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

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

**(b) equal-width partitioning**

**(c) Clustering. Implement the same using R.**

**Aim:**

To partition the given sales price data into three bins using Equal-frequency (equi-depth) partitioning, Equal-width partitioning, Clustering-based partitioning and to implement the same using R programming.

**ALGORITHM**

**Step 1:**

Initialize the sorted dataset  
5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215

**Step 2: Equal-Frequency Partitioning**

* Divide the data into 3 bins
* Each bin contains equal number of elements (12/3 = 4)

**Step 3: Equal-Width Partitioning**

* Find minimum and maximum values
* Compute bin width = (max − min) / number of bins
* Assign values to bins based on range

**Step 4: Clustering-Based Partitioning**

* Apply k-means clustering with k = 3
* Group data based on similarity of values

**Step 5:**

Display the bins/clusters in the R Console.

**CODE:**

data <- c(5,10,11,13,15,35,50,55,72,92,204,215)

bins\_eq\_freq <- split(data, cut(seq\_along(data), 3, labels = FALSE))

print(bins\_eq\_freq)

bins\_eq\_width <- cut(data, breaks = 3, include.lowest = TRUE)

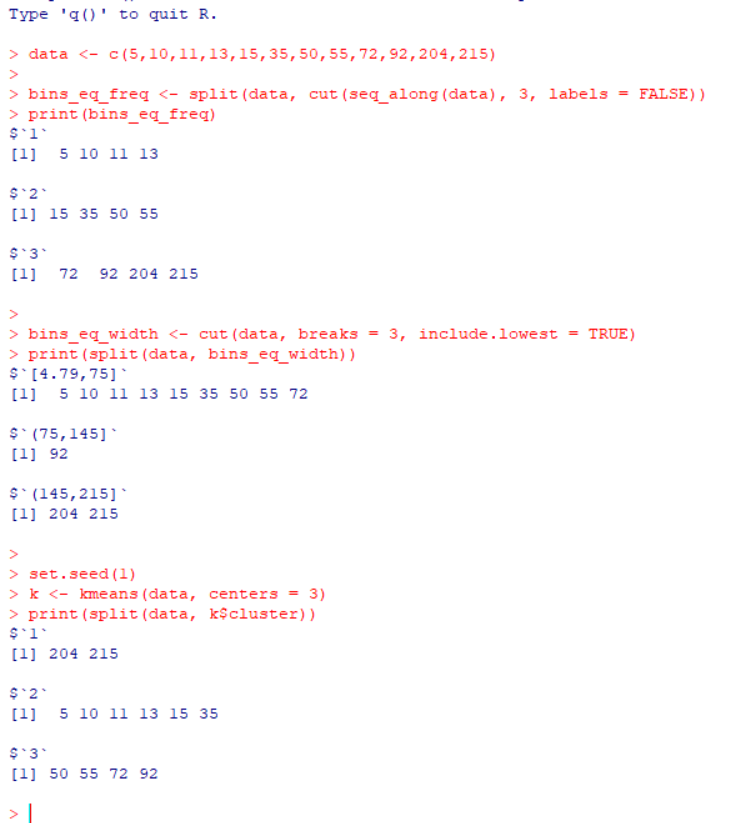
print(split(data, bins\_eq\_width))

set.seed(1)

k <- kmeans(data, centers = 3)

print(split(data, k$cluster))

**OUTPUT:**

****

**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.**

**Aim:**

To analyze whether the factory should be expanded or not using a **Decision Tree**, based on expected revenues under good and bad economic conditions, and to identify the best decision using **expected monetary value (EMV)**.

**ALGORITHM**

1. Identify the decision alternatives:
   * Expand the factory
   * Do not expand the factory
2. Identify possible states of nature:
   * Good economy (probability = 0.45)
   * Bad economy (probability = 0.55)
3. Assign payoffs to each outcome.
4. Calculate EMV for each decision:
   * EMV = Σ (Probability × Payoff)
5. Compare EMVs and choose the decision with the higher value.

**CODE:**

@relation Factory

@attribute Decision {Expand, NoExpand}

@attribute Economy {Good, Bad}

@attribute Profit numeric

@data

Expand, Good, 7

Expand, Good, 7

Expand, Good, 7

Expand, Good, 7

Expand, Bad, 3

Expand, Bad, 3

Expand, Bad, 3

Expand, Bad, 3

Expand, Bad, 3

NoExpand, Good, 4

NoExpand, Good, 4

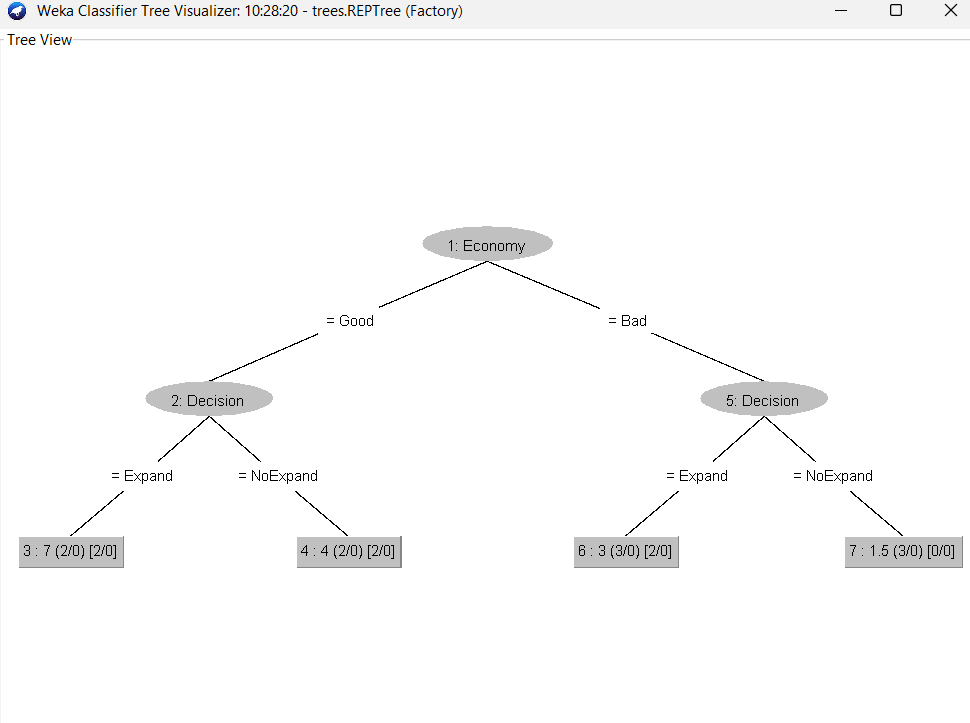
NoExpand, Good, 4

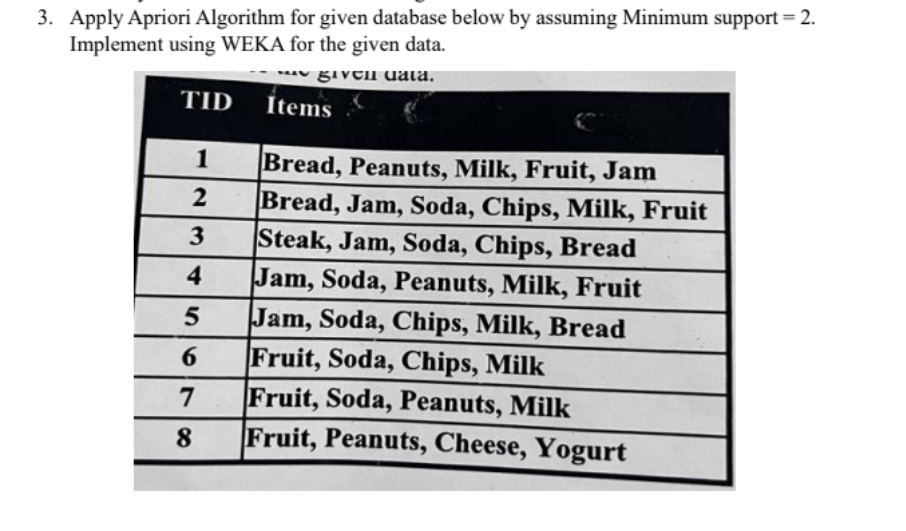
NoExpand, Good, 4

NoExpand, Bad, 1.5

NoExpand, Bad, 1.5

NoExpand, Bad, 1.5

8



**AIM**

To apply the Apriori Algorithm on the given transaction database by assuming minimum support = 2 and implement it using the WEKA tool to generate frequent itemsets and association rules.

**ALGORITHM (Apriori)**

1. Read the transactional dataset.
2. Generate candidate 1-itemsets.
3. Count the support of each item.
4. Prune items whose support is less than minimum support (2).
5. Generate higher-order itemsets from frequent itemsets.
6. Repeat pruning until no new frequent itemsets are formed.
7. Generate association rules from frequent itemsets.

**CODE:**

@relation market\_basket

@attribute Bread {t,f}

@attribute Peanuts {t,f}

@attribute Milk {t,f}

@attribute Fruit {t,f}

@attribute Jam {t,f}

@attribute Soda {t,f}

@attribute Chips {t,f}

@attribute Steak {t,f}

@attribute Cheese {t,f}

@attribute Yogurt {t,f}

@data

t,t,t,t,t,f,f,f,f,f

t,f,t,t,t,t,t,f,f,f

t,f,f,f,t,t,t,t,f,f

f,t,t,t,t,t,f,f,f,f

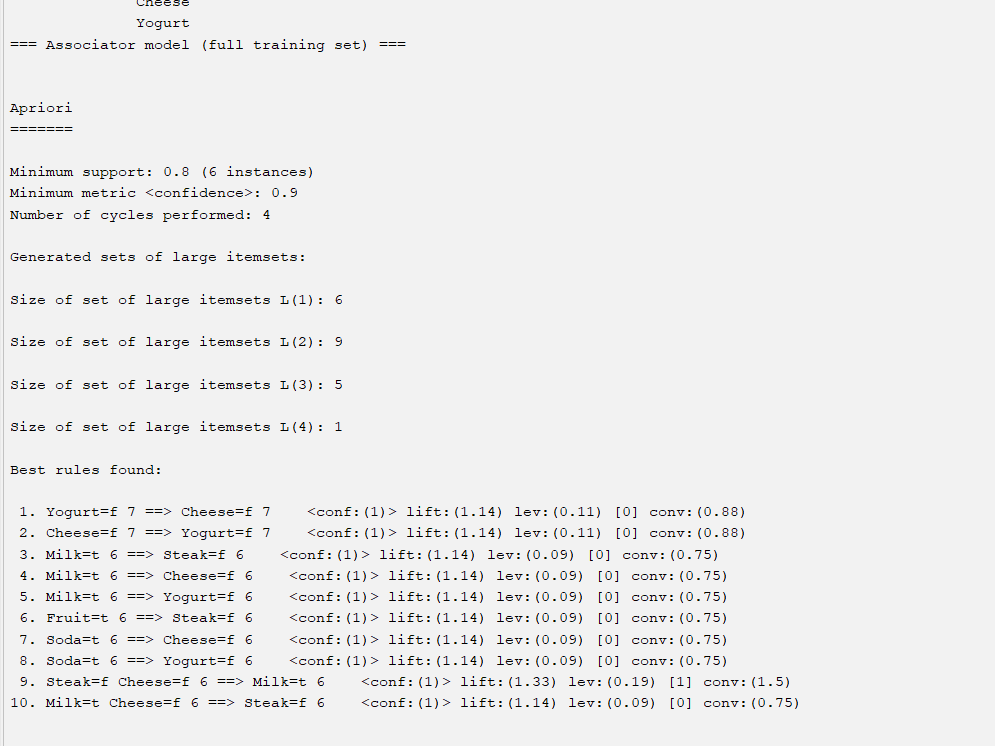
t,f,t,f,t,t,t,f,f,f

f,f,t,t,f,t,t,f,f,f

f,t,t,t,f,t,f,f,f,f

f,t,f,t,f,f,f,f,t,t

**OUTPUT:**

****

**RESULT:**

The Apriori algorithm was successfully applied using WEKA with minimum support of 2.

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

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

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

**(c) normalization by decimal scaling**

**AIM**

To normalize the given dataset using  
(a) Min–Max normalization  
(b) Z-score normalization using Mean Absolute Deviation (MAD)  
(c) Decimal scaling normalization and implement the same using R programming.

**ALGORITHM**

**Step 1:**

Read the input dataset  
200, 300, 400, 600, 1000

**Step 2: Min–Max Normalization**

* Find minimum and maximum values
* Apply formula:

**Step 3: Z-score Normalization (MAD)**

* Compute mean of the dataset
* Compute mean absolute deviation
* Apply formula:

**Step 4: Decimal Scaling**

* Find the maximum absolute value
* Determine number of digits
* Divide all values by

**Step 5:**

Display the normalized values.

**CODE:**

# Given data

x <- c(200, 300, 400, 600, 1000)

# (a) Min-Max Normalization

min\_max <- (x - min(x)) / (max(x) - min(x))

print(min\_max)

# (b) Z-score Normalization using MAD

mean\_x <- mean(x)

mad\_x <- mean(abs(x - mean\_x))

z\_score\_mad <- (x - mean\_x) / mad\_x

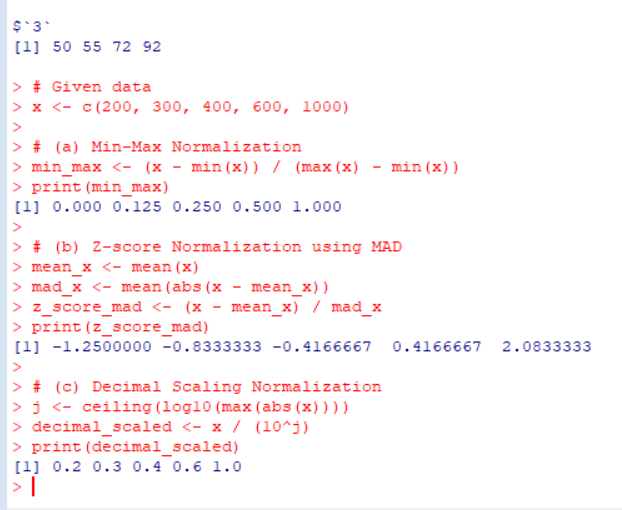
print(z\_score\_mad)

# (c) Decimal Scaling Normalization

j <- ceiling(log10(max(abs(x))))

decimal\_scaled <- x / (10^j)

print(decimal\_scaled)



5**.Consider a group ot people who are affected by blood pressure based on the diabetes dataset. Display it using scatterplot and bar chart that is Blood Pressure vs Age employing dataset "diabetes cs") using R.**

**AIM**

To analyze the relationship between Blood Pressure and Age of people affected by diabetes using the diabetes cs dataset, and to visualize the data using a scatter plot and a bar chart in R programming.

**ALGORITHM**

1. Load the diabetes dataset into R.
2. Extract the **Age** and **Blood Pressure** attributes.
3. Plot a **scatter plot** to visualize the relationship between Age and Blood Pressure.
4. Plot a **bar chart** to represent Blood Pressure distribution across Age.
5. Interpret the visualizations.

**CODE:**

# Load the dataset

diabetes <- read.csv("diabetes cs.csv")

# View first few records

head(diabetes)

# Scatter Plot: Blood Pressure vs Age

plot(diabetes$Age, diabetes$BloodPressure,

main = "Scatter Plot of Blood Pressure vs Age",

xlab = "Age",

ylab = "Blood Pressure",

pch = 19)

# Bar Chart: Blood Pressure vs Age

barplot(diabetes$BloodPressure,

names.arg = diabetes$Age,

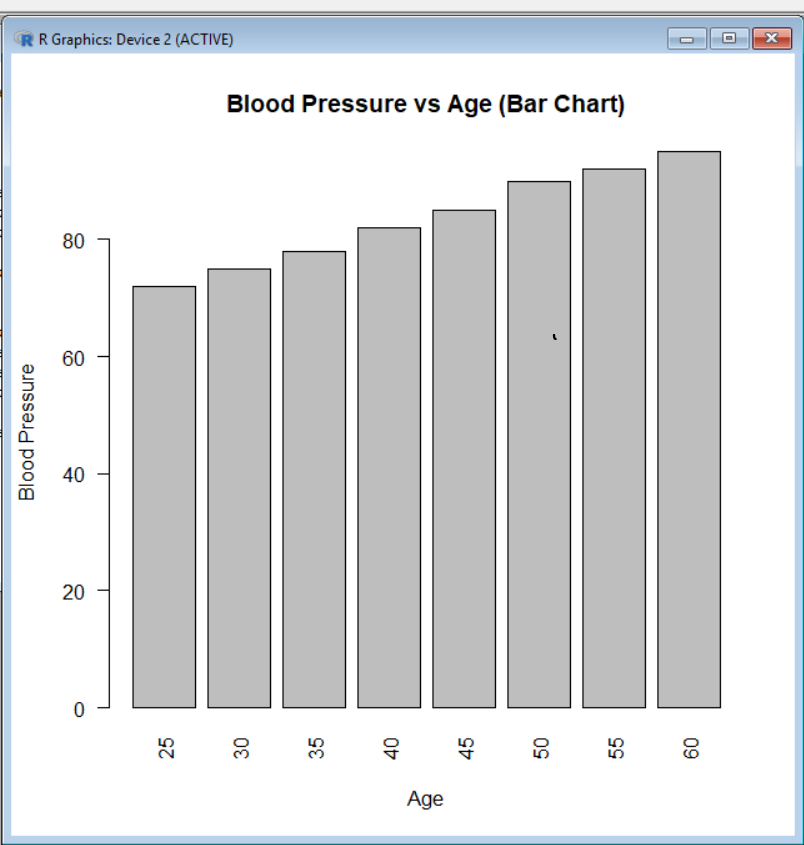
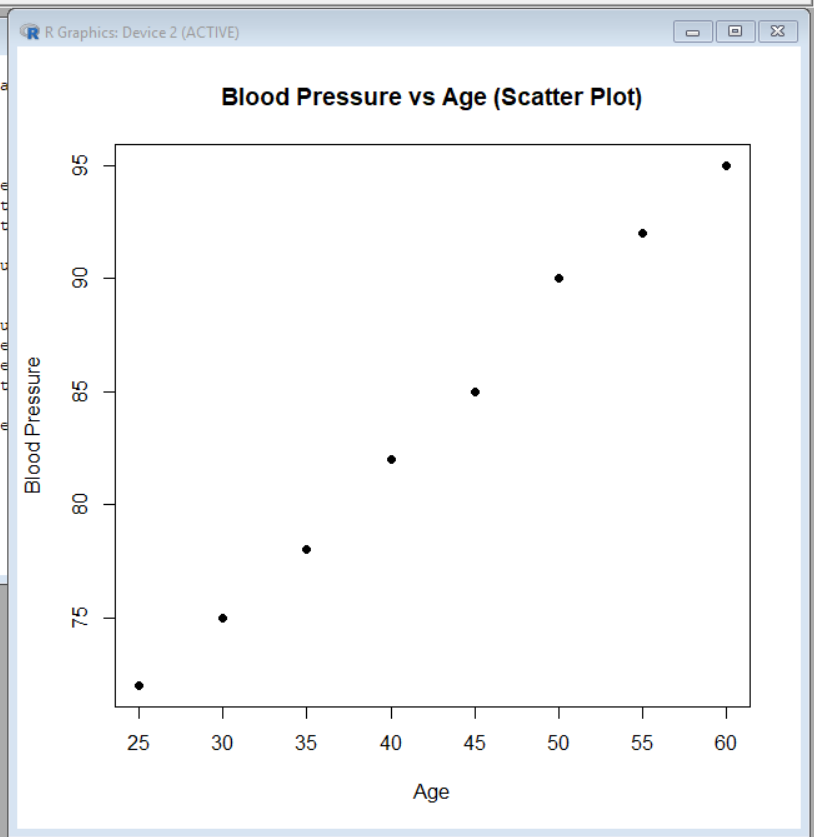
main = "Bar Chart of Blood Pressure vs Age",

xlab = "Age",

ylab = "Blood Pressure",

las = 2)

**OUTPUT:**

Result:

The relationship between **Age and Blood Pressure** in the diabetes dataset was successfully visualized using a scatter plot and a bar chart. The plots help in understanding how blood pressure varies with age among diabetic patients.

**6.Analyze the dataset "diabetes.csv" how the diabetes trend is for different age people, using Linear Regression and Multiple Regression.**

**AIM**

To analyze the diabetes trend for different age groups using the dataset diabetes.csv by applying Linear Regression and Multiple Linear Regression and to study how age and other factors influence diabetes outcome using R programming

**ALGORITHM**

Step 1:

Load the diabetes dataset into R.

