouette(cutree_result, distances)
> avg_silhouette_kmeans <- mean(silhouette_kmeans[, 3])
> avg_silhouette_hclust <- mean(silhouette_hclust[, 3])
> cat("Average Silhouette Score for k-means: ", avg_silhouette_kmeans, "\n")
Average Silhouette Score for k-means: 0.2824015
> cat("Average Silhouette Score for Hierarchical Clustering: ", avg_silhouette_hclust, "\n")
Average Silhouette Score for Hierarchical Clustering: 0.2509415
> # Visualize the clusters
> fviz_cluster(kmeans_result, data = customer_data_scaled, geom = "point", stand = FALSE, main = "K

```

Q17:

INPUT:

```
age<-c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)

fat<-c(9.5,26.5,7.8,17.8,31.4,25.9,27.4,27.2,31.2,34.6,42.5,28.8,33.4,30.2,34.1,32.9,41.2,35
.7)

mean(age)

median(age)

sd(age)

mean(fat)

median(fat)

sd(fat)

#boxplot

boxplot(age,fat)

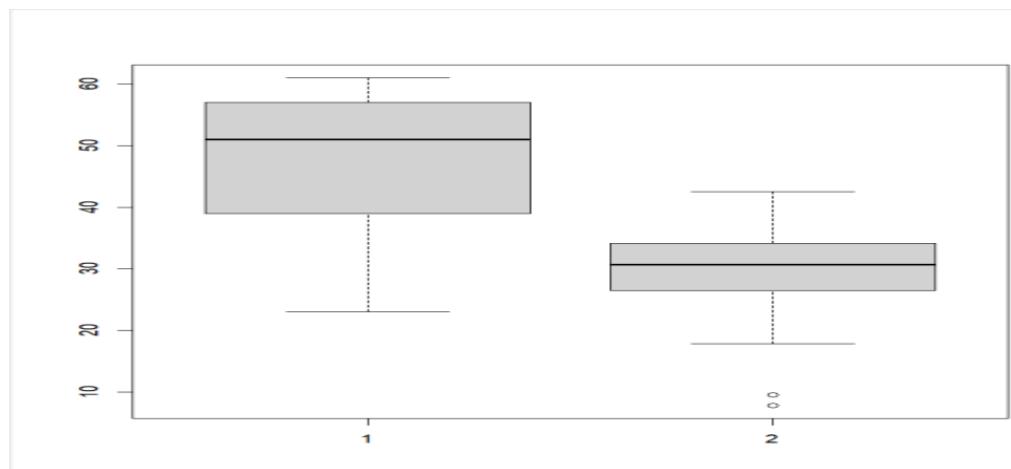
#scatter plot

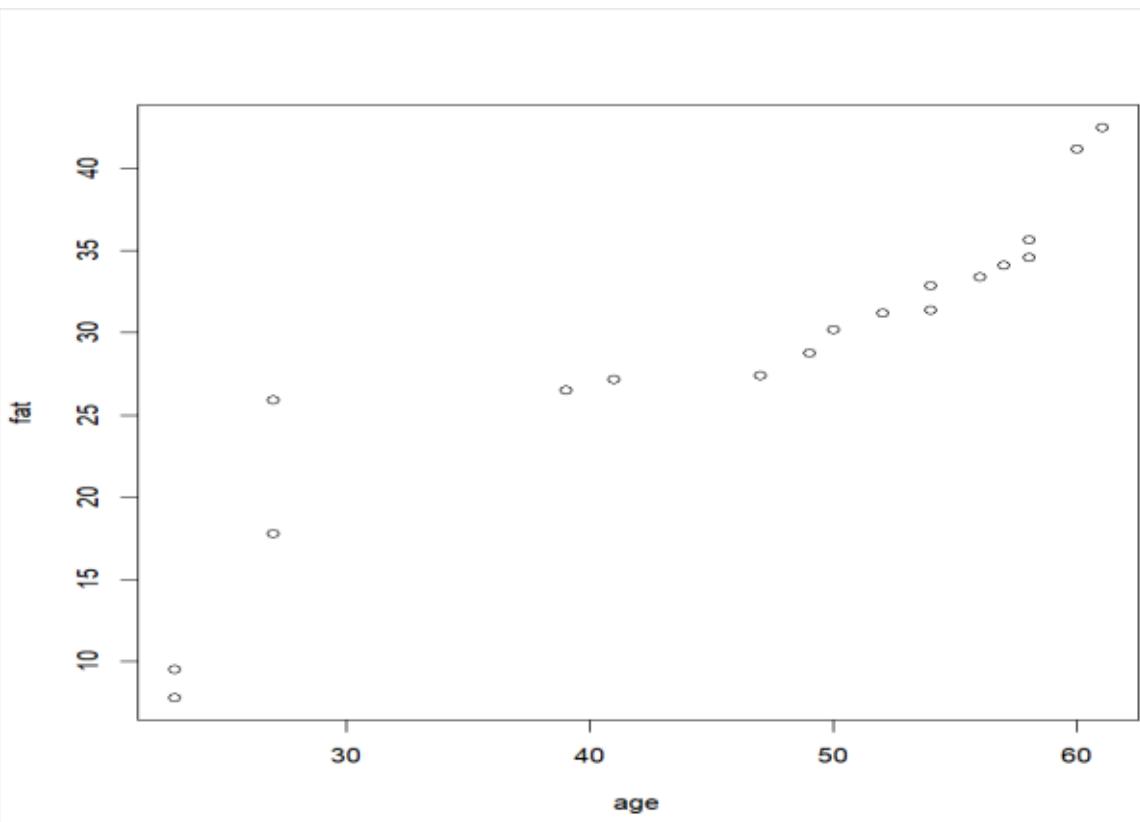
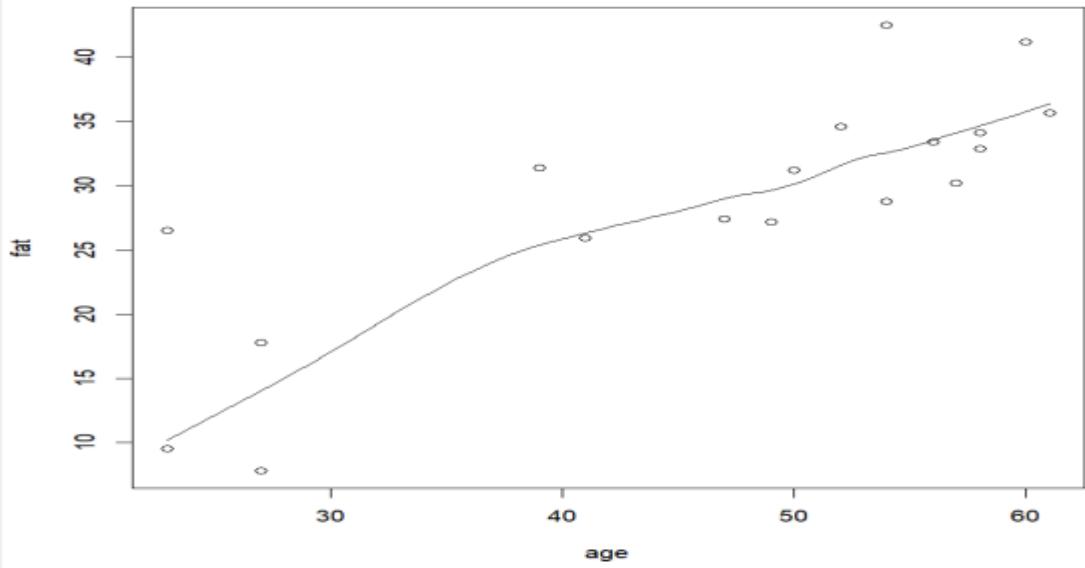
scatter.smooth(age,fat)

#qplot

qqplot(age,fat)
```

Output:





Q18:

INPUT:

```
v<-c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
```

```
min<-0
```

```
max<-1
```

```
#min_max
```

```
min_max=((35-min(v))/(max(v)-min(v)))
```

```
print(min_max)
```

```
#z-score
```

```
m=mean(v)
```

```
s<-12.94
```

```
z_score=(35-m)/s
```

```
print(z_score)
```

```
#decimal scaling
```

```
m<-35
```

```
j=max(m)<1
```

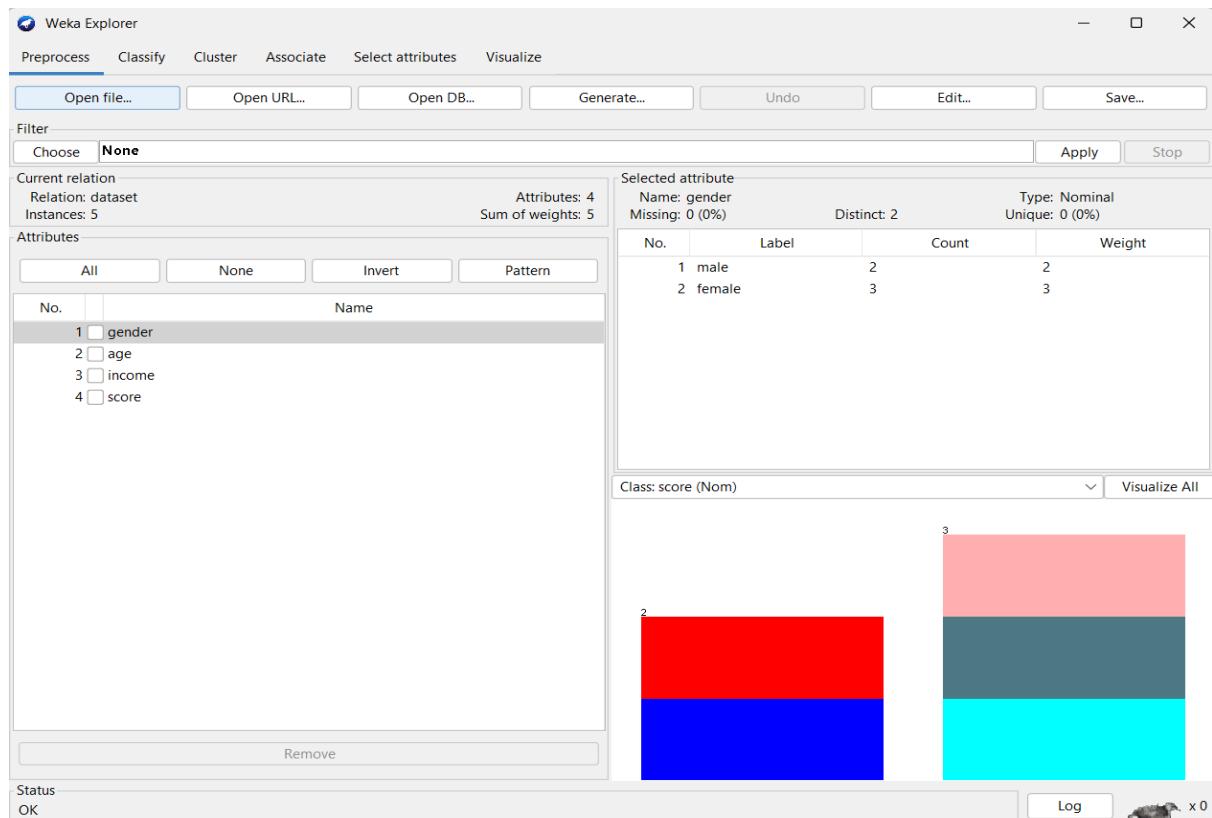
```
decimal_scaling=m/10^j
```