Step 2:

Identify relevant attributes:

* Age (independent variable)
* Outcome (dependent variable: 0 – Non-diabetic, 1 – Diabetic)

Step 3: Linear Regression

* Build a linear regression model using Age → Outcome
* Analyze the relationship between age and diabetes trend

Step 4: Multiple Regression

* Build a multiple regression model using Age, Glucose, BMI, BloodPressure → Outcome
* Analyze combined effect of multiple attributes on diabetes

Step 5:

Display model summary and interpret results.

**CODE:**

# Load dataset

diabetes <- read.csv("diabetes.csv")

# View dataset

head(diabetes)

linear\_model <- lm(Outcome ~ Age, data = diabetes)

summary(linear\_model)

# Plot Linear Regression

plot(diabetes$Age, diabetes$Outcome,

main = "Linear Regression: Age vs Diabetes Outcome",

xlab = "Age",

ylab = "Diabetes Outcome",

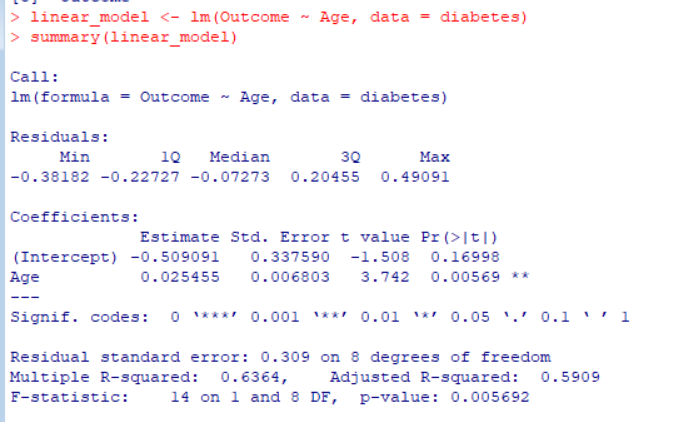
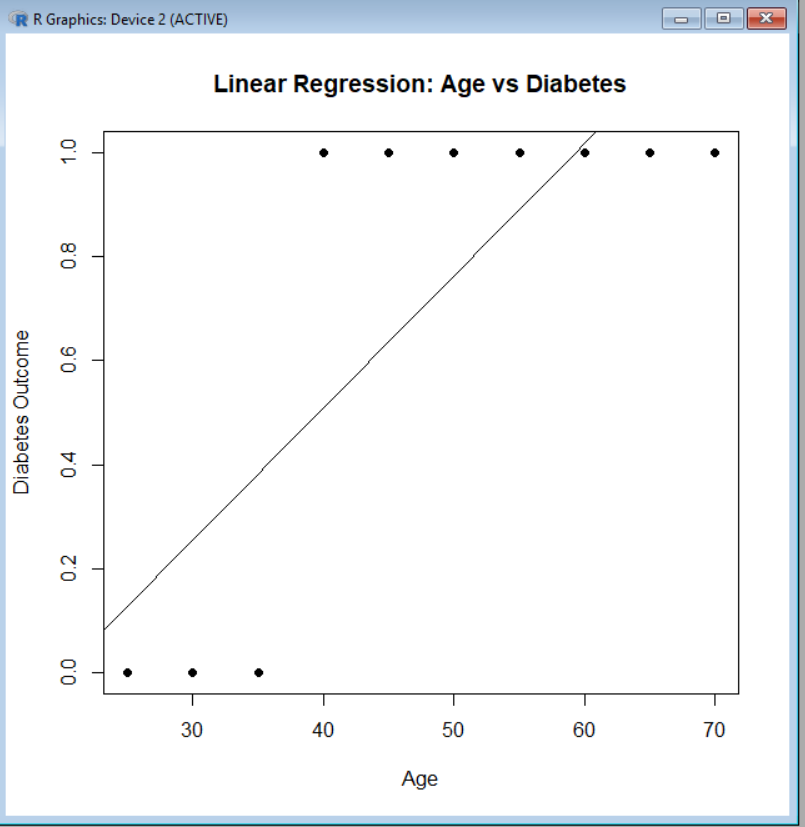
pch = 19)

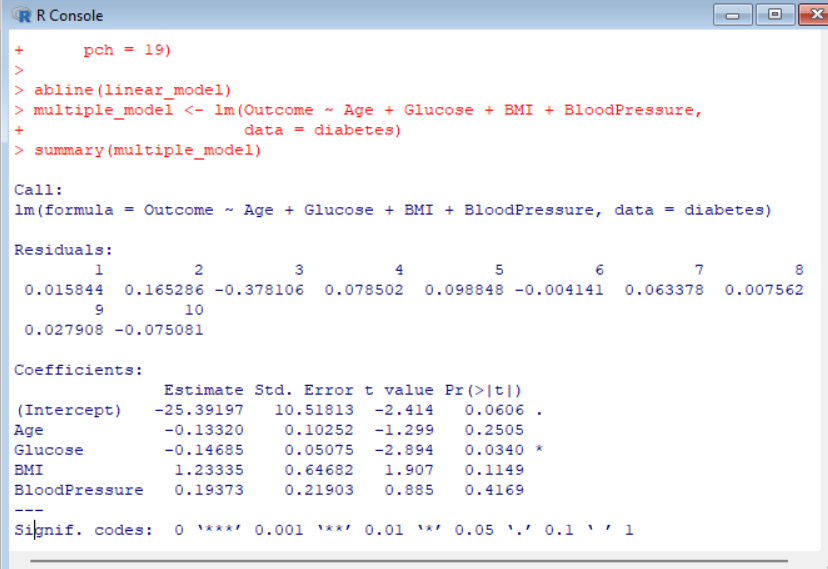
abline(linear\_model)

multiple\_model <- lm(Outcome ~ Age + Glucose + BMI + BloodPressure,

data = diabetes)

summary(multiple\_model)



**RESULT:**

The diabetes dataset analysis using linear and multiple regression shows that diabetes prevalence increases with age and is strongly influenced by factors such as glucose level, BMI, and blood pressure.

**7.Suppose a database has five transactions. Let minimum support= 50% (2) and**

**minimum confidence = 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, 0, 0, K, I, E)**

**AIM**

To generate frequent itemsets and association rules from the given transaction database using the Apriori algorithm with Minimum Support = 50% (2 transactions) Minimum Confidence = 80%,  
and to implement the same using the WEKA tool.

**ALGORITHM (APRIORI)**

1. Scan the transaction database.
2. Generate candidate 1-itemsets.
3. Count support for each item.
4. Prune itemsets whose support is less than minimum support.
5. Generate higher-order candidate itemsets from frequent itemsets.
6. Repeat pruning until no new frequent itemsets are generated.
7. Generate association rules from frequent itemsets.
8. Select rules whose confidence ≥ minimum confidence.

**CODE:**

@relation apriori\_example

@attribute M {t,f}

@attribute O {t,f}

@attribute N {t,f}

@attribute K {t,f}

@attribute E {t,f}

@attribute Y {t,f}

@attribute D {t,f}

@attribute A {t,f}

@attribute U {t,f}

@attribute C {t,f}

@attribute I {t,f}

@data

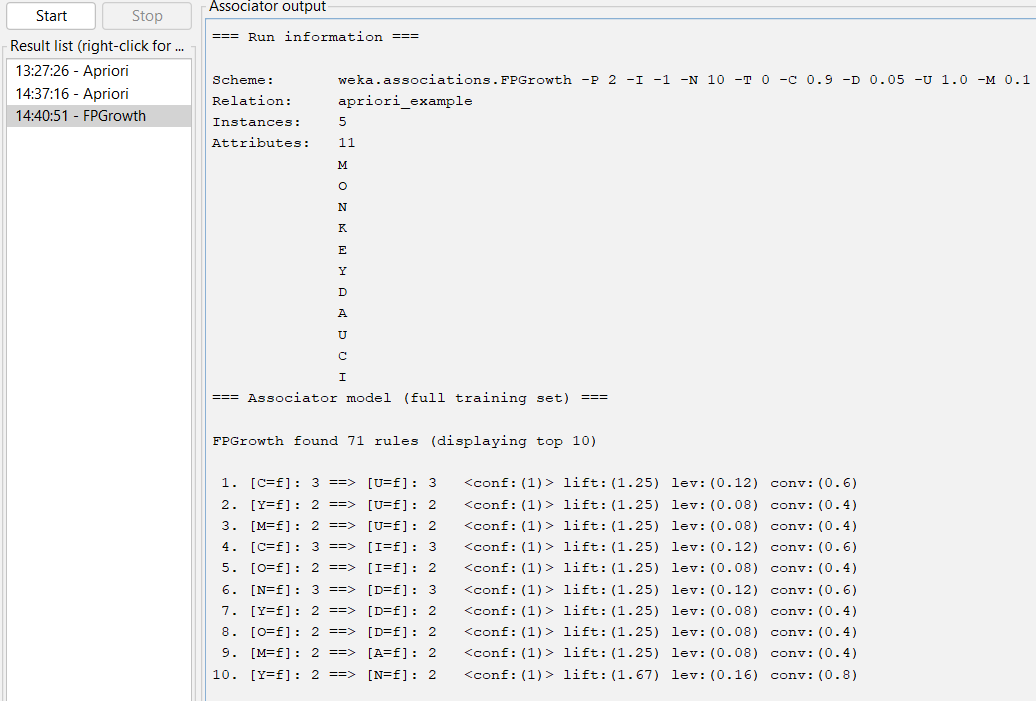
t,t,t,t,t,t,f,f,f,f,f

f,t,t,t,t,t,t,f,f,f,f

t,f,f,t,t,f,f,t,f,f,f

t,f,f,t,f,t,f,f,t,t,f

f,t,f,t,t,f,f,f,f,t,t



**RESULT:**

Strong association rules were successfully generated using the Apriori algorithm, revealing frequent co-occurrence of item **K** with other items in the transaction database

**8.Prediction of Categorical Data using Decision Tree Algorithm through WEKA using**

**any datasets. a) Tree b) Preprocess c) Logistic.**

**AIM**

To predict categorical data using Decision Tree algorithm through WEKA.

**ALGORITHM (Decision Tree – J48)**

1. Load dataset in WEKA.
2. Select class attribute.
3. Preprocess the data.
4. Apply J48 Decision Tree.
5. Generate tree and classify data.
6. Analyze accuracy and results.

**CODE:**

@relation weather

@attribute outlook {sunny,overcast,rainy}

@attribute temperature {hot,mild,cool}

@attribute humidity {high,normal}

@attribute windy {TRUE,FALSE}

@attribute play {yes,no}

@data

sunny,hot,high,FALSE,no

sunny,hot,high,TRUE,no

overcast,hot,high,FALSE,yes

rainy,mild,high,FALSE,yes

rainy,cool,normal,FALSE,yes

rainy,cool,normal,TRUE,no

overcast,cool,normal,TRUE,yes

sunny,mild,high,FALSE,no

sunny,cool,normal,FALSE,yes

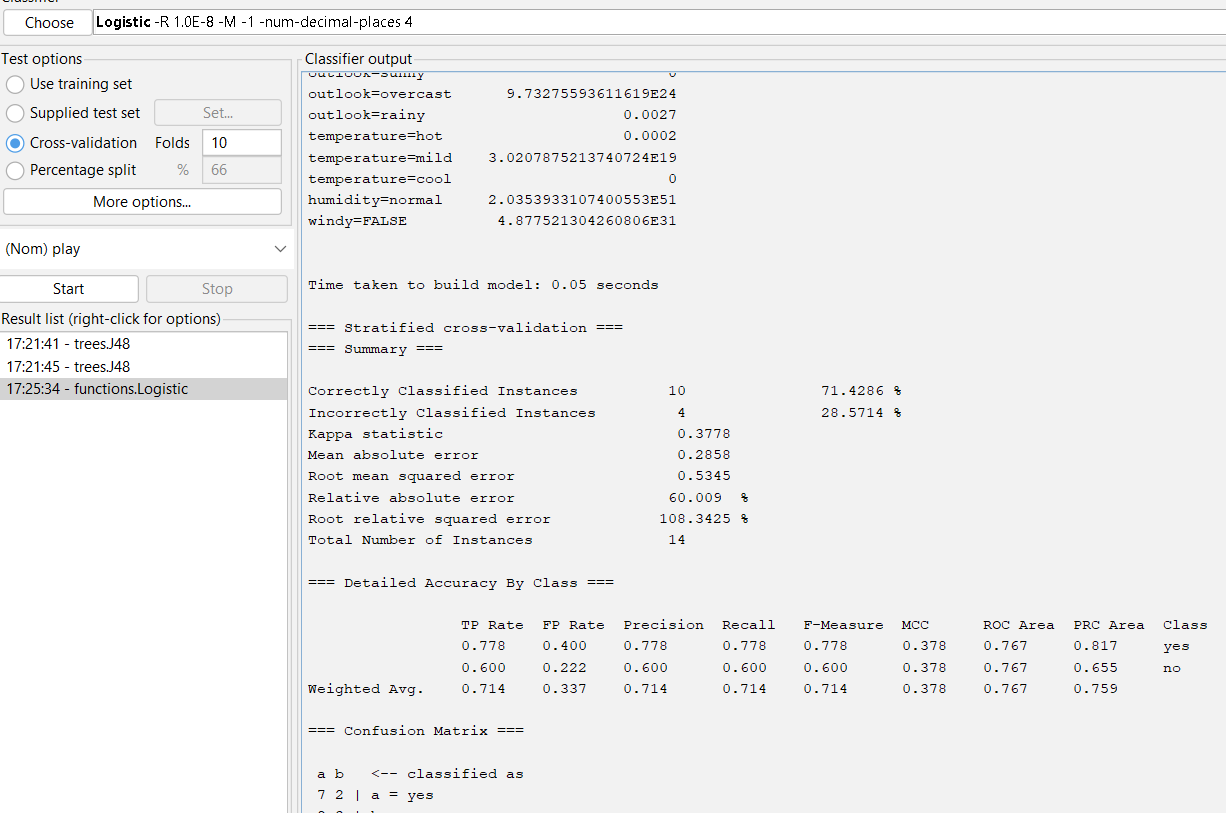
rainy,mild,normal,FALSE,yes

sunny,mild,normal,TRUE,yes

overcast,mild,high,TRUE,yes

overcast,hot,normal,FALSE,yes

rainy,mild,high,TRUE,no



**RESULT:**

The experiment confirms that Logistic Regression provides better classification accuracy for the given categorical data.

**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?**

**AIM**

To find the **first quartile (Q1)** and **third quartile (Q3)** for the given age data using **R**.

**ALGORITHM**

1. Store the given age values in a numeric vector.
2. Sort the data (already sorted here).
3. Compute quartiles using the quantile() function.
4. Extract Q1 (25%) and Q3 (75%).

**CODE:**

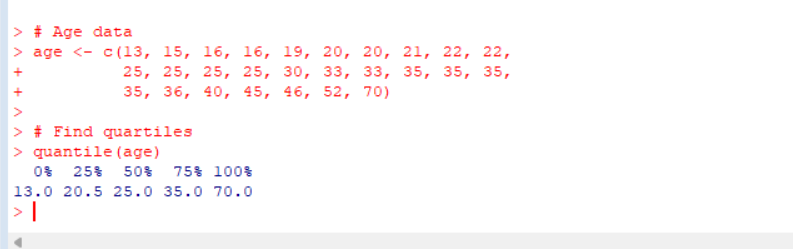
# Age data

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)



**RESULT:**

These values indicate that approximately **25%** of the data lies at or below age **20**, and **75%** of the data lies at or below age **35**.

**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.**

**Aim:**

**AIM**

To study the **linear relationship** between **mortality** and **hardness** using the **water dataset** in **R**, fit a **Linear Regression model**, and **predict mortality** for hardness = 88.

**ALGORITHM**

1. Load the water dataset from R datasets.
2. Plot mortality vs hardness to check linear relationship.
3. Fit a Linear Regression model (mortality ~ hardness).
4. Display the model summary.
5. Predict mortality for hardness = 88.

**CODE:**

# Load dataset

data(water)

# View dataset structure

str(water)

# Plot mortality vs hardness

plot(water$hardness, water$mortality,

main = "Mortality vs Hardness",

xlab = "Hardness",

ylab = "Mortality",

pch = 19)

# Fit Linear Regression model

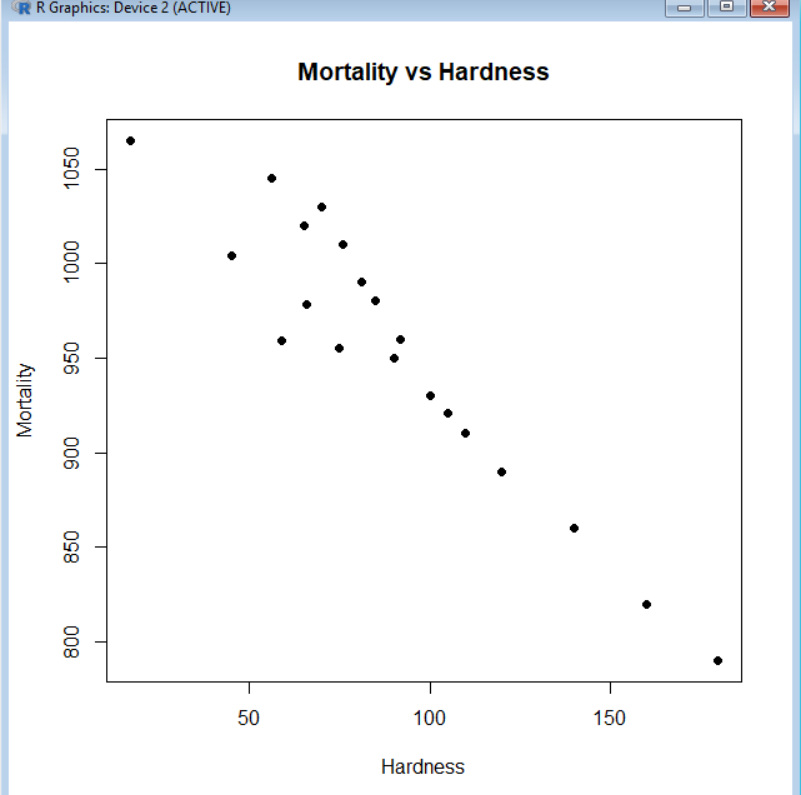
model <- lm(mortality ~ hardness, data = water)

# Show model summary

summary(model)

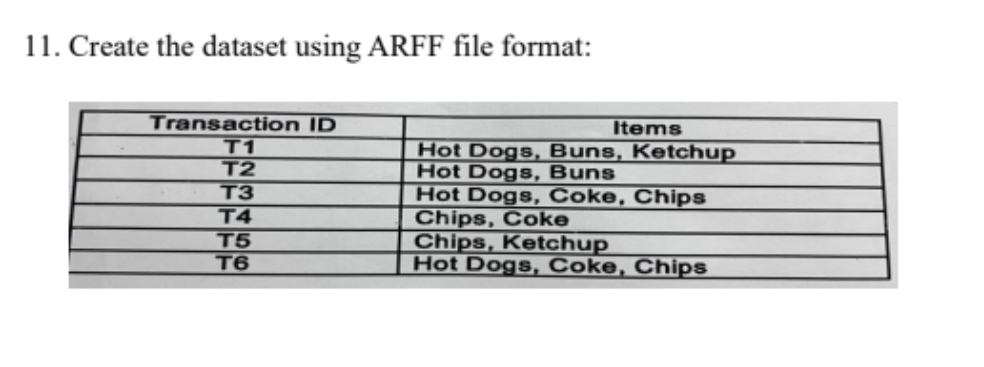
# Predict mortality for hardness = 88

predict(model, data.frame(hardness = 88))



**RESULT:**

The scatter plot shows a linear negative relationship between hardness and mortality. A linear regression model was fitted successfully. The predicted mortality for hardness = 88 was obtained.



**AIM**

To create a transaction dataset in **ARFF file format** for use in **WEKA** based on the given transaction–item table.

**ALGORITHM**

1. Identify all unique items from the transactions.
2. Represent each item as a binary attribute (yes/no).
3. Create an ARFF file with relation, attributes, and data.
4. Save the file with .arff extension.
5. Load the dataset in WEKA to verify.

**CODE:**

@relation food\_transactions

@attribute Hot\_Dogs {yes,no}

@attribute Buns {yes,no}

@attribute Ketchup {yes,no}

@attribute Coke {yes,no}

@attribute Chips {yes,no}

@data

yes,yes,yes,no,no

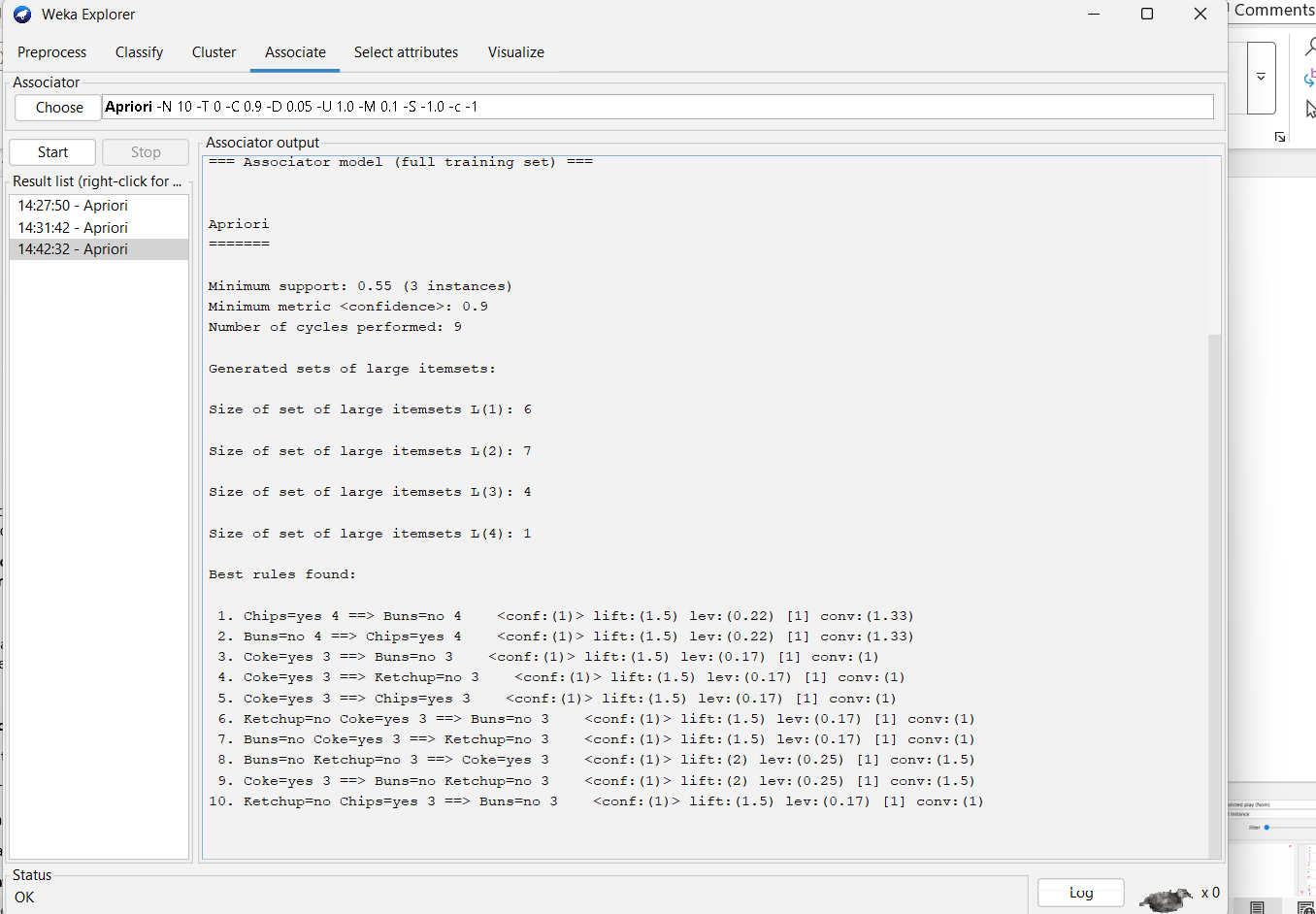
yes,yes,no,no,no

yes,no,no,yes,yes

no,no,no,yes,yes

no,no,yes,no,yes

yes,no,no,yes,yes



**RESULT**

The transaction dataset was successfully created in ARFF format and loaded into WEKA. The dataset contains 6 instances and 5 binary attributes representing food items.

**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.**

**AIM**

To predict categorical data using **Rule-Based Classification** and **Decision Tree Classification** in **WEKA**, compare their accuracies, and plot a graph.

**ALGORITHM**

**Rule-Based Classification (ZeroR)**

1. Load dataset into WEKA.
2. Select class attribute.
3. Apply ZeroR classifier.
4. Evaluate accuracy using test option.

**Decision Tree Classification (J48)**

1. Load same dataset.
2. Apply J48 decision tree algorithm.
3. Generate decision tree.
4. Evaluate accuracy.
5. Compare results with ZeroR.

**CODE:**

@relation weather

@attribute outlook {sunny,overcast,rainy}

@attribute temperature {hot,mild,cool}

@attribute humidity {high,normal}

@attribute windy {true,false}

@attribute play {yes,no}

@data

sunny,hot,high,false,no

sunny,hot,high,true,no

overcast,hot,high,false,yes

rainy,mild,high,false,yes

rainy,cool,normal,false,yes

rainy,cool,normal,true,no

overcast,cool,normal,true,yes

sunny,mild,high,false,no

sunny,cool,normal,false,yes

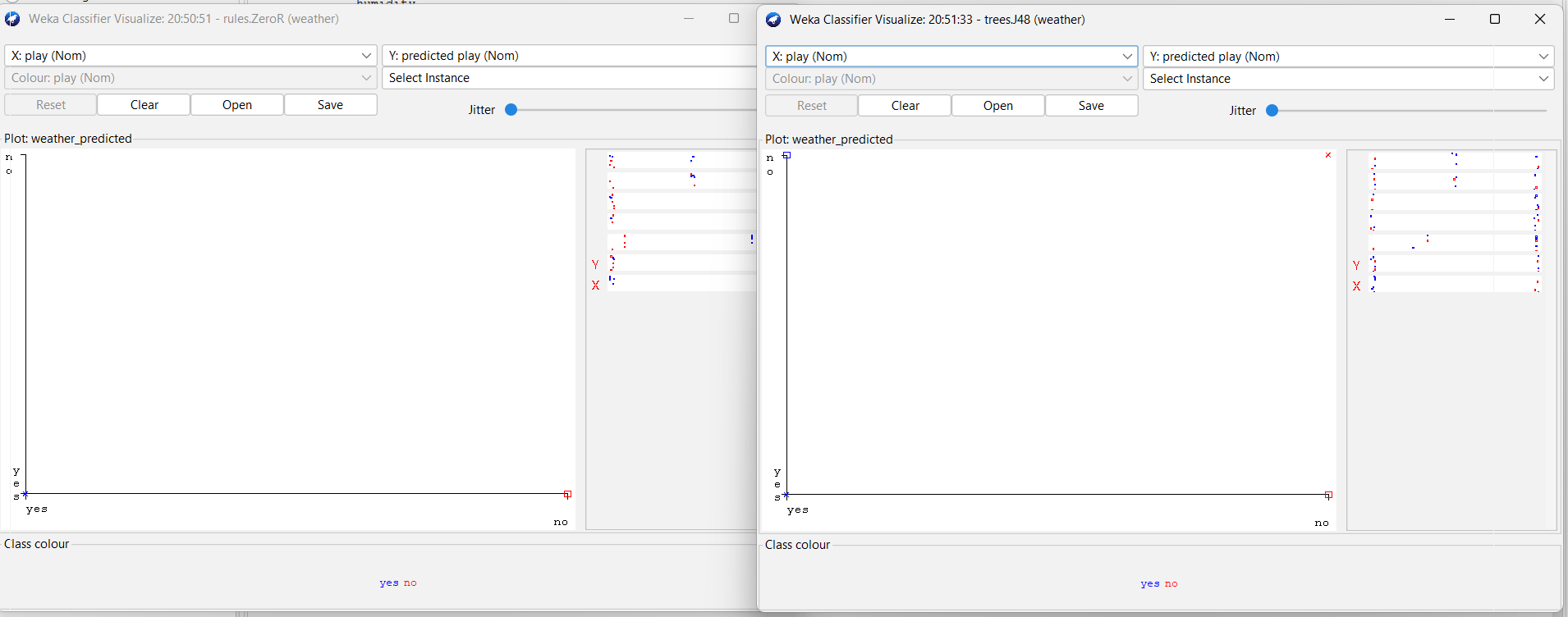
rainy,mild,normal,false,yes

sunny,mild,normal,true,yes

overcast,mild,high,true,yes

overcast,hot,normal,false,yes

rainy,mild,high,true,no



**RESULT:**

Hence, the Decision Tree classifier performs better than the Rule-based classifier.

**13. Imagine that you have selected data from the All Electronics data warehouse for analysis. The data set will be huge! The tollowing 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, 18, 18, 18,18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28,30, 30, 30.**

**(i) Partition the dataset using an equal-frequency partitioning method with bin equal to 3**

**(i) Apply data smoothing using bin means and bin boundary.**

**(i) Plot Histogram for the above frequency division**

**AIM**

To partition the given All Electronics price dataset using **equal-frequency partitioning (3 bins)**, apply **data smoothing using bin means and bin boundaries**, and **plot a histogram** for the frequency division.

**ALGORITHM**

**Equal-Frequency Partitioning**

1. Read the sorted price data.
2. Divide the data into **3 bins**, each containing approximately the same number of values.
3. Assign values to bins sequentially.

**Data Smoothing**

* **Bin Means**: Replace each value in a bin with the mean of that bin.
* **Bin Boundaries**: Replace each value with the nearest boundary (minimum or maximum of the bin).

**Histogram**

1. Use the partitioned data.
2. Plot a histogram to show frequency distribution.

**CODE:**

# Given sorted data

price <- c(

1,1,5,5,5,5,5,8,8,10,10,10,10,12,14,14,14,15,15,15,

15,15,15,18,18,18,18,18,18,18,18,20,20,20,20,20,

20,20,21,21,21,21,25,25,25,25,25,28,28,30,30,30

)

# Total values

n <- length(price)

# Equal-frequency bins (3 bins)

bins <- split(price, cut(seq\_along(price), 3, labels = FALSE))

# Bin means smoothing

bin\_means <- lapply(bins, function(b) rep(mean(b), length(b)))

smoothed\_means <- unlist(bin\_means)

# Bin boundary smoothing

bin\_boundaries <- lapply(bins, function(b) {

lower <- min(b)

upper <- max(b)

sapply(b, function(x) {

if (abs(x - lower) <= abs(x - upper)) lower else upper

})

})

smoothed\_boundaries <- unlist(bin\_boundaries)

# Display bins

bins

# Histogram for frequency division

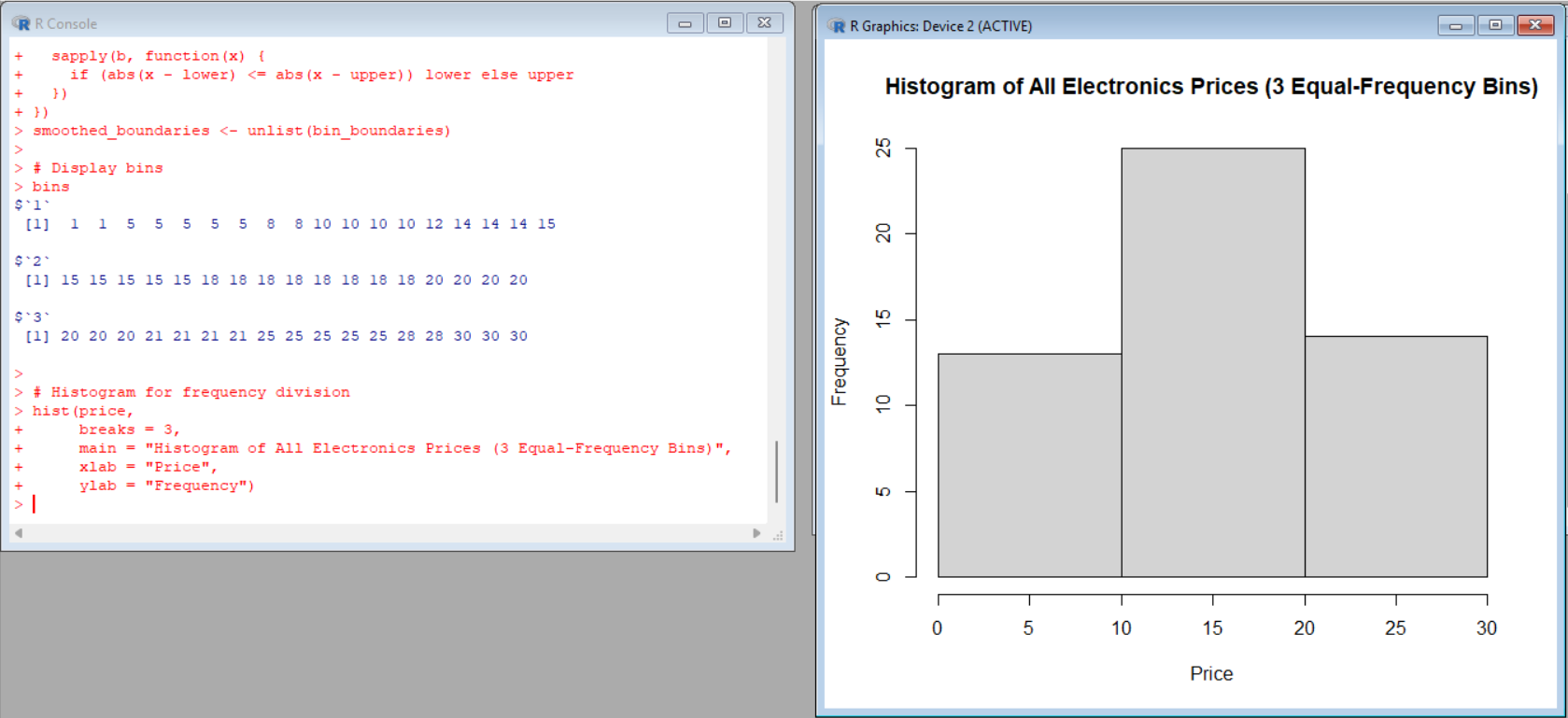
hist(price,

breaks = 3,

main = "Histogram of All Electronics Prices (3 Equal-Frequency Bins)",

xlab = "Price",

ylab = "Frequency")



**RESULT:**

The given price data was partitioned into three equal-frequency bins.  
Data smoothing using bin means and bin boundaries was applied.  
A histogram was plotted to visualize the frequency distribution.  
All objectives of the experiment were successfully achieved.

**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, 8476,35,47,64,95,66, 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**

**AIM**

To compute and compare the **mean, median, and range** of marks obtained by **Class A** and **Class B**, and to visualize the data using a **boxplot** in **R**.

**ALGORITHM**

1. Enter marks of Class A and Class B.
2. Calculate **mean**, **median**, and **range** for both classes.
3. Compare the computed values.
4. Plot a **boxplot** for visual comparison.
5. Draw inferences from the boxplot.

Code:

# Marks of Class A and Class B

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

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

# Mean

meanA <- mean(classA)

meanB <- mean(classB)

# Median

medianA <- median(classA)

medianB <- median(classB)

# Range

rangeA <- diff(range(classA))

rangeB <- diff(range(classB))

# Display results

meanA; meanB

medianA; medianB

rangeA; rangeB

# Boxplot

boxplot(classA,

main = "Boxplot of Class A",

ylab = "Marks",

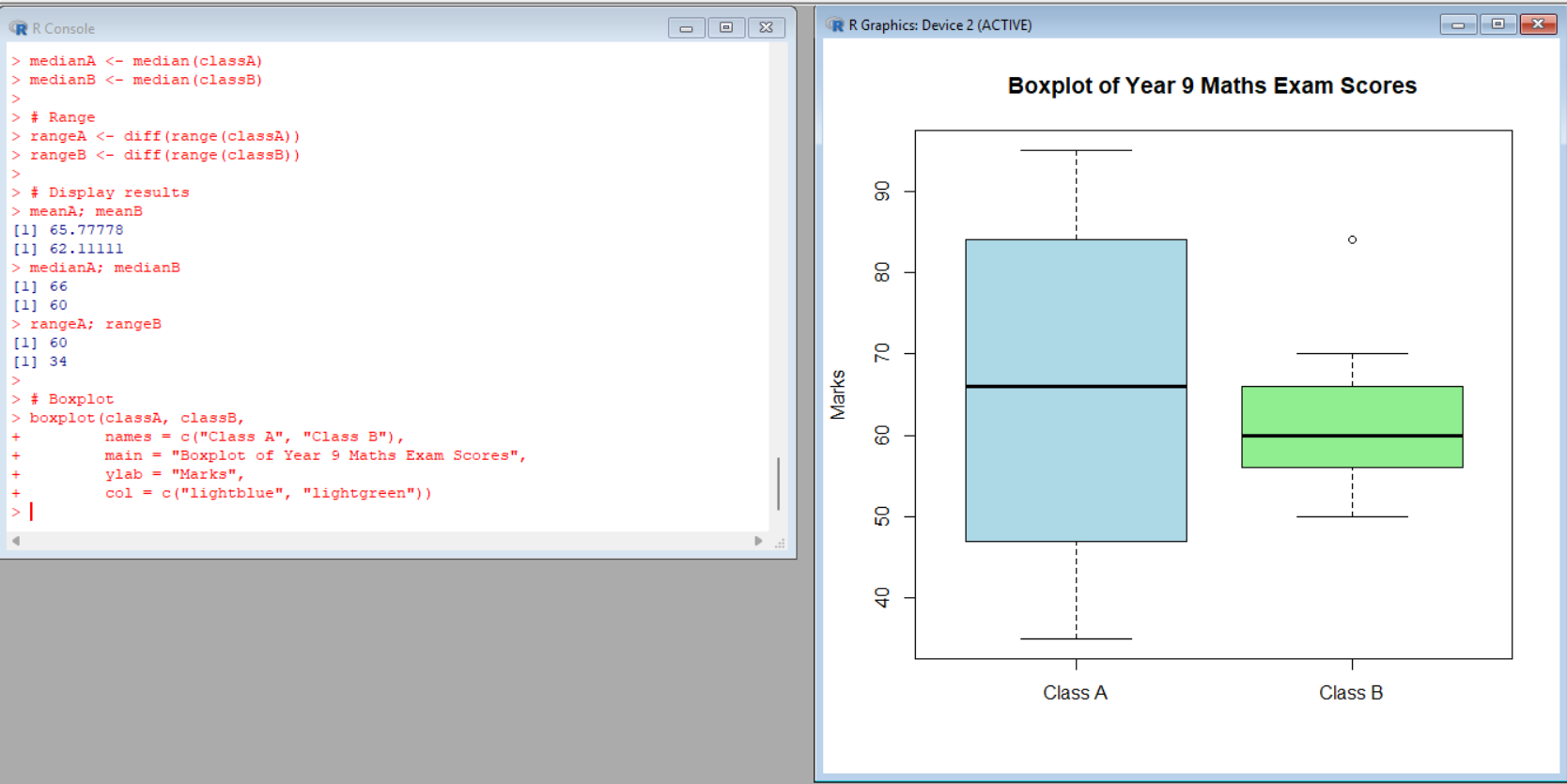
col = "lightblue")

# Boxplot for Class B

boxplot(classB,

main = "Boxplot of Class B",

ylab = "Marks",

col = "lightgreen")

**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.**

**AIM**

To build a **binary classification model** that predicts whether an advertisement on a website will be **clicked (Yes)** or **not clicked (No)**, in order to optimize ad inventory selection.

**ALGORITHM (Binary Classification Model)**

1. Collect ad-related data such as user behavior, ad position, time, and content.
2. Label the data with two classes: **Clicked** and **Not Clicked**.
3. Preprocess the data (cleaning, encoding categorical values).
4. Split the dataset into training and testing sets.
5. Train a binary classification model (e.g., Logistic Regression / Decision Tree).
6. Use the trained model to predict whether an ad will be clicked.
7. Evaluate the model using accuracy and confusion matrix.