```
print(decimal_scaling)
```

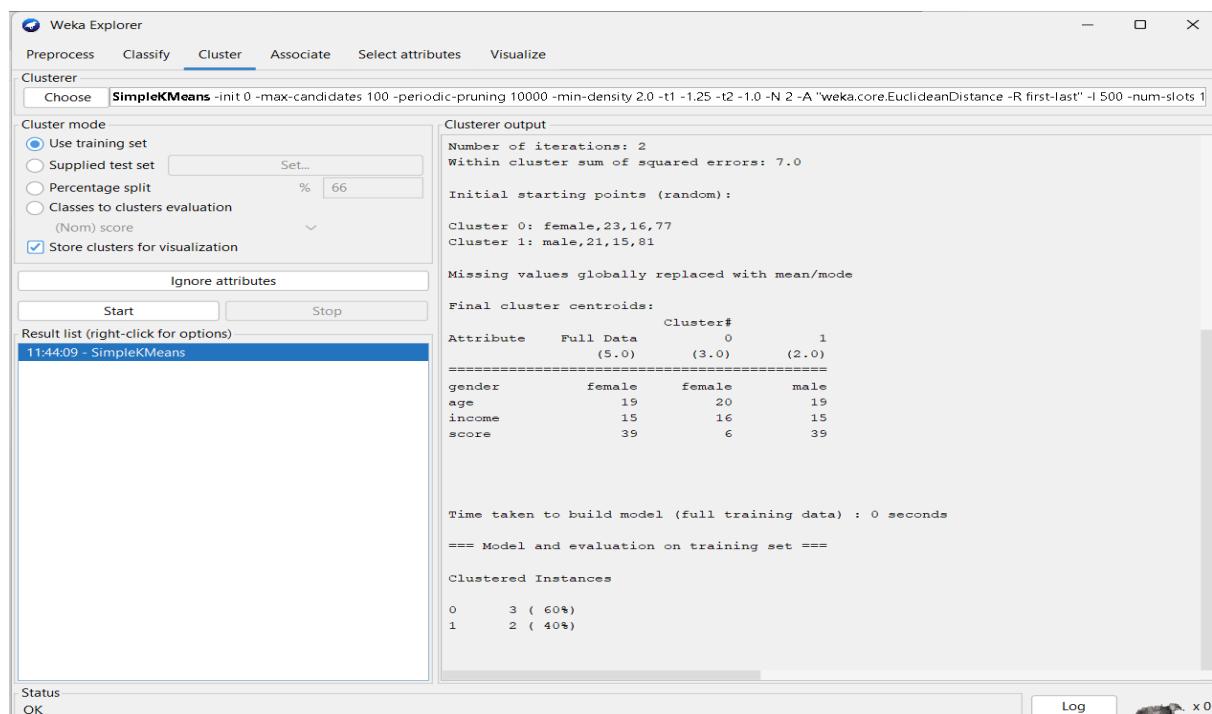
Output:

```
print(min_max)
[1] 0.3157895
#z-score
m=mean(v)
s<-12.94
z_score=(35-m)/s
print(z_score)
[1] -0.8844238
#decimal scaling
m<-35
j=max(m)<1
decimal_scaling=m/10^j
print(decimal_scaling)
[1] 35
```

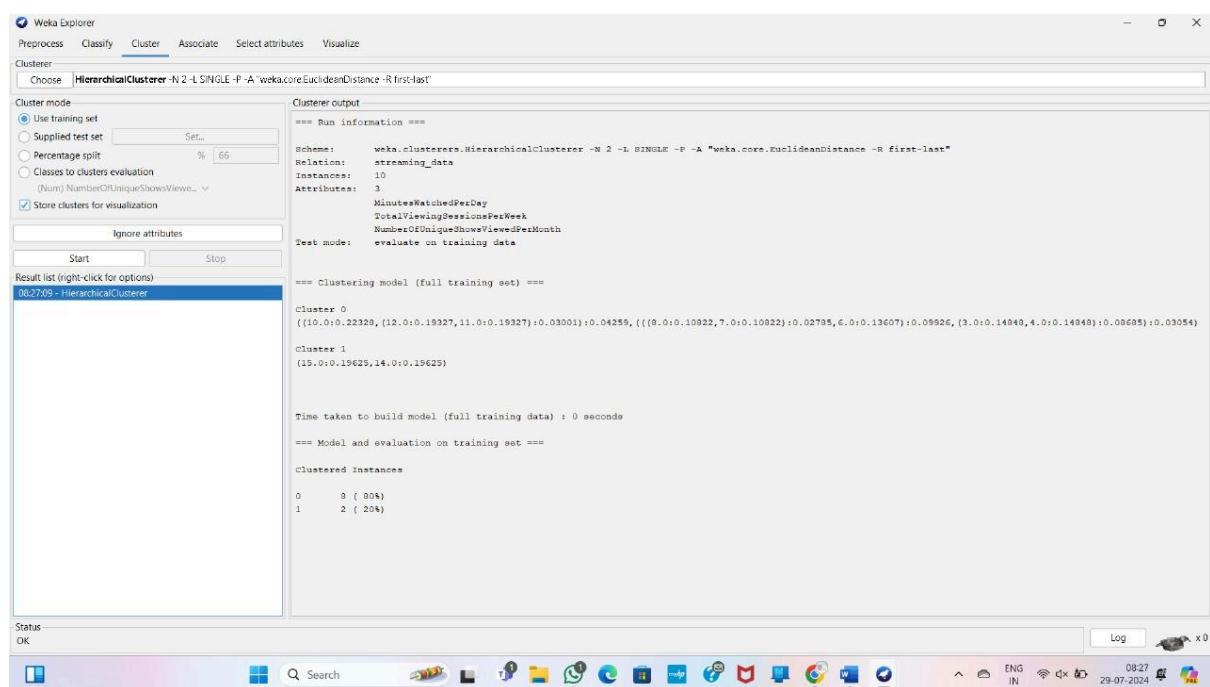
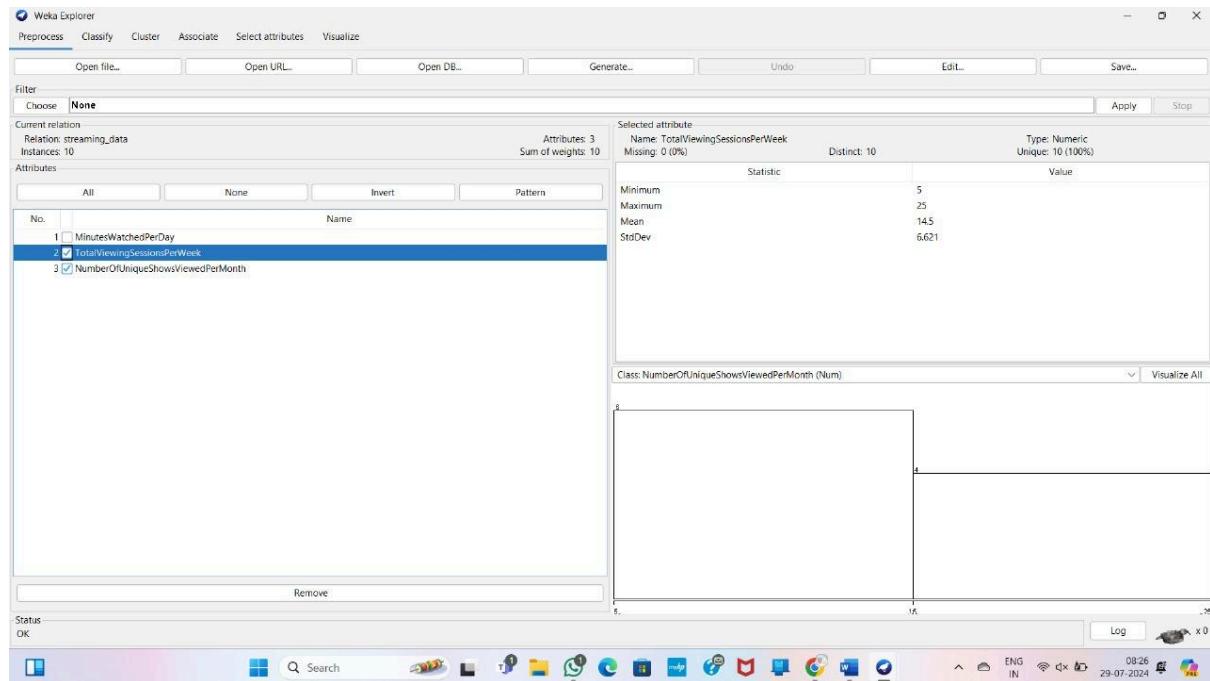
Q19:



Clustering Analysis:



Q20:



Q21:

INPUT:

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

output:

The screenshot shows the RStudio interface. The top panel displays the R script code for Q21. The bottom panel shows the R console output where the code is run and the results are displayed.

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

```
R 4.4.1 · ~/ 
> pencils<-c(25,23,12,11,6,7,8,9,10)
> mean(pencils)
[1] 12.33333
> median(pencils)
[1] 10
> mode=names(table(pencils))[table(pencils)==max(table(pencils))]
>
> |
```

Q22:

INPUT:

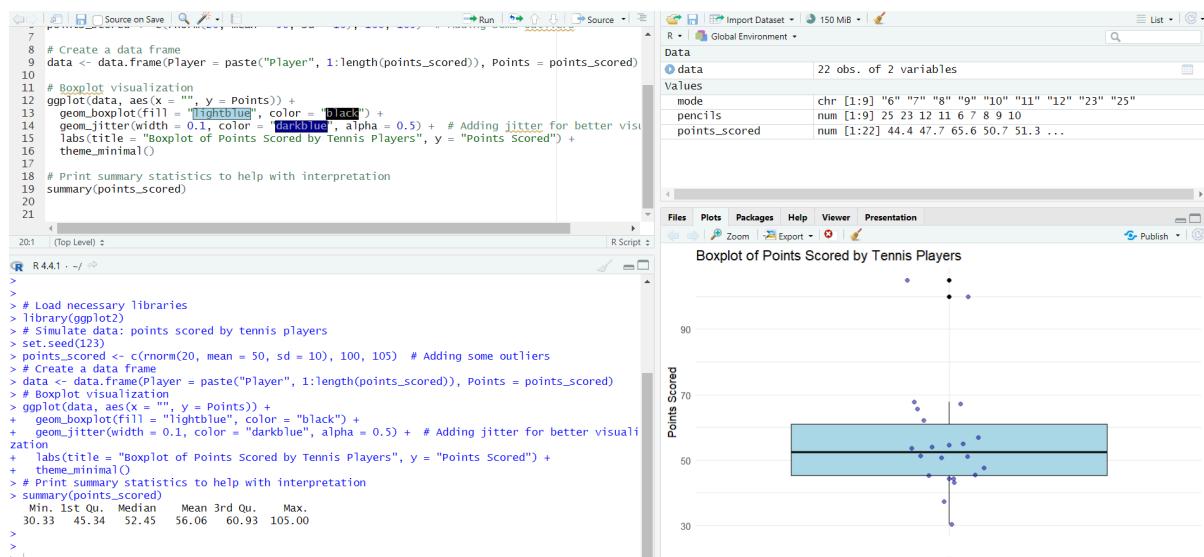
```
library(ggplot2)

set.seed(123)