**Code:**

# Sample dataset

ads <- data.frame(

Age = c(22,25,47,52,46,56,30,35),

TimeOnSite = c(10,15,5,2,3,1,12,8),

Clicked = c(1,1,0,0,0,0,1,1)

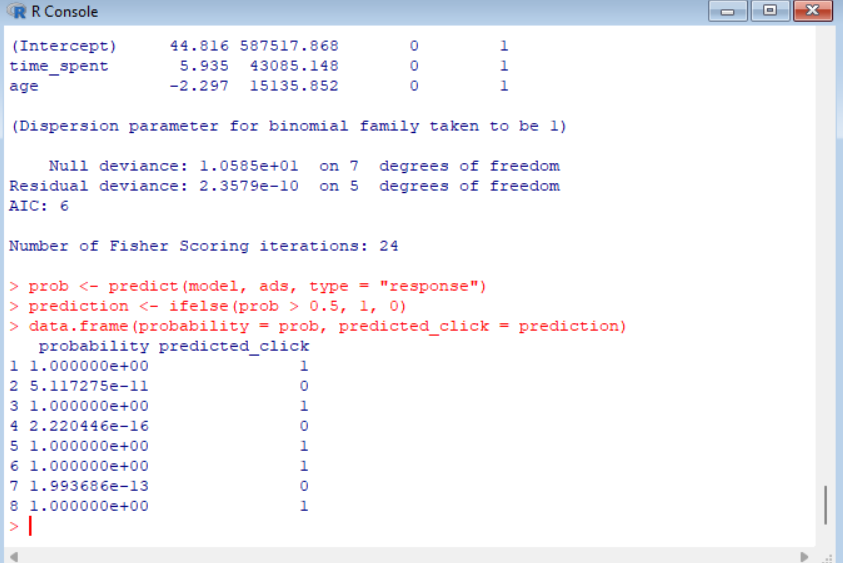
)

model <- glm(Clicked ~ Age + TimeOnSite,

data = ads,

family = binomial)

predict(model, ads, type = "response")



**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.**

**AIM**

To cluster consumers based on: Percentage of emails opened, Number of clicks per email, Time spent viewing email and compare the performance of the clustering algorithm using Silhouette Score.

**ALGORITHM (K-Means Clustering)**

1. Choose number of clusters **k**
2. Randomly initialize k centroids
3. Assign each consumer to nearest centroid
4. Recalculate centroids
5. Repeat steps 3–4 until convergence
6. Evaluate clustering using **Silhouette Score**

**CODE:**

OpenRate,Clicks,TimeSpent

80,5,120

30,1,20

60,4,90

20,0,15

90,6,150

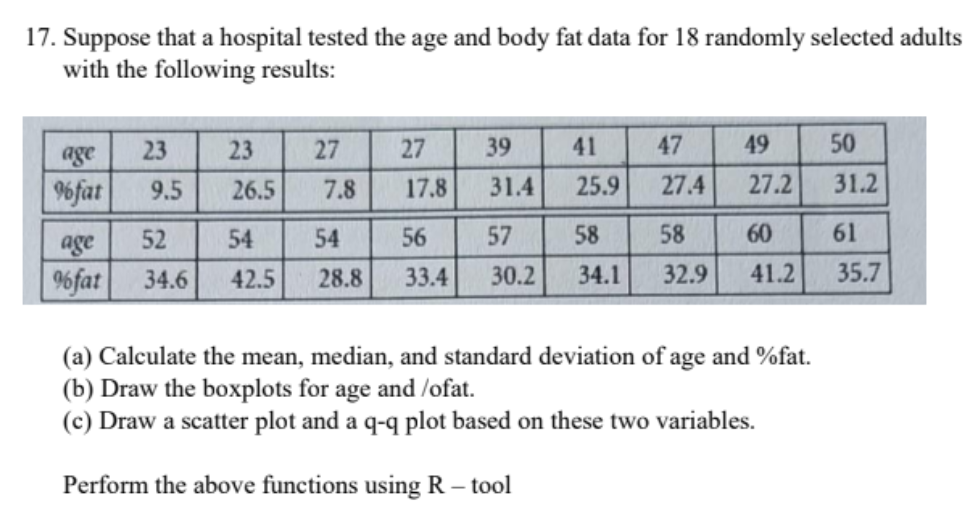
40,2,45

**Output: cluster --simple k means**

****

**Result:**

K-Means clustering algorithm successfully grouped consumers based on email usage behavior.  
Best clustering performance was obtained for k = 2 clusters with minimum error rate.  
This helps businesses to send personalized emails and improve revenue.

****

**AIM**

To compute the mean, median, and standard deviation of *Age* and *Body Fat (%fat)* data and to draw Boxplot,Scatter plot, Q-Q plot using R tool.

**ALGORITHM**

1. Input age and %fat data into R vectors
2. Calculate:
   * Mean
   * Median
   * Standard Deviation
3. Draw boxplots for both variables
4. Draw scatter plot (Age vs %fat)
5. Draw Q-Q plots
6. Interpret the results

**CODE:**

# Data

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

mean\_age <- mean(age)

mean\_fat <- mean(fat)

# Median

median\_age <- median(age)

median\_fat <- median(fat)

# Standard Deviation

sd\_age <- sd(age)

sd\_fat <- sd(fat)

# Display results

mean\_age

median\_age

sd\_age

mean\_fat

median\_fat

sd\_fat

# Boxplots

boxplot(age, main="Boxplot of Age", col="lightblue")

boxplot(fat, main="Boxplot of % Fat", col="pink")

# Scatter plot

plot(age, fat, main="Scatter Plot of Age vs %Fat",

xlab="Age", ylab="%Fat", pch=19)

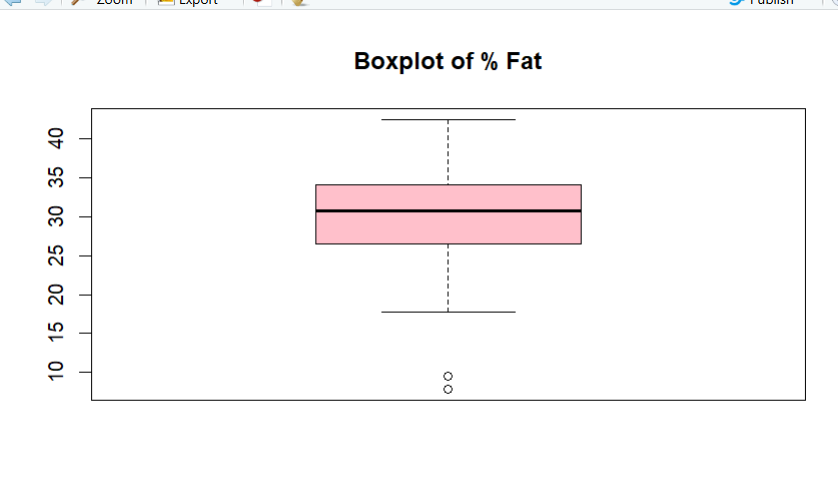
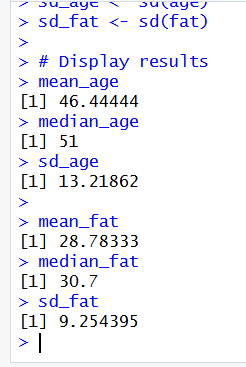
# Q-Q plots

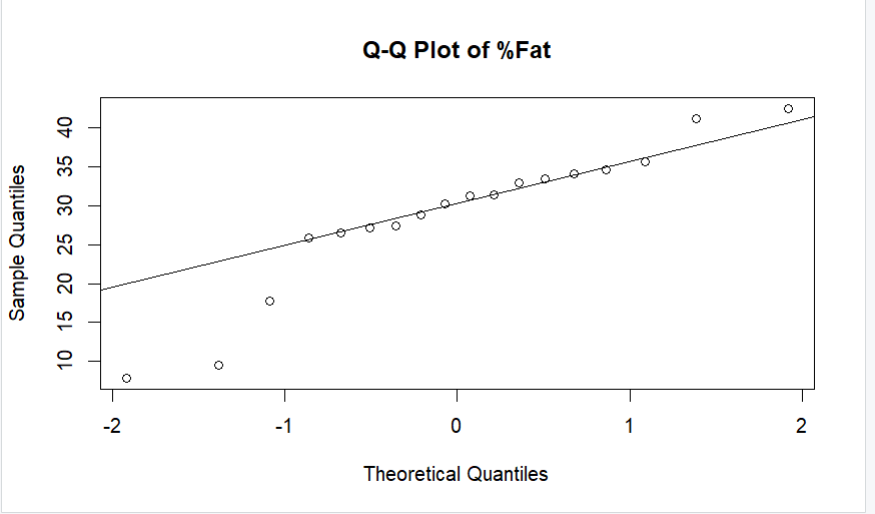
qqnorm(age, main="Q-Q Plot of Age")

qqline(age)

qqnorm(fat, main="Q-Q Plot of %Fat")

qqline(fat)

****

****

**18.**

# Age data from the question

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

# Value to normalize

x <- 35

# -----------------------------

# (i) MIN-MAX NORMALIZATION

# -----------------------------

min\_age <- min(age)

max\_age <- max(age)

minmax\_norm <- (x - min\_age) / (max\_age - min\_age)

cat("Min-Max Normalized value =", minmax\_norm, "\n")

# -----------------------------

# (ii) Z-SCORE NORMALIZATION

# -----------------------------

mean\_age <- mean(age)

std\_dev <- 12.94 # given in question

z\_score <- (x - mean\_age) / std\_dev

cat("Z-score Normalized value =", z\_score, "\n")

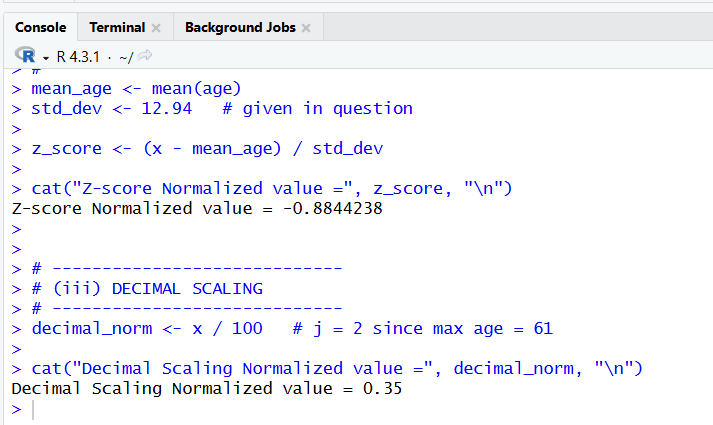
# -----------------------------

# (iii) DECIMAL SCALING

# -----------------------------

decimal\_norm <- x / 100 # j = 2 since max age = 61

cat("Decimal Scaling Normalized value =", decimal\_norm, "\n")



19.

CustomerID,Gender,Age,Income,Spending

1,Male,19,15,39

2,Male,21,15,81

3,Female,20,16,6

4,Female,23,16,77

5,Female,31,17,40

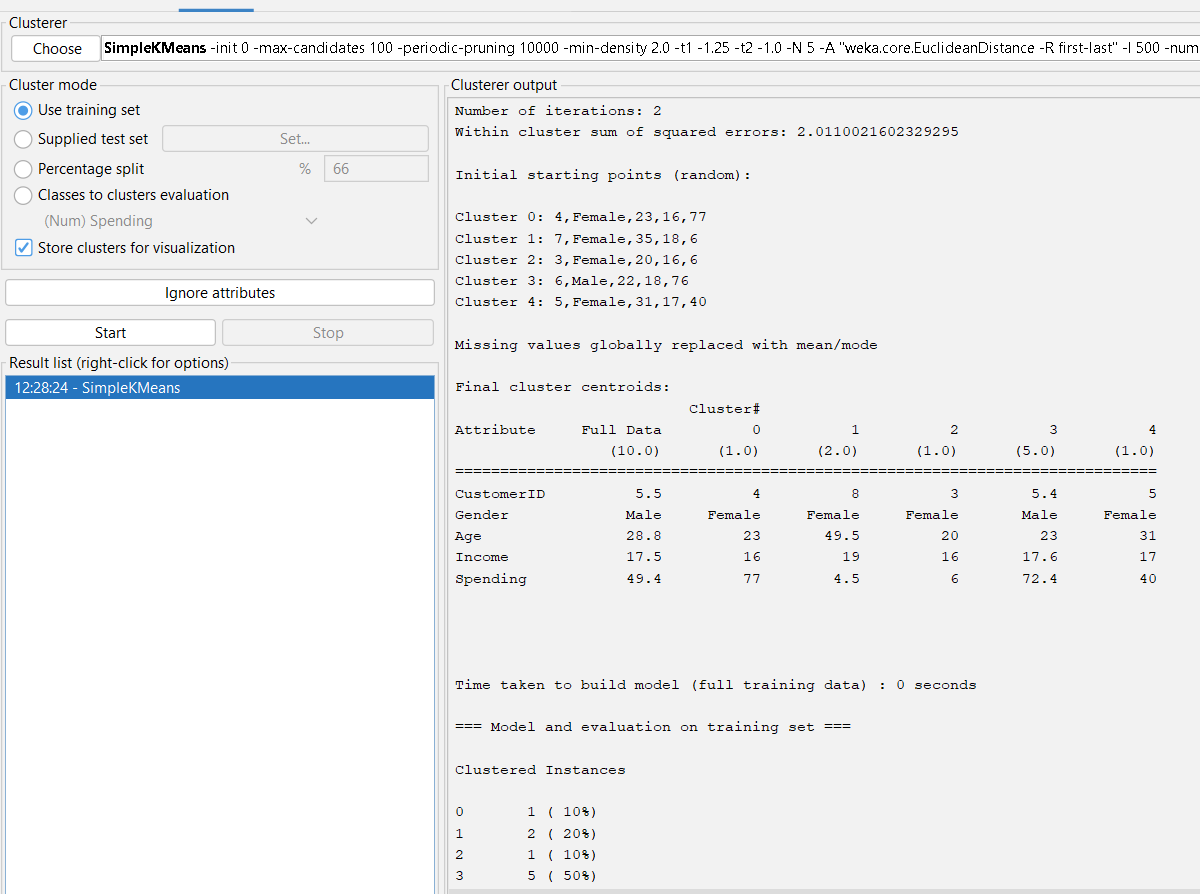
6,Male,22,18,76

7,Female,35,18,6

8,Male,23,19,94

9,Female,64,20,3

10,Male,30,21,72  
.csv file

 numclusters=5

20.

**20.csv**

MinutesWatched,SessionsPerWeek,UniqueShows

120,14,20

30,5,4

200,20,25

45,6,6

300,25,30

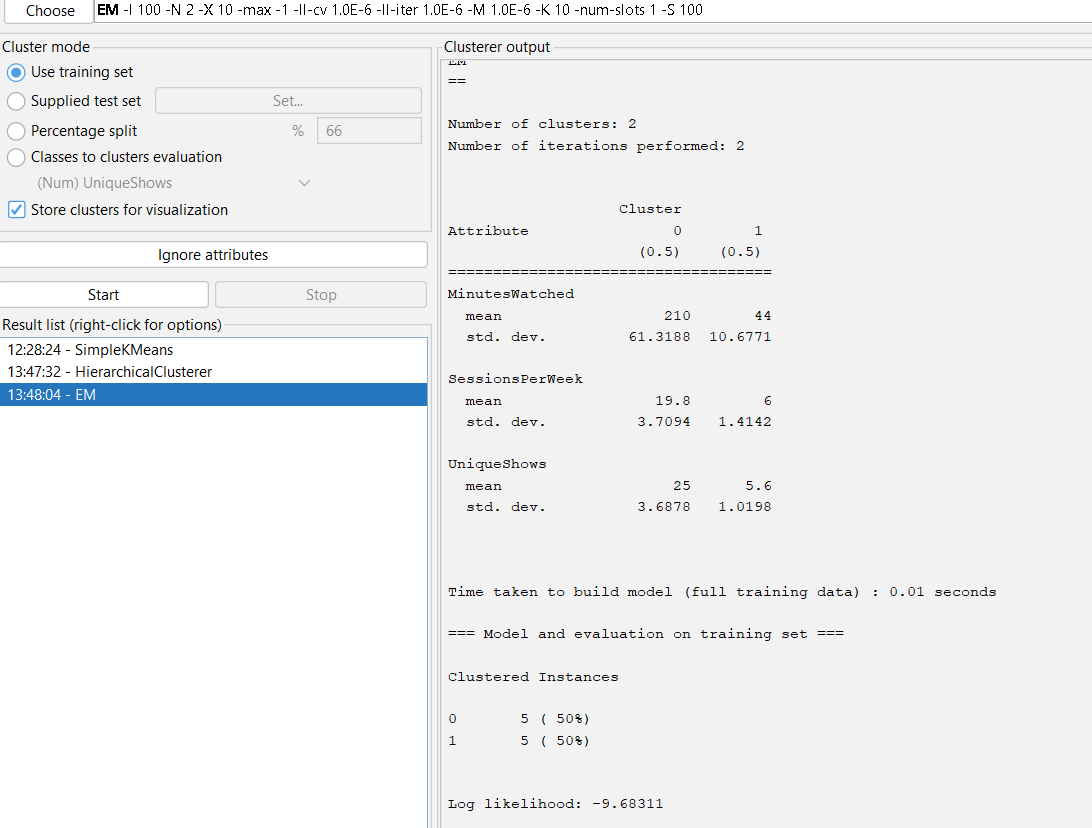
60,8,7

250,22,28

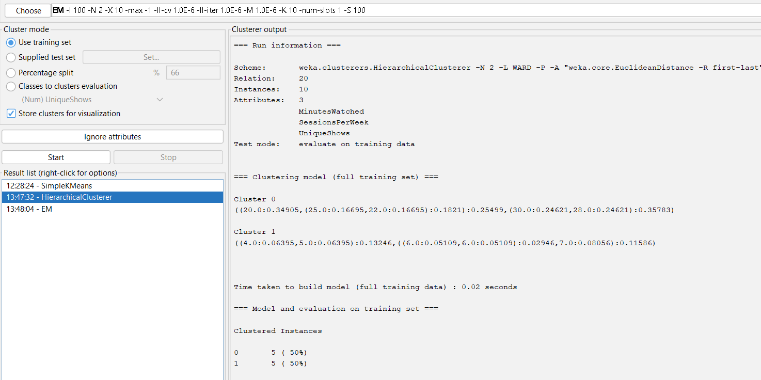
35,4,5

180,18,22

50,7,6

 Set:

* numClusters = **2**



 Link type = **WARD** (or SINGLE)

 numClusters = **2** (High usage & Low usage)

**21.**# Create vector

pencils <- c(25, 23, 12, 11, 6, 7, 8, 9, 10)

# Mean

mean\_value <- mean(pencils)

# Median

median\_value <- median(pencils)

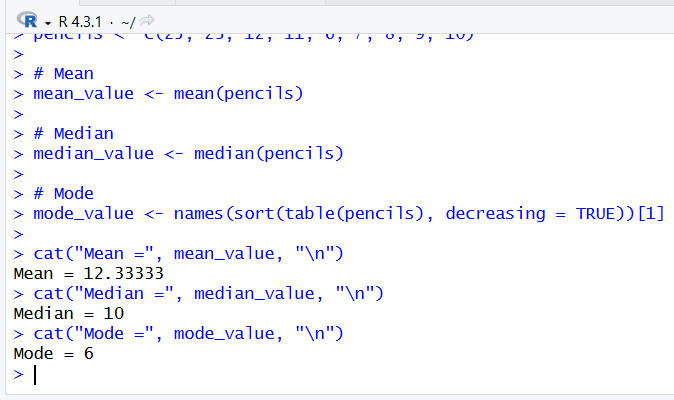
# Mode

mode\_value <- names(sort(table(pencils), decreasing = TRUE))[1]

cat("Mean =", mean\_value, "\n")

cat("Median =", median\_value, "\n")

cat("Mode =", mode\_value, "\n")



22.

# Step 1: Player scores

scores <- c(45, 50, 52, 48, 47, 49, 51, 46, 53, 120)

# Step 2: Draw boxplot

boxplot(scores,

main = "Boxplot of Tennis Players' Scores",

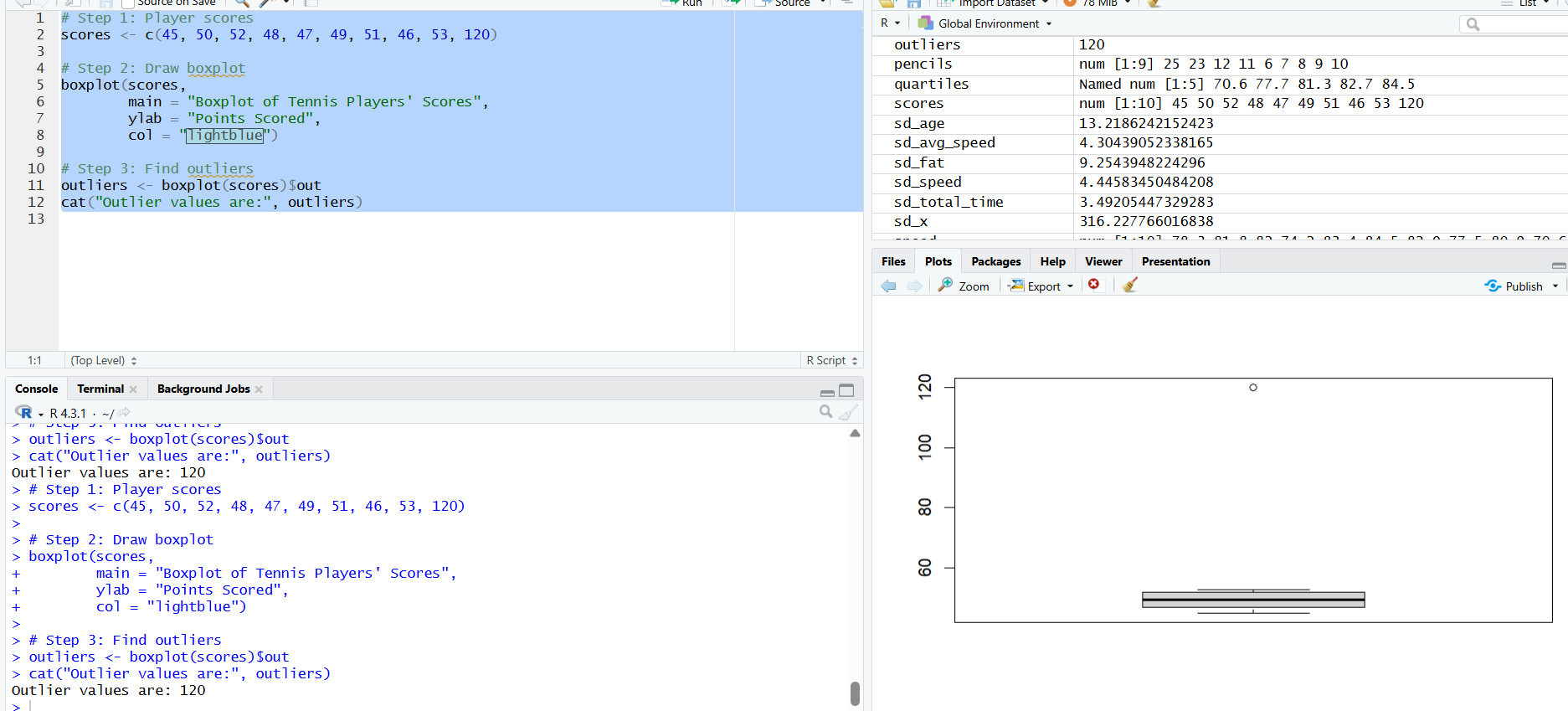
ylab = "Points Scored",

col = "lightblue")

# Step 3: Find outliers

outliers <- boxplot(scores)$out

cat("Outlier values are:", outliers)



**23.EmployeeID,Gender,Age,Salary,Credit**

**111,Male,28,150000,39**

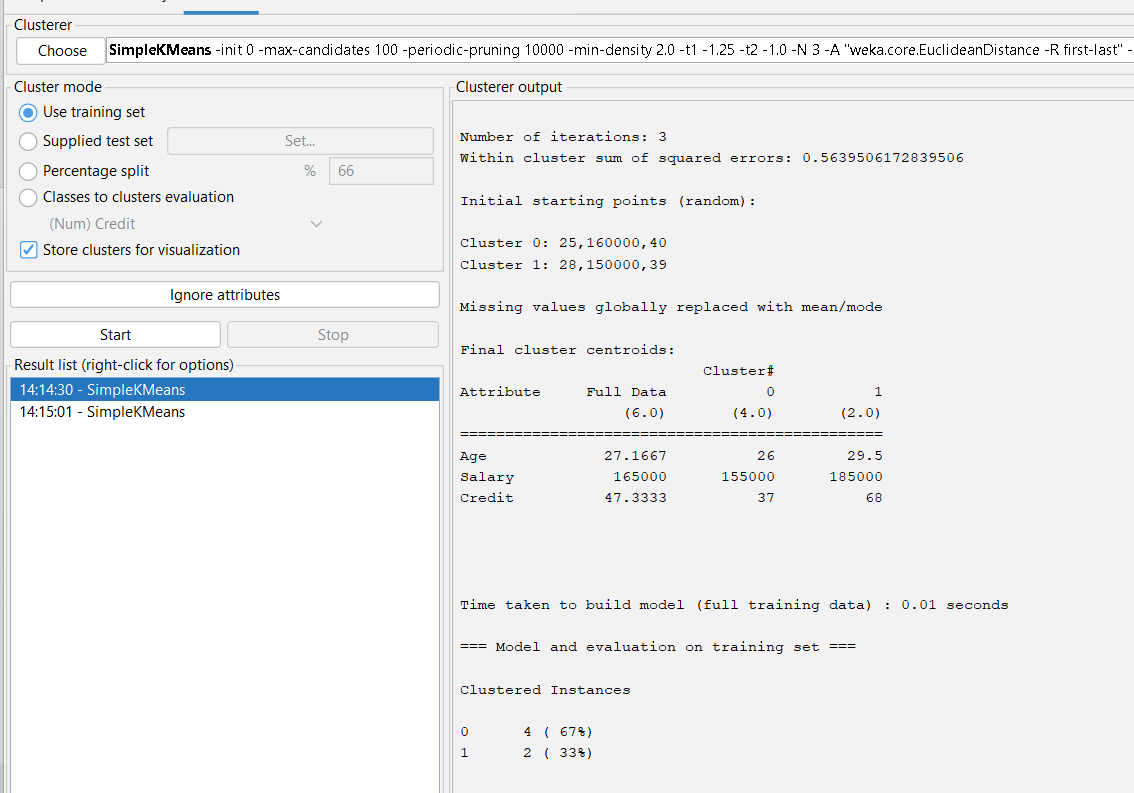
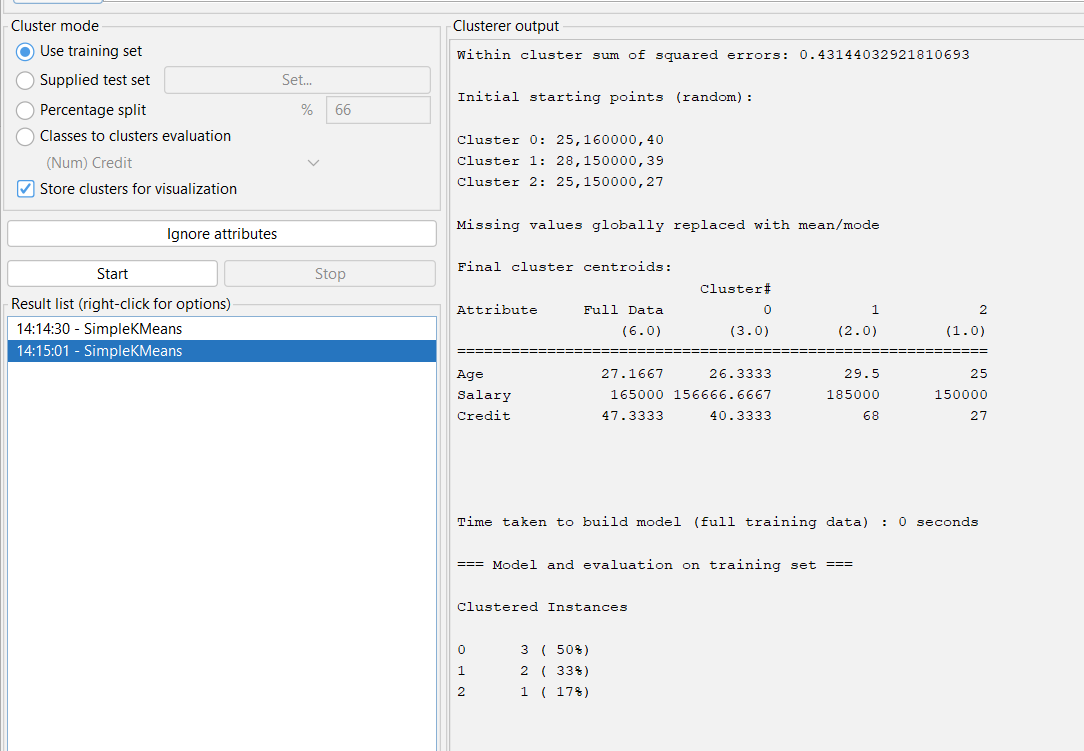
**222,Male,25,150000,27**

**333,Female,26,160000,42**

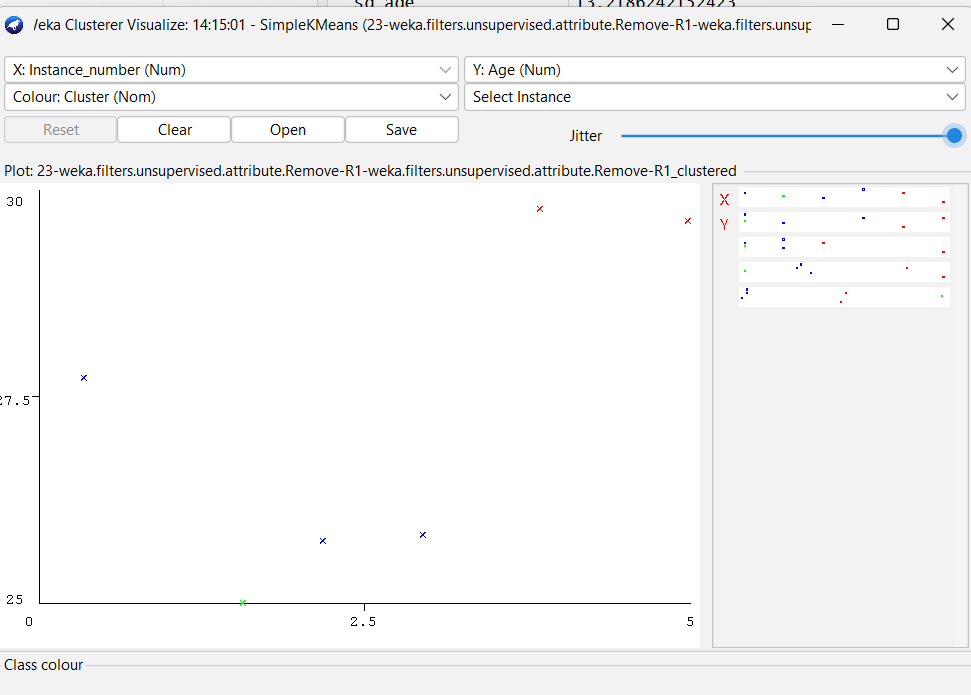
**444,Female,25,160000,40**

**555,Female,30,170000,64**

**666,Male,29,200000,72**

****

**Cluster=2 cluster=3**

**Visual cluster assignments**

**24.**

**@relation weather**

**@attribute outlook {sunny,overcast,rainy}**

**@attribute temperature {hot,mild,cool}**

**@attribute humidity {high,normal}**

**@attribute windy {true,false}**

**@attribute play {yes,no}**

**@data**

**sunny,hot,high,false,no**

**sunny,hot,high,true,no**

**overcast,hot,high,false,yes**

**rainy,mild,high,false,yes**

**rainy,cool,normal,false,yes**

**rainy,cool,normal,true,no**

**overcast,cool,normal,true,yes**

**sunny,mild,high,false,no**

**sunny,cool,normal,false,yes**

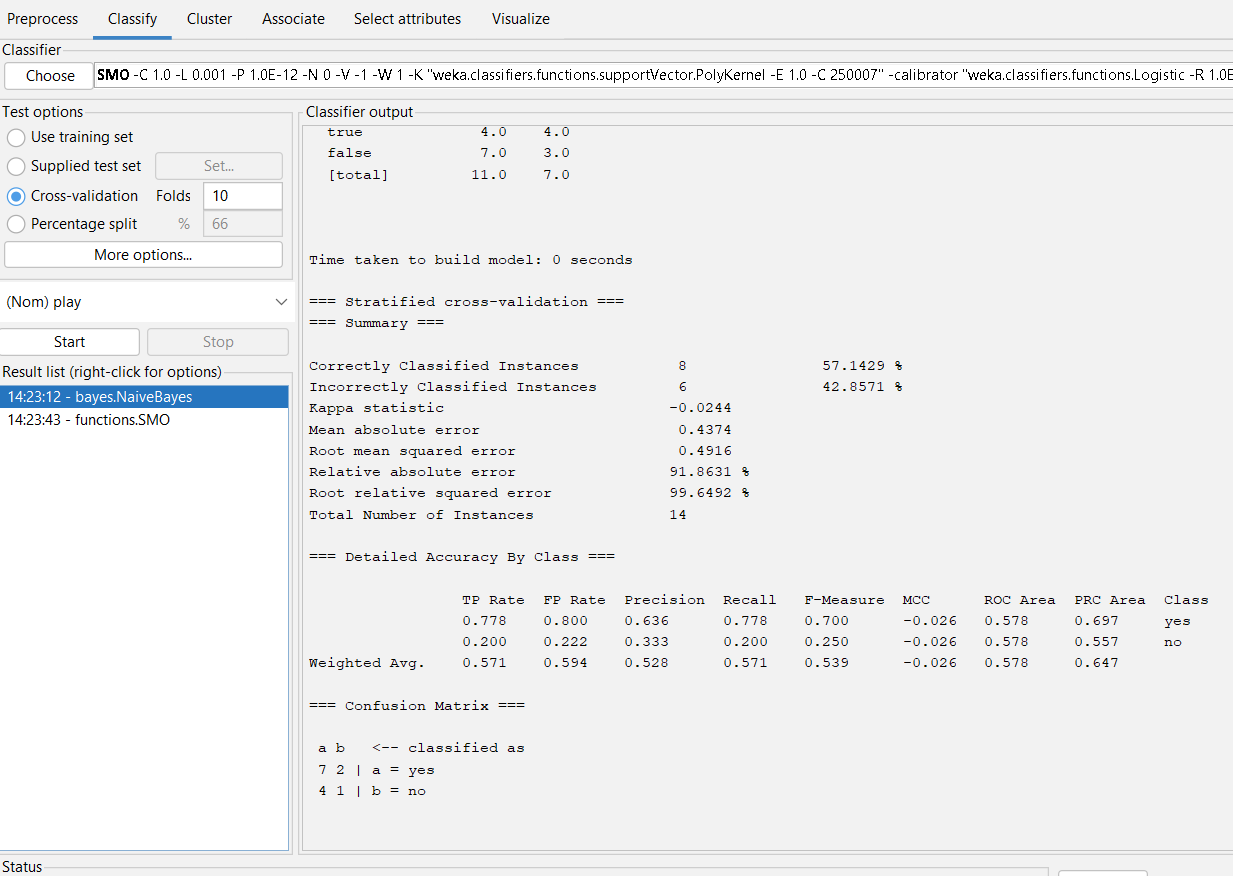
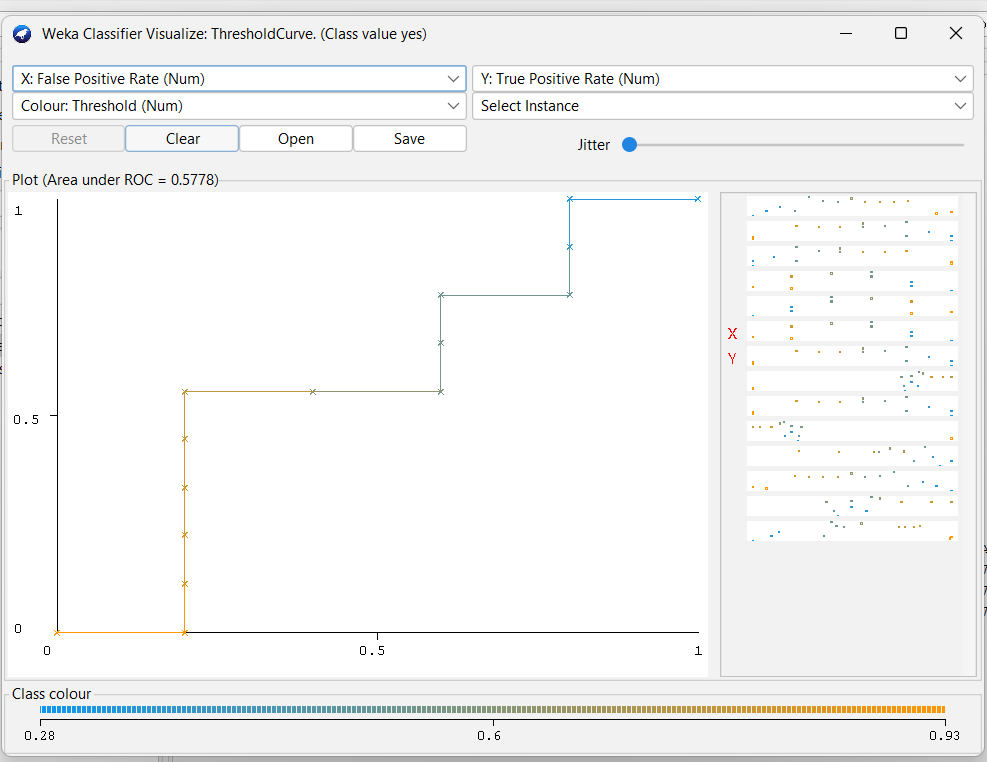
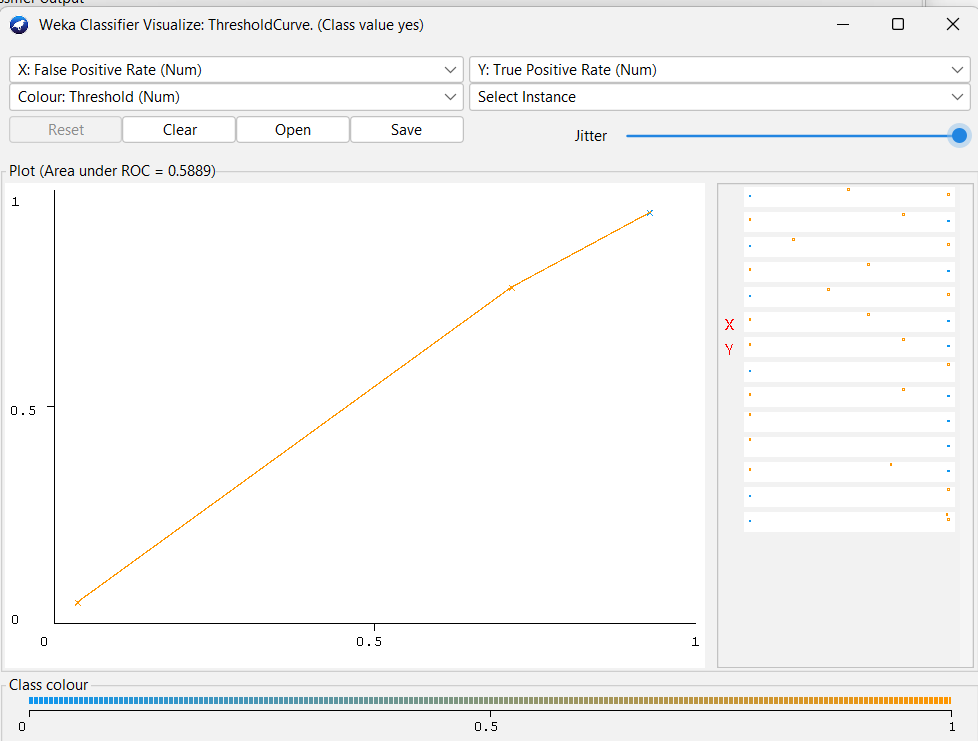
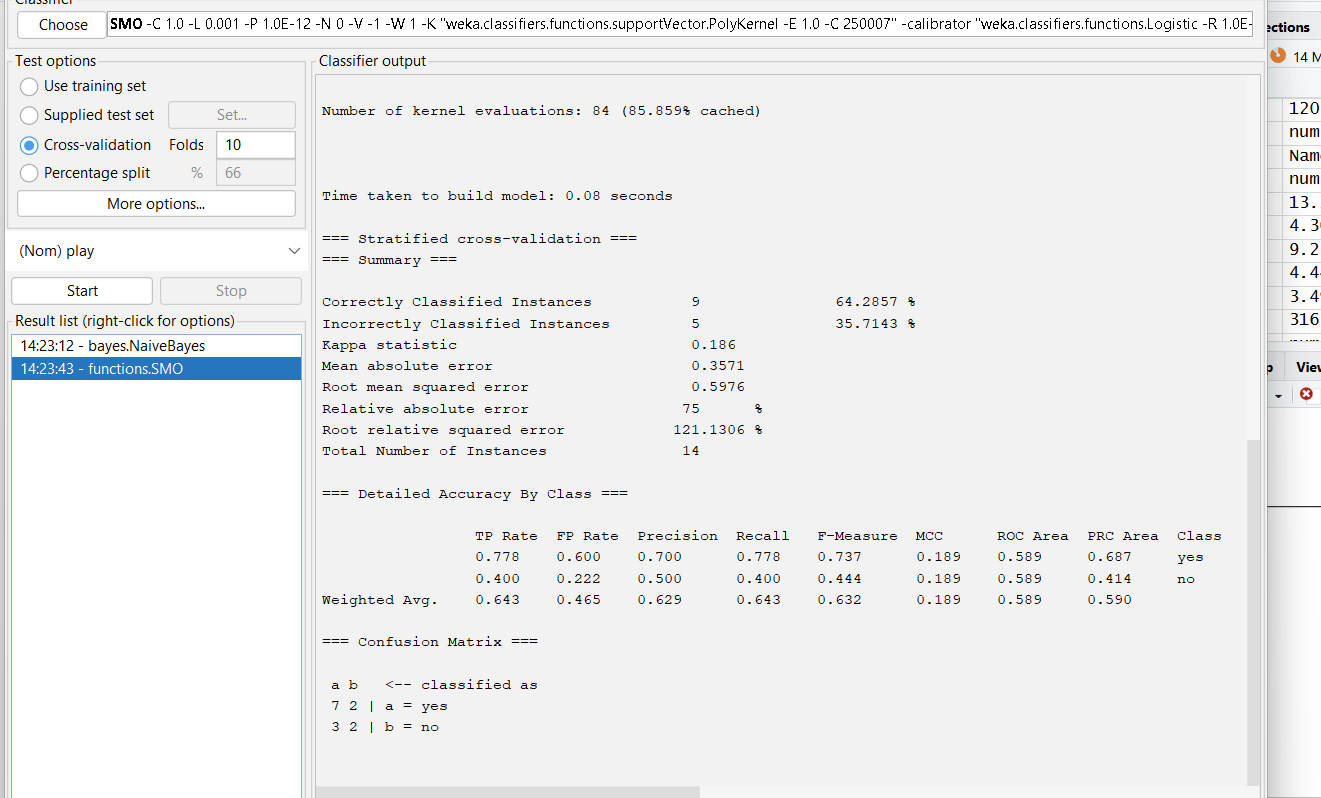
**rainy,mild,normal,false,yes**