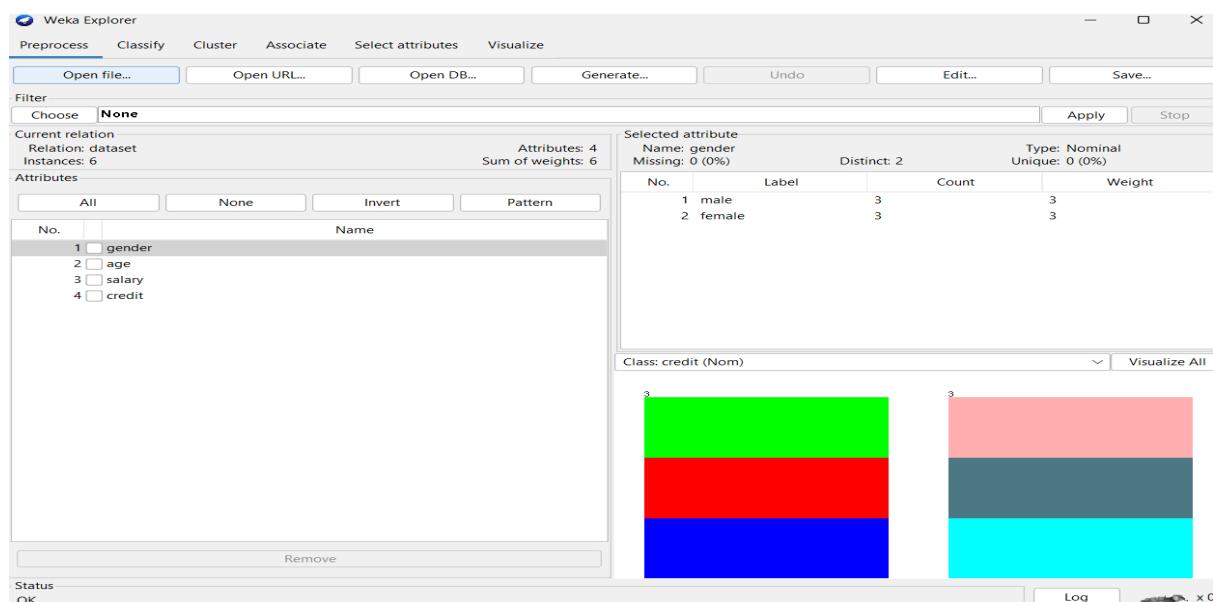
points_scored <- c(rnorm(20, mean = 50, sd = 10), 100, 105) # Adding some outliers