**sunny,mild,normal,true,yes**

**overcast,mild,high,true,yes**

**overcast,hot,normal,false,yes**

**rainy,mild,high,true,no**

****

**25.**

**# Step 1: Create vector**

**veg\_status <- c("yes","yes","yes","no","yes","no","no","yes","yes","yes")**

**# Step 2: Count values**

**count\_table <- table(veg\_status)**

**# Step 3: Extract counts**

**veg\_count <- count\_table["yes"]**

**nonveg\_count <- count\_table["no"]**

**# Step 4: Display results**

**cat("Number of Vegetarians =", veg\_count, "\n")**

**cat("Number of Non-Vegetarians =", nonveg\_count, "\n")**

**# Step 5: Compare**

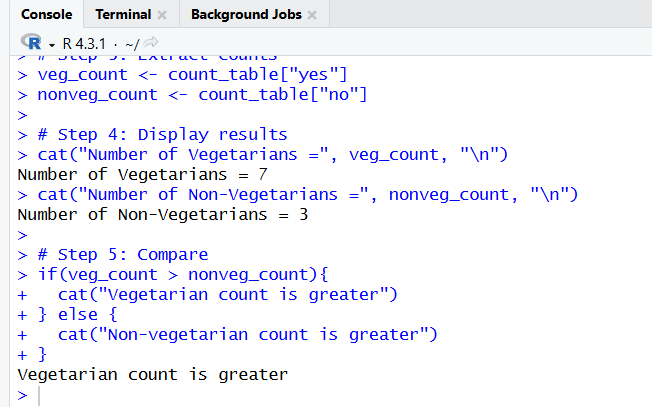
**if(veg\_count > nonveg\_count){**

**cat("Vegetarian count is greater")**

**} else {**

**cat("Non-vegetarian count is greater")**

**}**

****

**26, code:**

x <- c(4, 1, 5, 7, 10, 2, 50, 25, 90, 36)

y <- c(12, 5, 13, 19, 31, 7, 153, 72, 275, 110)

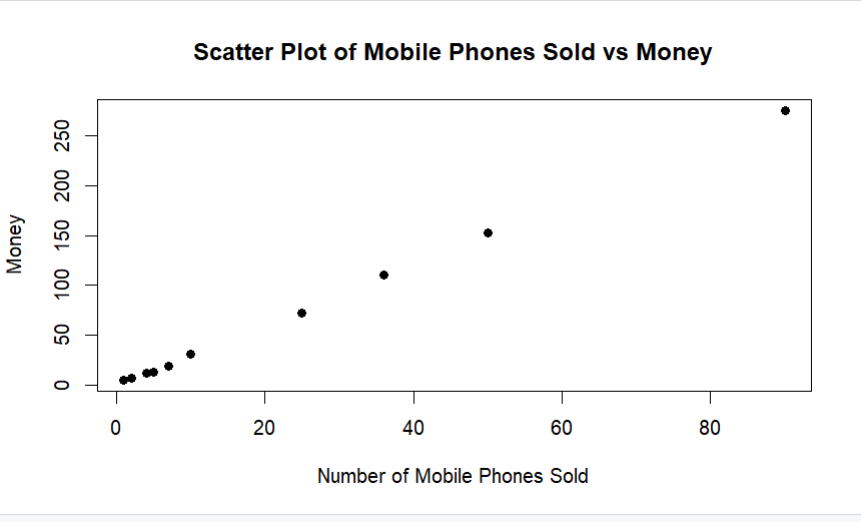
plot(x, y,

main = "Scatter Plot of Mobile Phones Sold vs Money",

xlab = "Number of Mobile Phones Sold",

ylab = "Money",

pch = 19)

****

**27.@relation market**

**@attribute Bread {yes,no}**

**@attribute Cheese {yes,no}**

**@attribute Egg {yes,no}**

**@attribute Juice {yes,no}**

**@attribute Milk {yes,no}**

**@attribute Yogurt {yes,no}**

**@data**

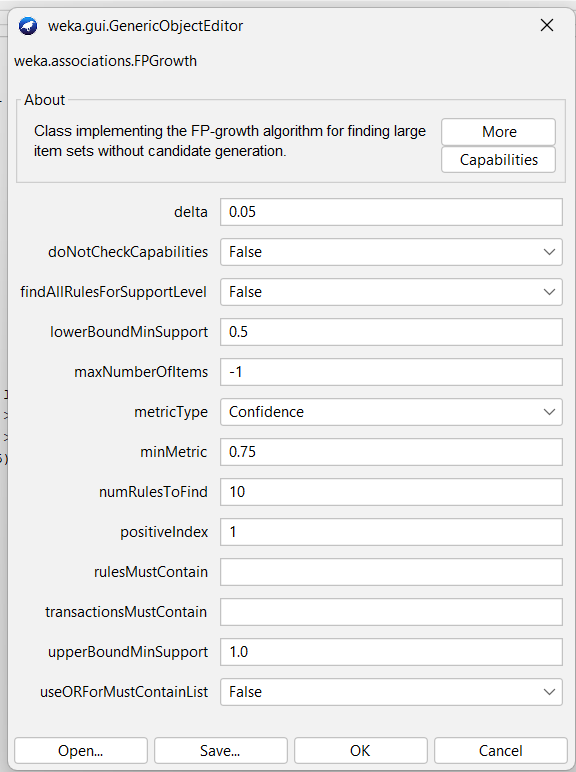
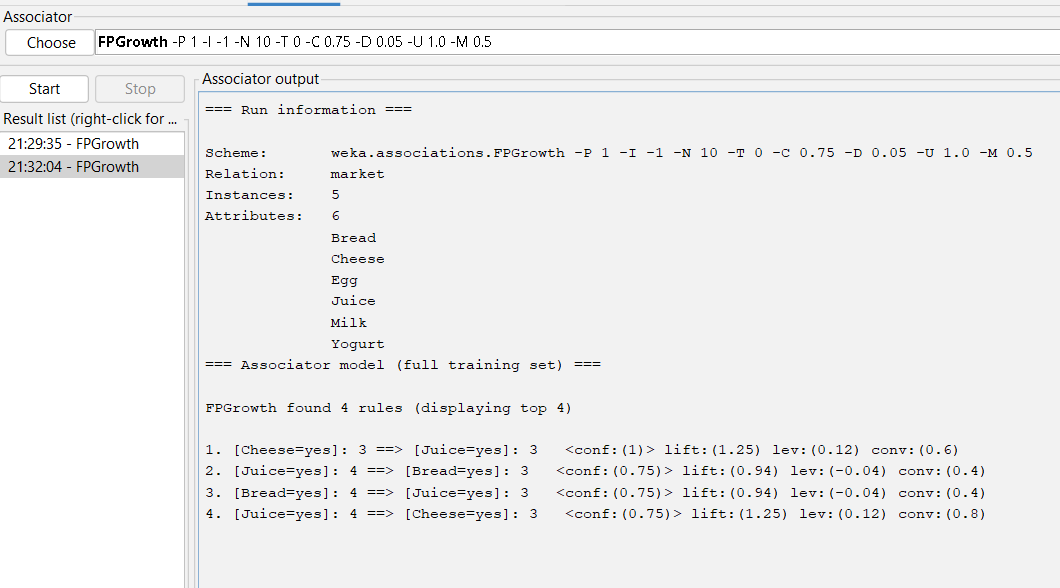
**yes,yes,yes,yes,no,no**

**yes,yes,no,yes,no,no**

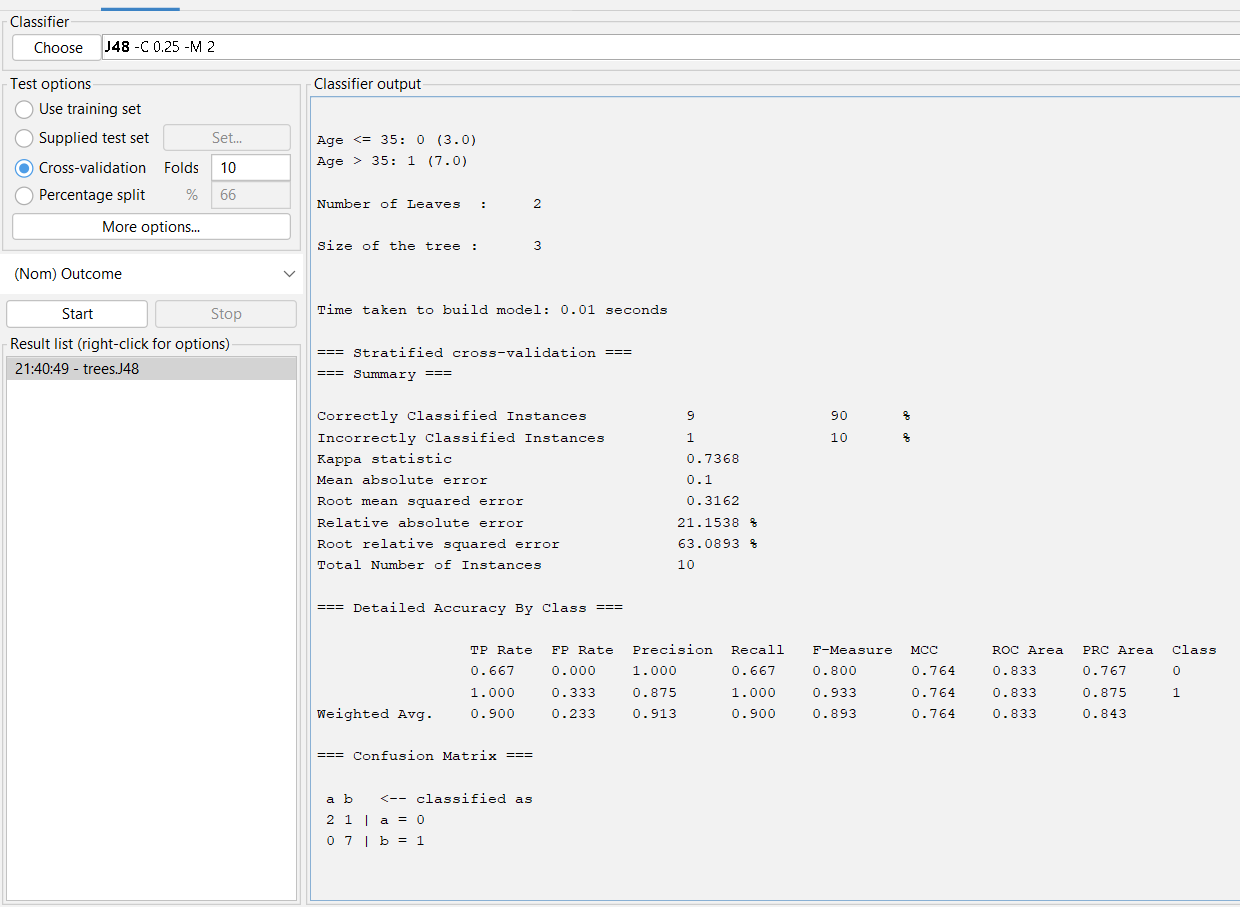
**yes,no,no,no,yes,yes**

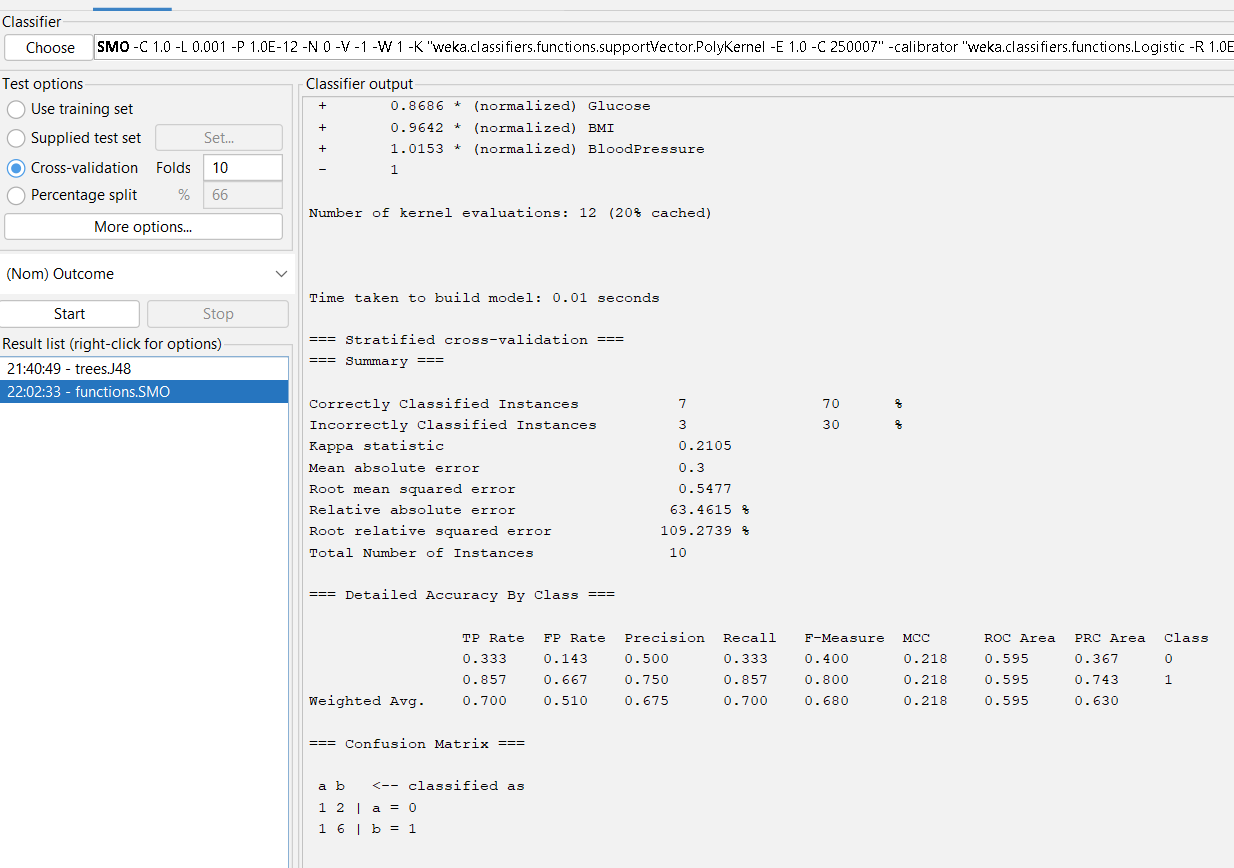
**yes,no,no,yes,yes,no**

**no,yes,no,yes,yes,no**

** **

**28.**

** 28**

****

****

**29**. marks <- c(55,60,71,63,55,65,50,55,58,59,61,63,65,67,71,72,75)

# Equal Frequency

bins\_eq\_freq <- cut(marks,

breaks = quantile(marks, probs = seq(0,1, length=4)),

include.lowest = TRUE)

hist(marks, breaks = quantile(marks, probs = seq(0,1, length=4)),

main="Equal Frequency Binning", xlab="Marks")

# Equal Width

bins\_eq\_width <- cut(marks, breaks=3)

hist(marks, breaks=3,

main="Equal Width Binning", xlab="Marks")

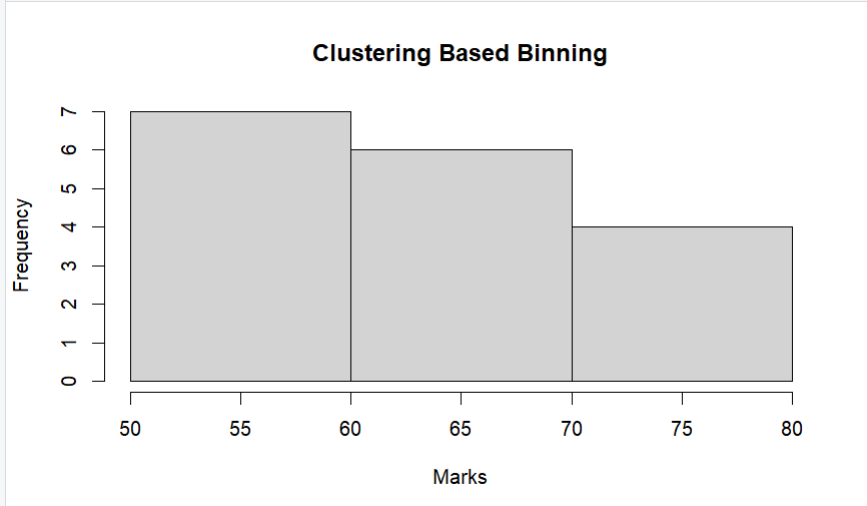
# Clustering

set.seed(1)

cl <- kmeans(marks, centers=3)

hist(marks, breaks=3,

main="Clustering Based Binning", xlab="Marks")



30. @relation gender

@attribute height numeric

@attribute weight numeric

@attribute class {Male,Female}

@data

185,85,Male

190,90,Male

182,78,Male

170,85,Male

175,88,Male

160,55,Female

165,60,Female

170,65,Female

155,50,Female

180,70,Female

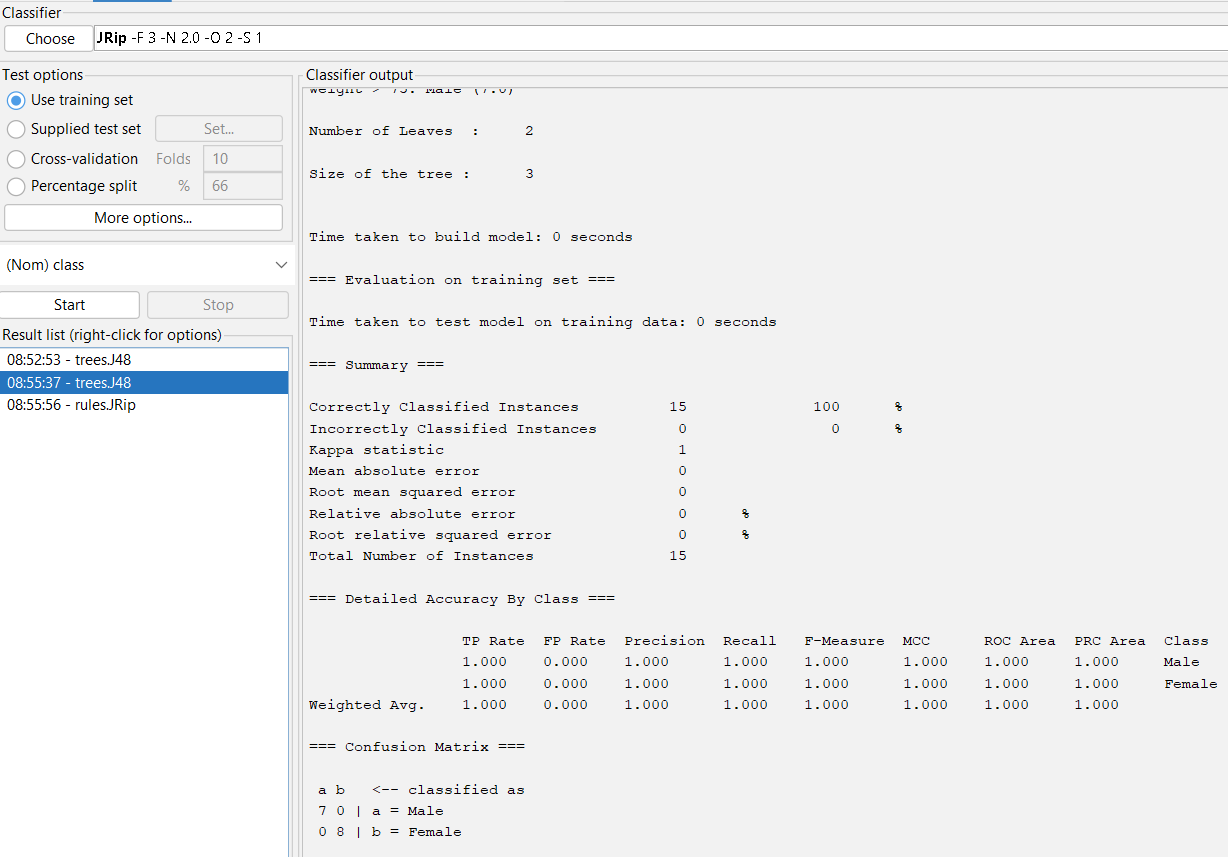
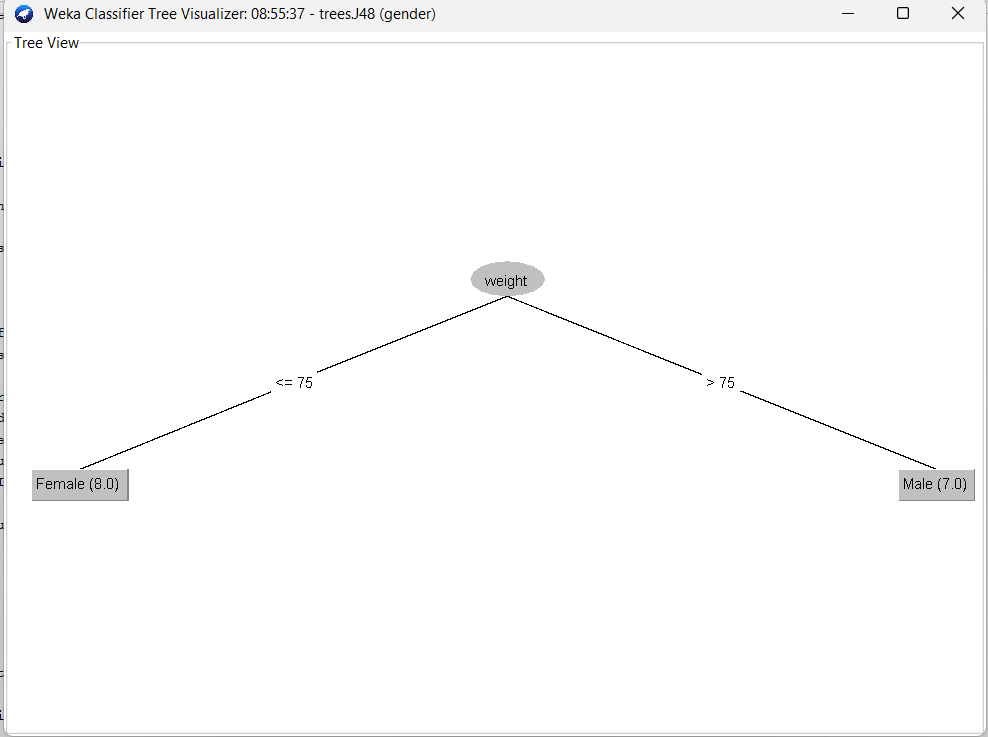
178,75,Female

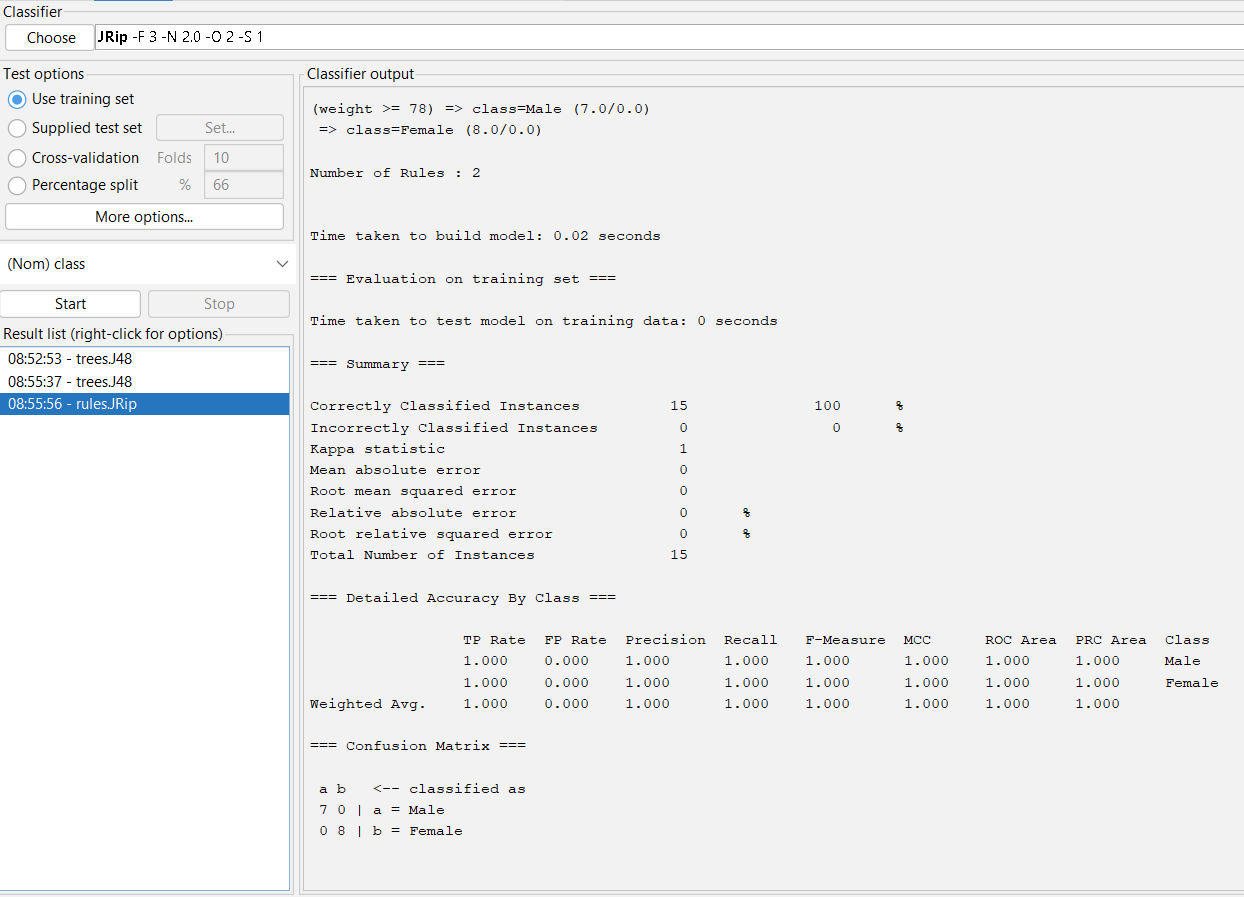
172,82,Male

168,68,Female

174,90,Male

158,52,Female

 a 



31.@relation electronics

@attribute SONY {t,f}

@attribute BPL {t,f}

@attribute LG {t,f}

@attribute SAMSUNG {t,f}

@attribute ONIDA {t,f}

@data

t,t,t,f,f % T1: SONY, BPL, LG

f,t,f,t,f % T2: BPL, SAMSUNG

f,t,f,f,t % T3: BPL, ONIDA

t,t,f,t,f % T4: SONY, BPL, SAMSUNG

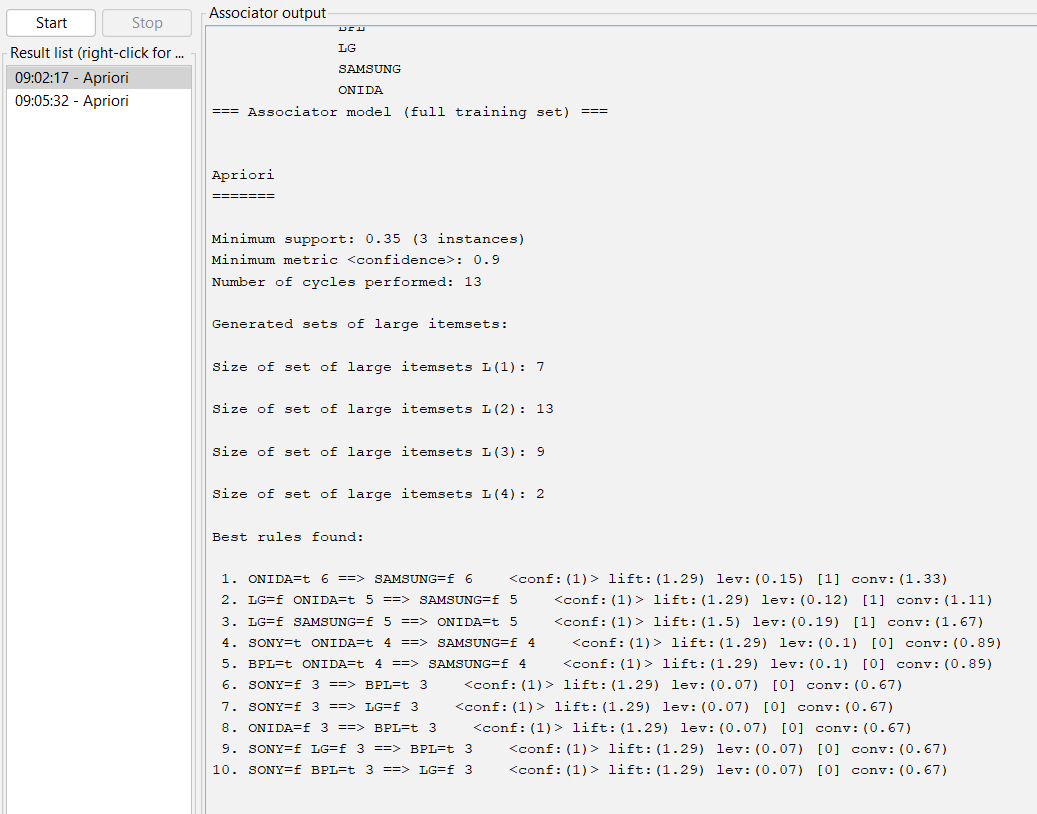
t,f,f,f,t % T5: SONY, ONIDA

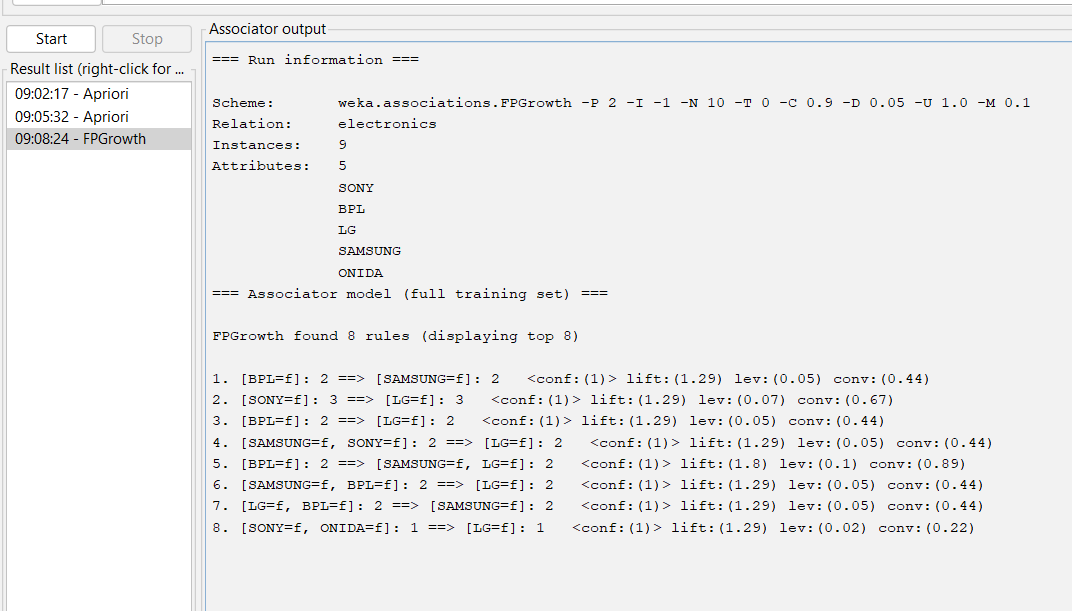
f,t,f,f,t % T6: BPL, ONIDA

t,f,f,f,t % T7: SONY, ONIDA

t,t,t,f,t % T8: SONY, BPL, ONIDA, LG

t,t,f,f,t % T9: SONY, BPL, ONIDA





32. # Input data

x <- c(100, 70, 60, 90, 90)

cat("Original Data:\n")

print(x)

min\_val <- min(x)

max\_val <- max(x)

minmax\_norm <- (x - min\_val) / (max\_val - min\_val)

cat("\n(a) Min-Max Normalization (0 to 1):\n")

print(minmax\_norm)

mean\_x <- mean(x)

sd\_x <- sd(x)

zscore\_norm <- (x - mean\_x) / sd\_x

cat("\n(b) Z-score Normalization (using SD):\n")

print(zscore\_norm)

mad\_val <- mean(abs(x - mean\_x))

zscore\_mad <- (x - mean\_x) / mad\_val

cat("\n(c) Z-score using Mean Absolute Deviation:\n")

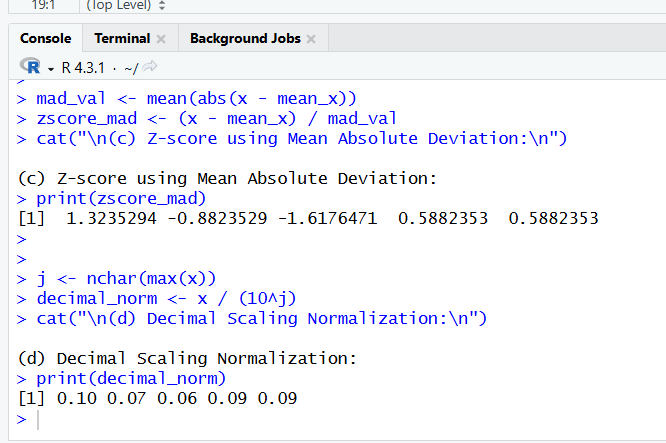
print(zscore\_mad)

j <- nchar(max(x))

decimal\_norm <- x / (10^j)

cat("\n(d) Decimal Scaling Normalization:\n")

print(decimal\_norm)



33.

# Data

avg\_speed <- c(78, 81, 82, 74, 83, 82, 77, 80, 70)

total\_time <- c(39, 37, 36, 42, 35, 36, 40, 38, 46)

# Mean

mean(avg\_speed)

mean(total\_time)

# Standard Deviation

sd\_avg\_speed <- sd(avg\_speed)

sd\_total\_time <- sd(total\_time)

cat("Standard Deviation of Avg Speed =", sd\_avg\_speed, "\n")

cat("Standard Deviation of Total Time =", sd\_total\_time, "\n")

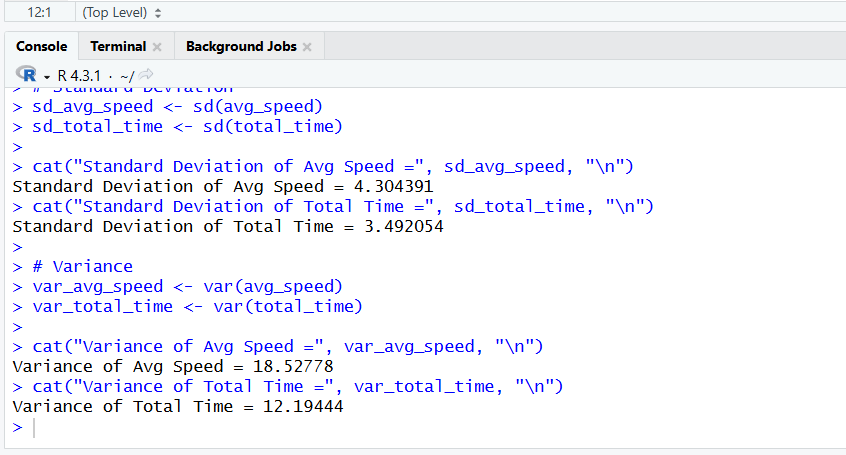
# Variance

var\_avg\_speed <- var(avg\_speed)

var\_total\_time <- var(total\_time)

cat("Variance of Avg Speed =", var\_avg\_speed, "\n")

cat("Variance of Total Time =", var\_total\_time, "\n")



34.

# Create dataset

age <- c(1, 6, 11, 16, 21)

tumor\_size <- c(2.1, 2.8, 3.5, 4.0, 4.6)

inv\_nodes <- c(1, 3, 5, 7, 9)

data <- data.frame(age, tumor\_size, inv\_nodes)

print(data)

# Histogram

hist(tumor\_size,

main = "Histogram of Tumor Size",

xlab = "Tumor Size (cm)",

ylab = "Frequency",

col = "lightblue")

# Scatter plot

plot(age, tumor\_size,

main = "Scatter Plot of Age vs Tumor Size",

xlab = "Age Group",

ylab = "Tumor Size (cm)",

pch = 19)

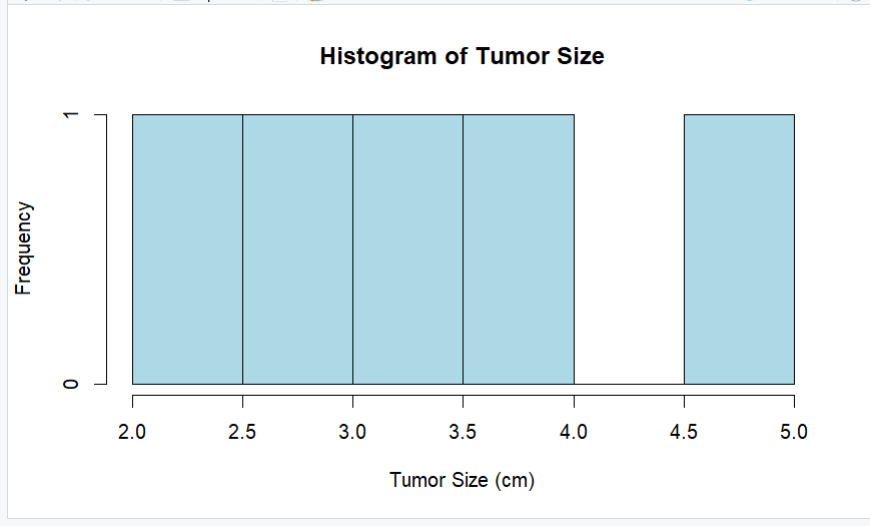
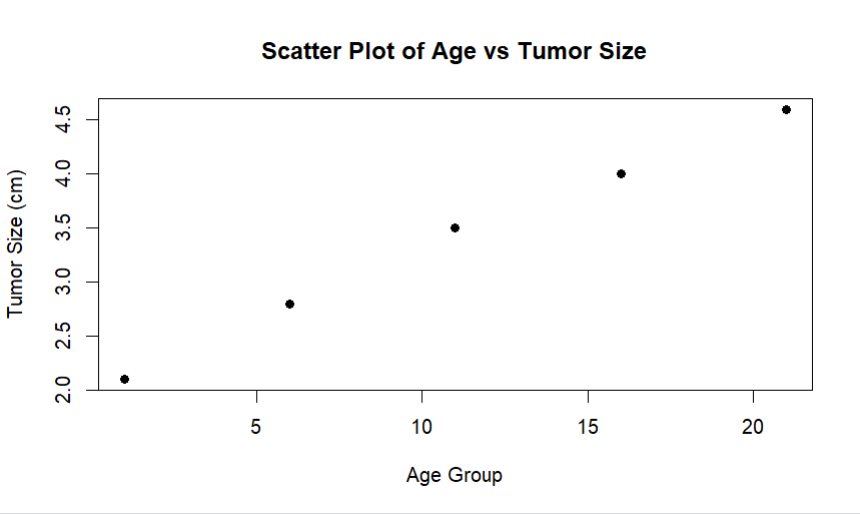
# Box plot

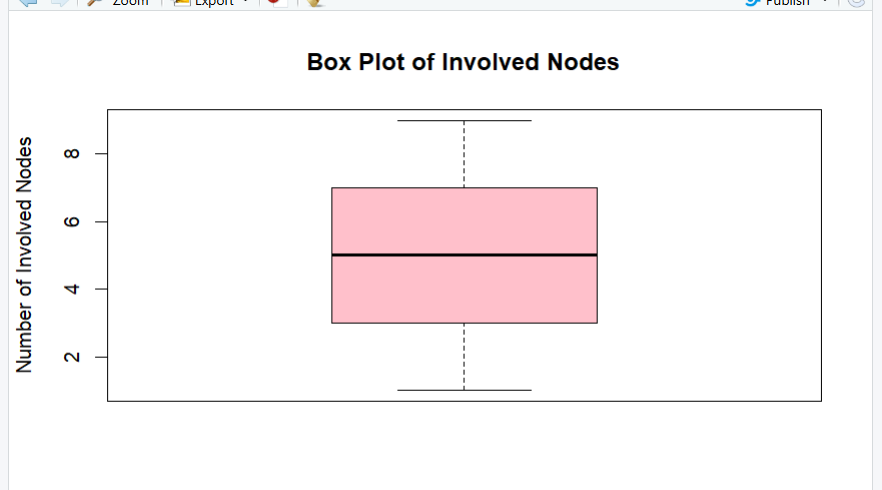
boxplot(inv\_nodes,

main = "Box Plot of Involved Nodes",

ylab = "Number of Involved Nodes",

col = "pink")



35.