data <- data.frame(Player = paste("Player", 1:length(points_scored)), Points =
points_scored)
ggplot(data, aes(x = "", y = Points)) +
  geom_boxplot(fill = "lightblue", color = "black") +
  geom_jitter(width = 0.1, color = "darkblue", alpha = 0.5) + # Adding jitter for better
  visualization
  labs(title = "Boxplot of Points Scored by Tennis Players", y = "Points Scored") +
  theme_minimal()
summary(points_scored)
```

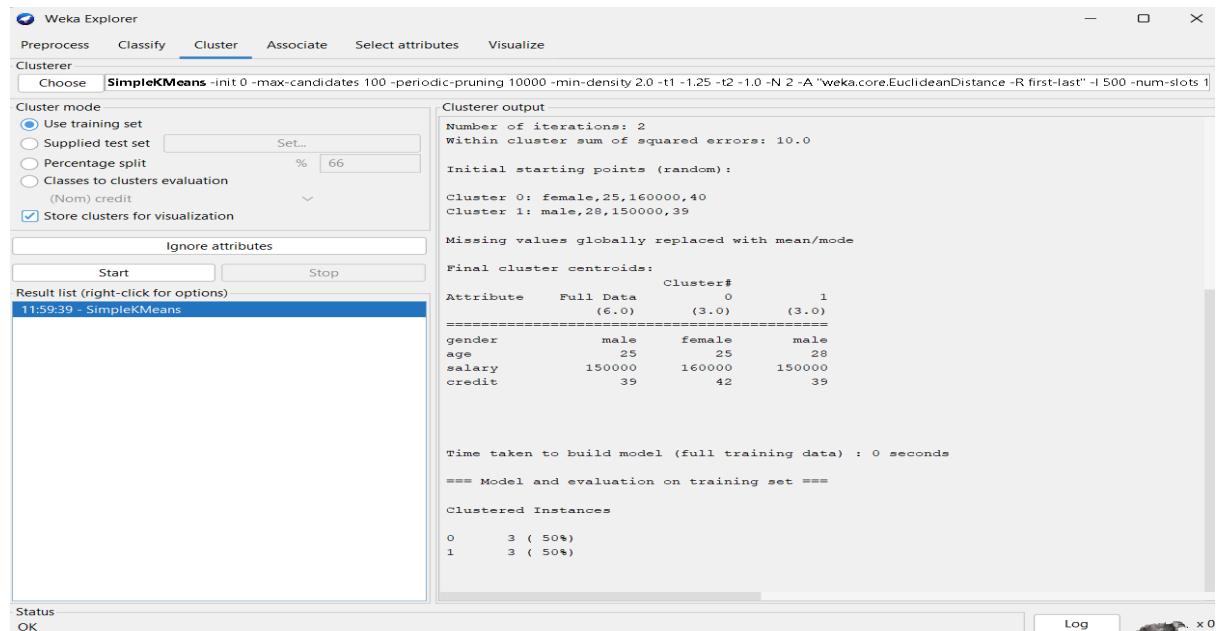
Output:



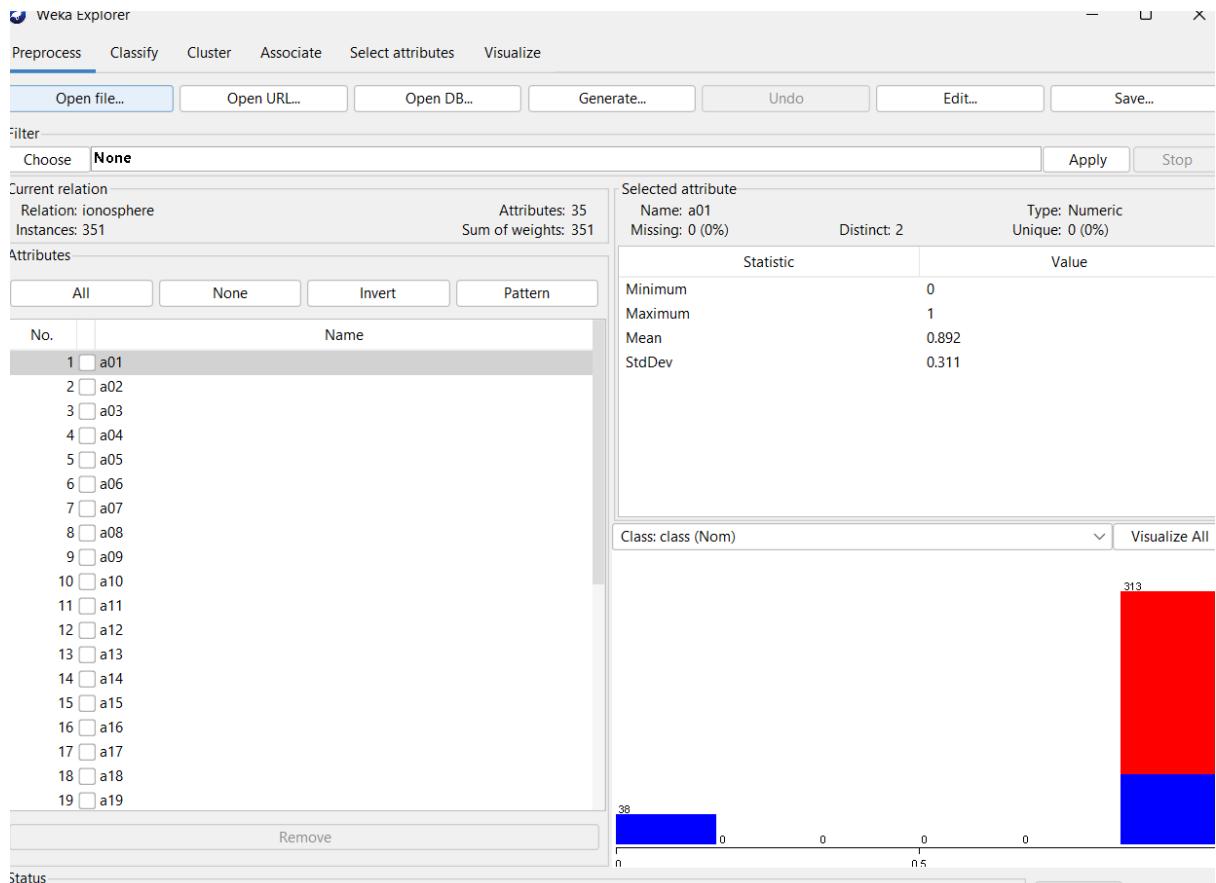
Q23:



K Means:



Q24:



Naïve Bayes classification:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. Under 'Classifier', 'Logistic' is chosen with options: -R 1.0E-8 -M -1 -num-decimal-places 4. In the 'Test options' section, 'Cross-validation' is selected with 10 folds. The 'Result list' pane shows '18:50:13 - bayes.NaiveBayes' and '18:50:26 - functions.Logistic'. The main output pane displays:

```

Classifier output
std. dev.      0.5633  0.4027
weight sum      126     225
precision       0.0076  0.0076

Time taken to build model: 0.04 seconds

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

Correctly Classified Instances      290      82.6211 %
Incorrectly Classified Instances   61      17.3789 %
Kappa statistic                   0.6394
Mean absolute error               0.1736
Root mean squared error          0.3935
Relative absolute error           37.7001 %
Root relative squared error      82.0203 %
Total Number of Instances        351

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

      TP Rate   FP Rate   Precision   Recall   F-Measure   MCC   ROC Area   PRC Area   Class
      0.865    0.196    0.712      0.865    0.781      0.648   0.935    0.917      b
      0.804    0.135    0.914      0.804    0.856      0.648   0.935    0.958      g
Weighted Avg.      0.826    0.157    0.842      0.826    0.829      0.648   0.935    0.943

==== Confusion Matrix ====

      a      b  <-- classified as
109    17 |  a = b
 44   181 |  b = g

```

SVM:

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. Under 'Classifier', 'Logistic' is chosen with options: -R 1.0E-8 -M -1 -num-decimal-places 4. In the 'Test options' section, 'Cross-validation' is selected with 10 folds. The 'Result list' pane shows '18:50:13 - bayes.NaiveBayes' and '18:50:26 - functions.Logistic'. The main output pane displays:

```

Classifier output
a31      0.2054
a32      0.5627
a33      0.8734
a34      56.1574

Time taken to build model: 0.13 seconds

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

Correctly Classified Instances      312      88.8889 %
Incorrectly Classified Instances   39      11.1111 %
Kappa statistic                   0.753
Mean absolute error               0.1283
Root mean squared error          0.3035
Relative absolute error           27.8593 %
Root relative squared error      63.2601 %
Total Number of Instances        351

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

      TP Rate   FP Rate   Precision   Recall   F-Measure   MCC   ROC Area   PRC Area   Class
      0.794    0.058    0.885      0.794    0.837      0.756   0.870    0.896      b
      0.942    0.206    0.891      0.942    0.916      0.756   0.870    0.832      g
Weighted Avg.      0.889    0.153    0.889      0.889    0.887      0.756   0.870    0.855

==== Confusion Matrix ====

      a      b  <-- classified as
100    26 |  a = b
 13   212 |  b = g

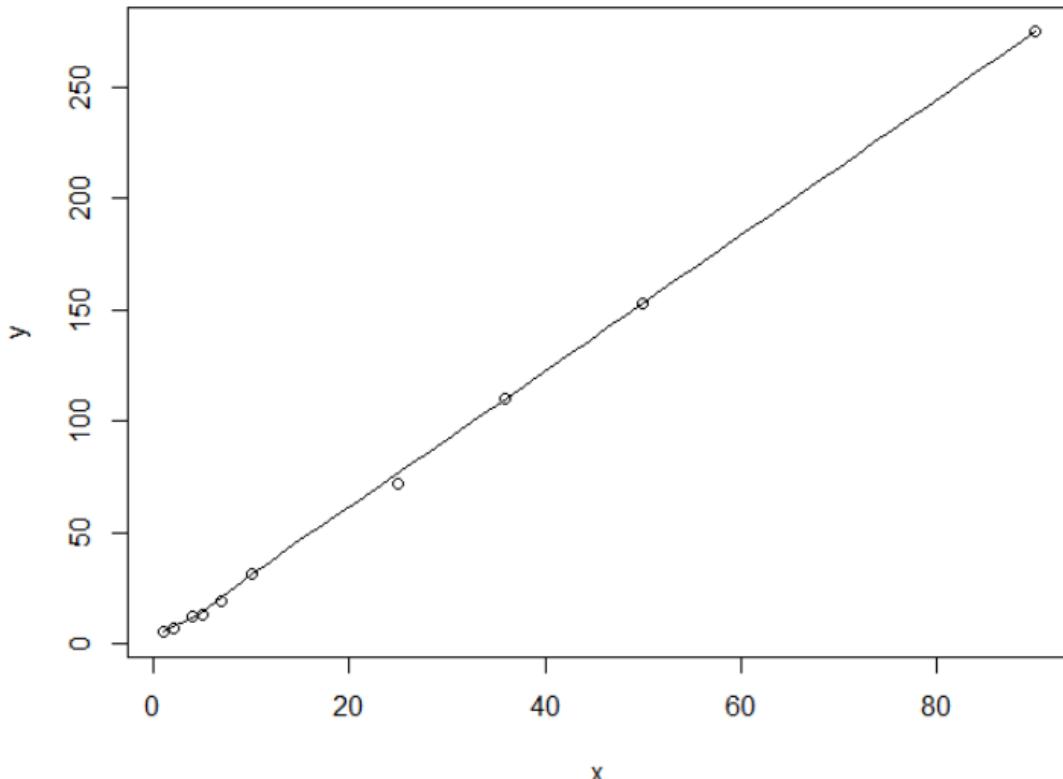
```

Q25:

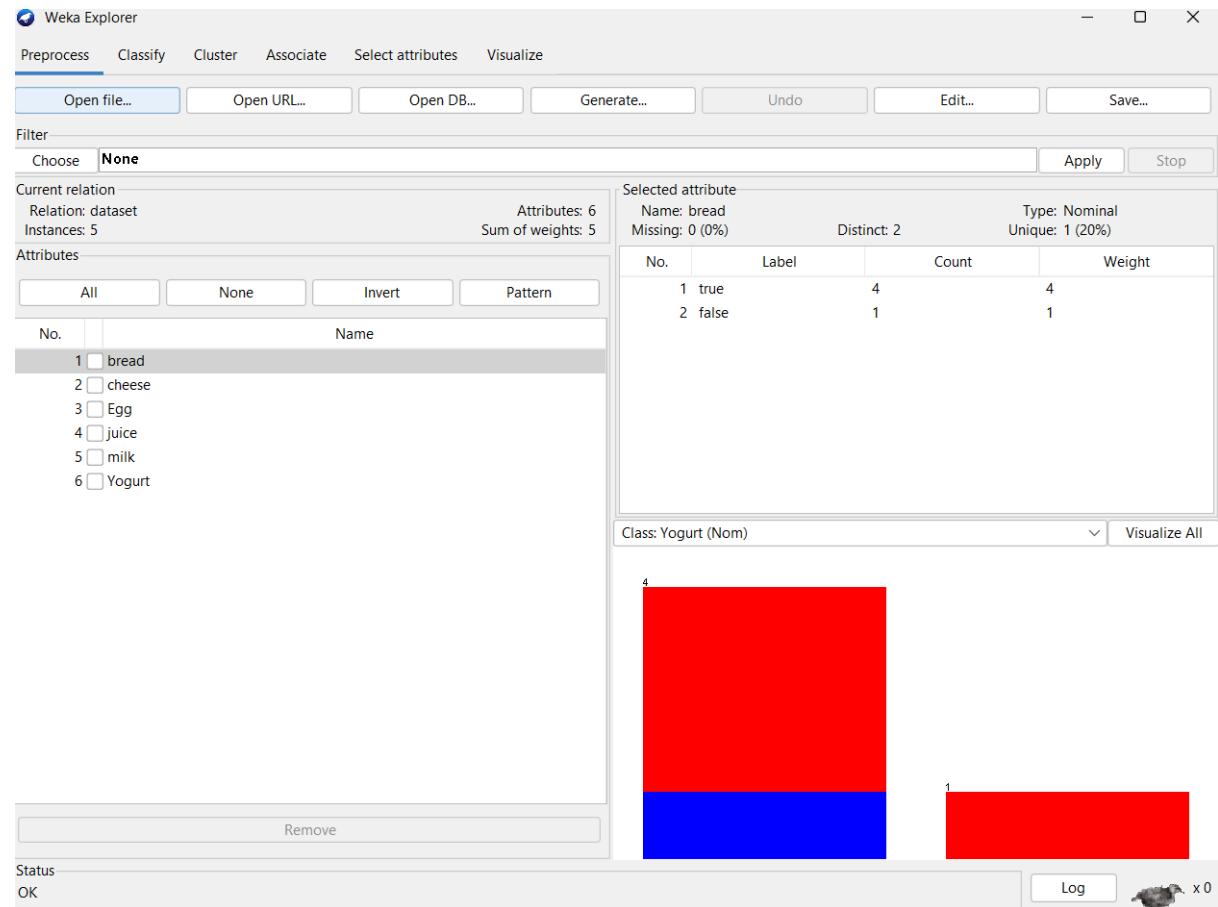
```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
+ Go to file/function Addins
7.R* x
1 # Data
2 persons <- c("Gopu", "Babu", "Baby", "Gopal", "Krishna", "Jai", "Dev", "Malini", "Hema", "Anu")
3 vegetarian_status <- c("yes", "yes", "yes", "no", "yes", "no", "no", "yes", "yes", "yes")
4
5 # Create a data frame
6 data <- data.frame(persons, vegetarian_status)
7
8 # Count the number of vegetarians and non-vegetarians
9 veg_count <- sum(vegetarian_status == "yes")
10
25:1 (Top Level) ↴ R Script ↴
R 4.4.1 -/-
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.