@relation BoyWhoCriedWolf

@attribute AlarmRaised {Yes, No}

@attribute WolfPresent {Yes, No}

@attribute VillagersResponse {Believe, Ignore}

@attribute Outcome {Safe, Loss}

@data

Yes, No, Believe, Safe

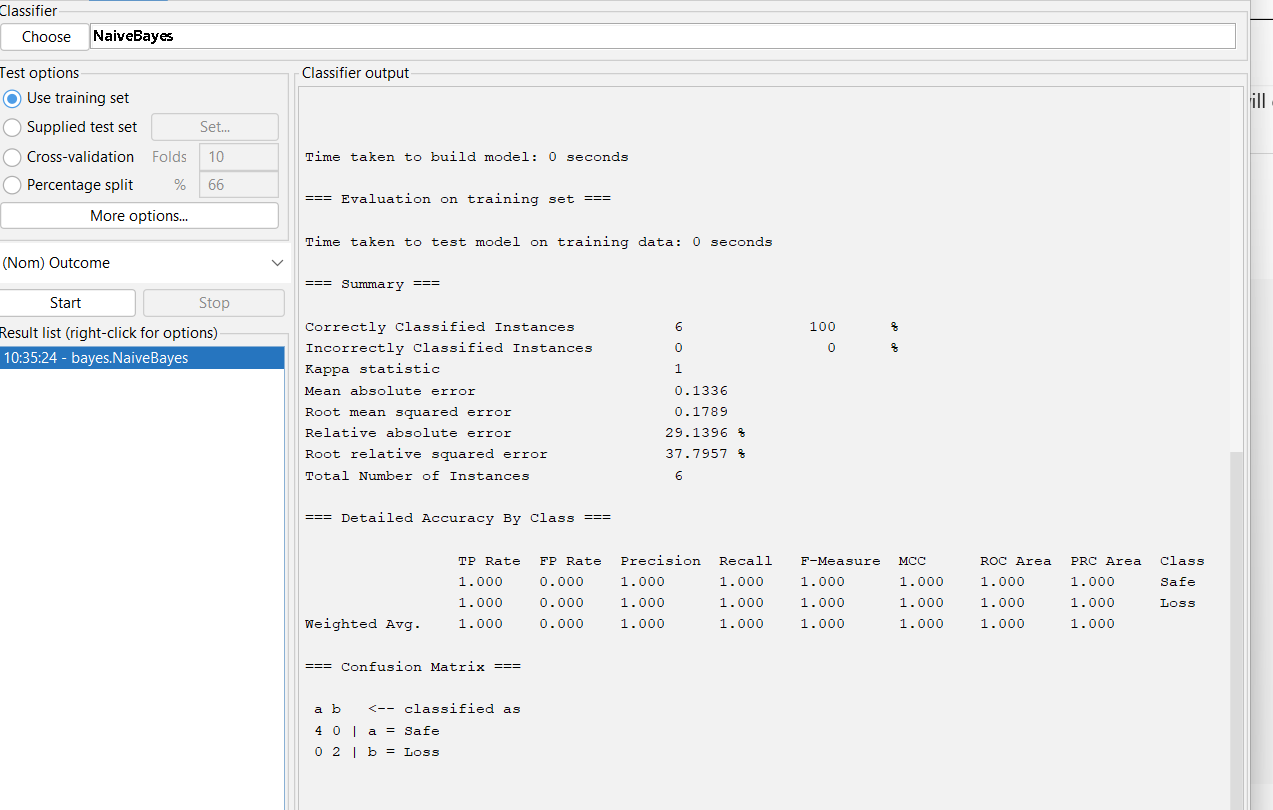
Yes, No, Believe, Safe

Yes, No, Believe, Safe

Yes, Yes, Ignore, Loss

No, Yes, Ignore, Loss

No, No, Ignore, Safe



36.@relation playtennis

@attribute Outlook {Sunny,Overcast,Rain}

@attribute Temperature {Hot,Mild,Cool}

@attribute Humidity {High,Normal}

@attribute Wind {Weak,Strong}

@attribute PlayTennis {Yes,No}

@data

Sunny,Hot,High,Weak,No

Sunny,Hot,High,Strong,No

Overcast,Hot,High,Weak,Yes

Rain,Mild,High,Weak,Yes

Rain,Cool,Normal,Weak,Yes

Rain,Cool,Normal,Strong,No

Overcast,Cool,Normal,Strong,Yes

Sunny,Mild,High,Weak,No

Sunny,Cool,Normal,Weak,Yes

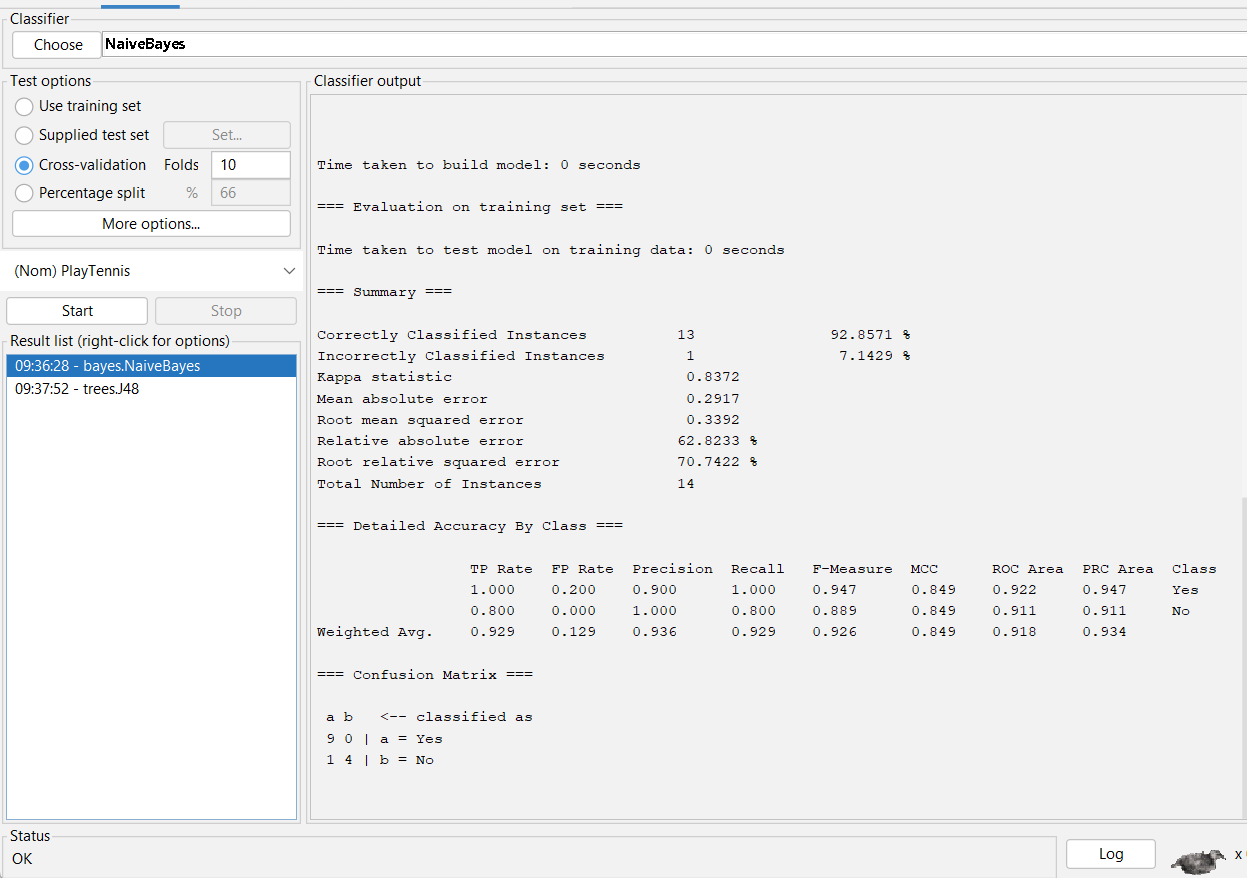
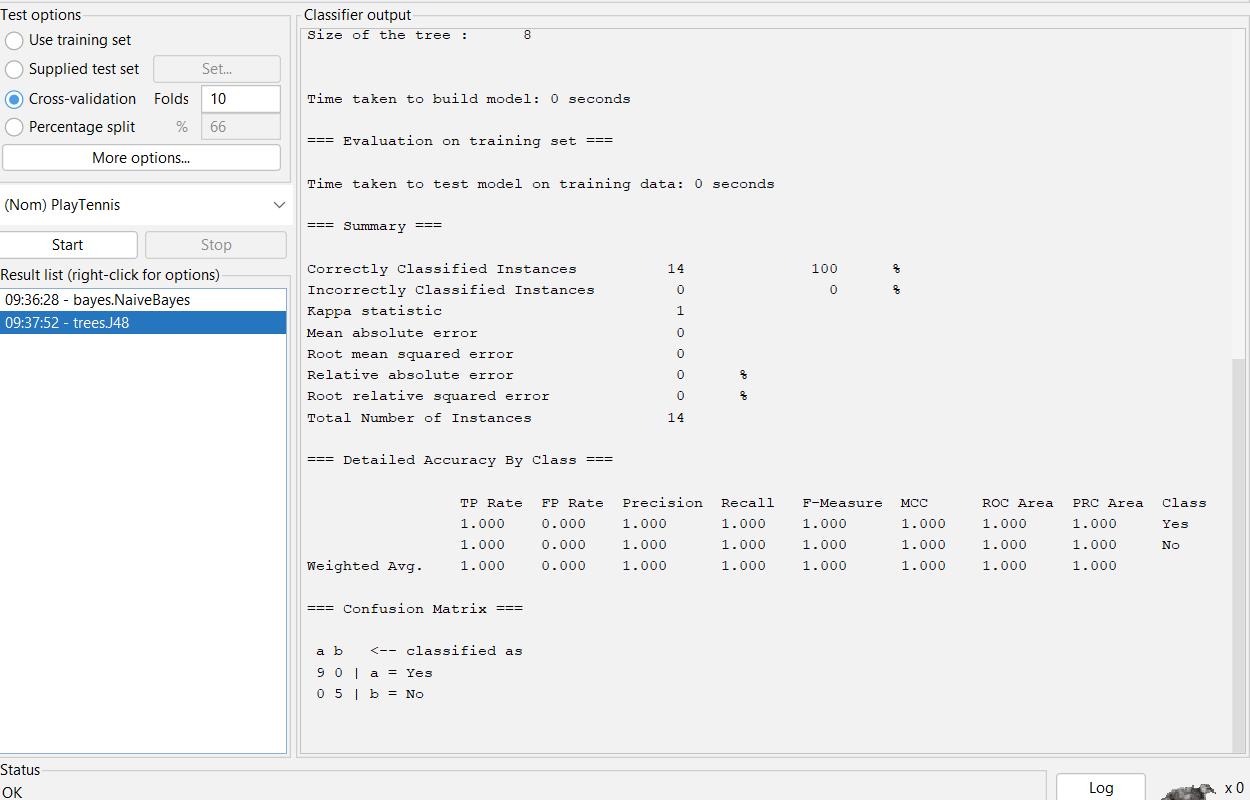
Rain,Mild,Normal,Weak,Yes

Sunny,Mild,Normal,Strong,Yes

Overcast,Mild,High,Strong,Yes

Overcast,Hot,Normal,Weak,Yes

Rain,Mild,High,Strong,No

37.

age <- c(30,57,68,96,39,40,20,19,42,12,

25,25,65,35,30,23,23,35,45,85)

cat("Age Data:\n")

print(age)

mean\_age <- mean(age)

cat("\nMean of Age =", mean\_age, "\n")

# -------- PART (b): IQR and SD of Speed --------

speed <- c(78.3,81.8,82,74.2,83.4,84.5,82.9,77.5,80.9,70.6)

cat("\nSpeed Data:\n")

print(speed)

# Quartiles

quartiles <- quantile(speed)

cat("\nQuartiles:\n")

print(quartiles)

# Interquartile Range

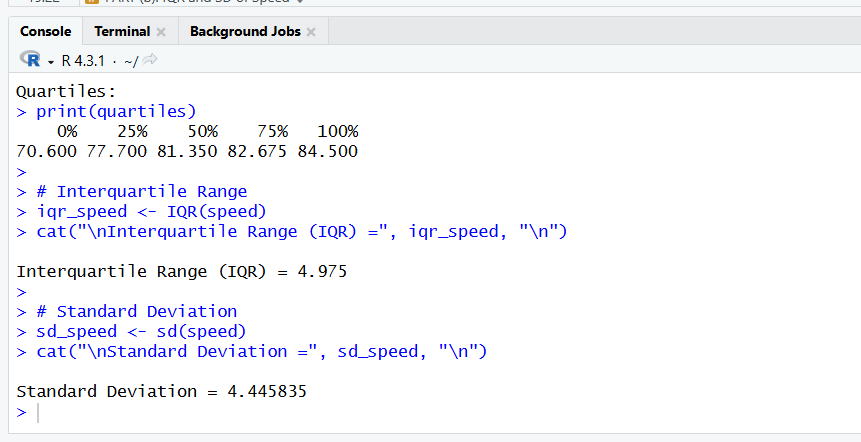
iqr\_speed <- IQR(speed)

cat("\nInterquartile Range (IQR) =", iqr\_speed, "\n")

# Standard Deviation

sd\_speed <- sd(speed)

cat("\nStandard Deviation =", sd\_speed, "\n")



38.

# -------- PART (a) --------

min\_val <- 50000

max\_val <- 100000

v <- 80000

v\_norm <- (v - min\_val) / (max\_val - min\_val)

cat("Part (a) Min-Max normalized value =", v\_norm, "\n")

# -------- PART (b) --------

x <- c(200, 300, 400, 600, 1000)

cat("\nOriginal Data:\n")

print(x)

# Min-Max Normalization (0 to 1)

min\_x <- min(x)

max\_x <- max(x)

minmax\_norm <- (x - min\_x) / (max\_x - min\_x)

cat("\nMin-Max Normalization (0 to 1):\n")

print(minmax\_norm)

# Z-score Normalization

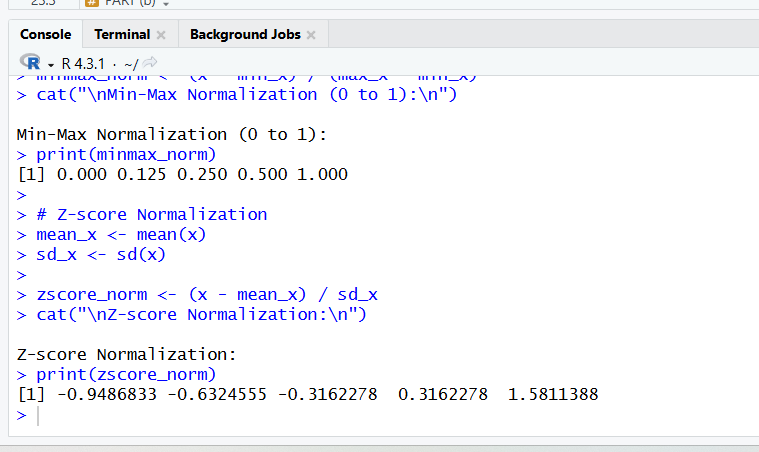
mean\_x <- mean(x)

sd\_x <- sd(x)

zscore\_norm <- (x - mean\_x) / sd\_x

cat("\nZ-score Normalization:\n")

print(zscore\_norm)



39.

@relation shopping

@attribute M {t,f}

@attribute O {t,f}

@attribute N {t,f}

@attribute K {t,f}

@attribute E {t,f}

@attribute Y {t,f}

@attribute D {t,f}

@attribute A {t,f}

@attribute U {t,f}

@attribute C {t,f}

@attribute I {t,f}

@data

t,t,t,t,t,t,f,f,f,f,f % T100

f,t,t,t,t,t,t,f,f,f,f % T200

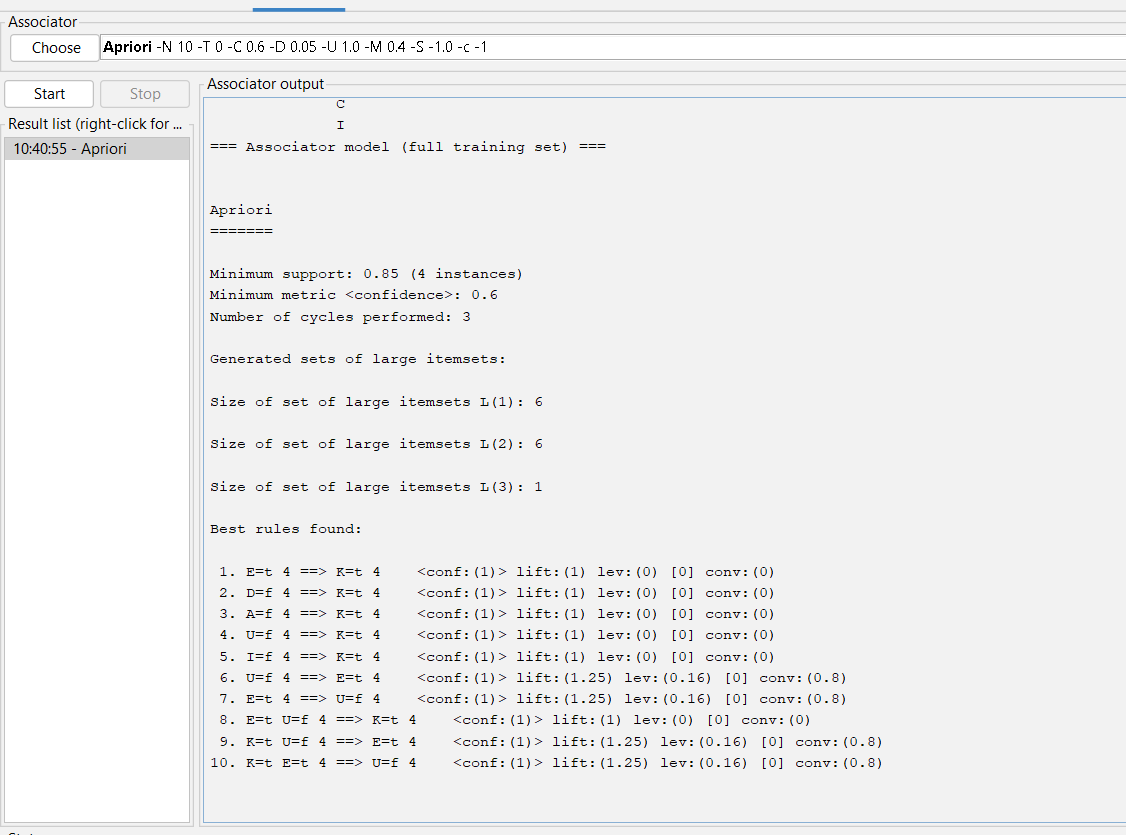
t,f,f,t,t,f,f,t,f,f,f % T300