> # Data
> persons <- c("Gopu", "Babu", "Baby", "Gopal", "Krishna", "Jai", "Dev", "Malini", "Hema", "Anu")
> vegetarian_status <- c("yes", "yes", "yes", "no", "yes", "no", "no", "yes", "yes", "yes")
> # Create a data frame
> data <- data.frame(persons, vegetarian_status)
> # Count the number of vegetarians and non-vegetarians
> veg_count <- sum(vegetarian_status == "yes")
> non_veg_count <- sum(vegetarian_status == "no")
> # Determine which type has the greater count
> if (veg_count > non_veg_count) {
+   greater_count_type <- "Vegetarians"
+ } else if (non_veg_count > veg_count) {
+   greater_count_type <- "Non-Vegetarians"
+ } else {
+   greater_count_type <- "Both are equal"
+ }
> # Results
> cat("Number of Vegetarians:", veg_count, "\n")
Number of Vegetarians: 7
> cat("Number of Non-Vegetarians:", non_veg_count, "\n")
Number of Non-Vegetarians: 3
> cat("Type with Greater Count:", greater_count_type, "\n")
Type with Greater Count: Vegetarians
>
> |
```

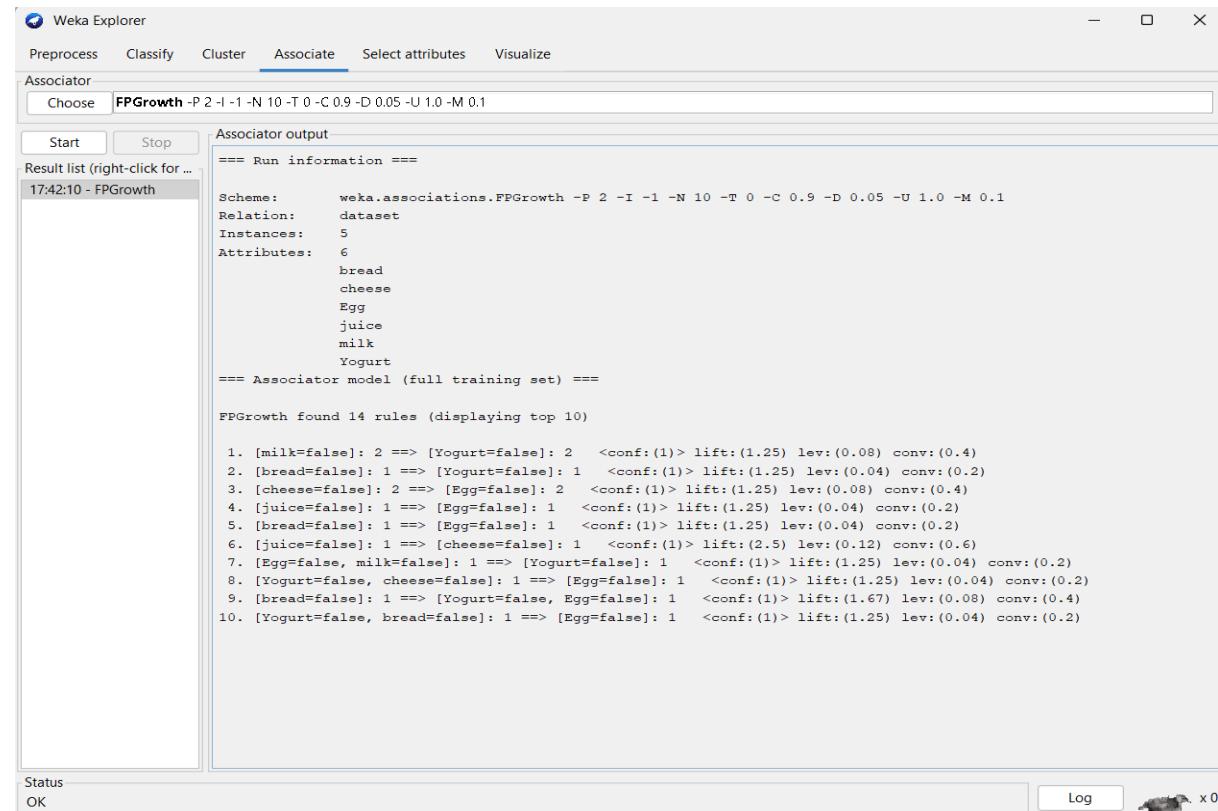
Q26:



Q27:



FP GROWTH:



Q28: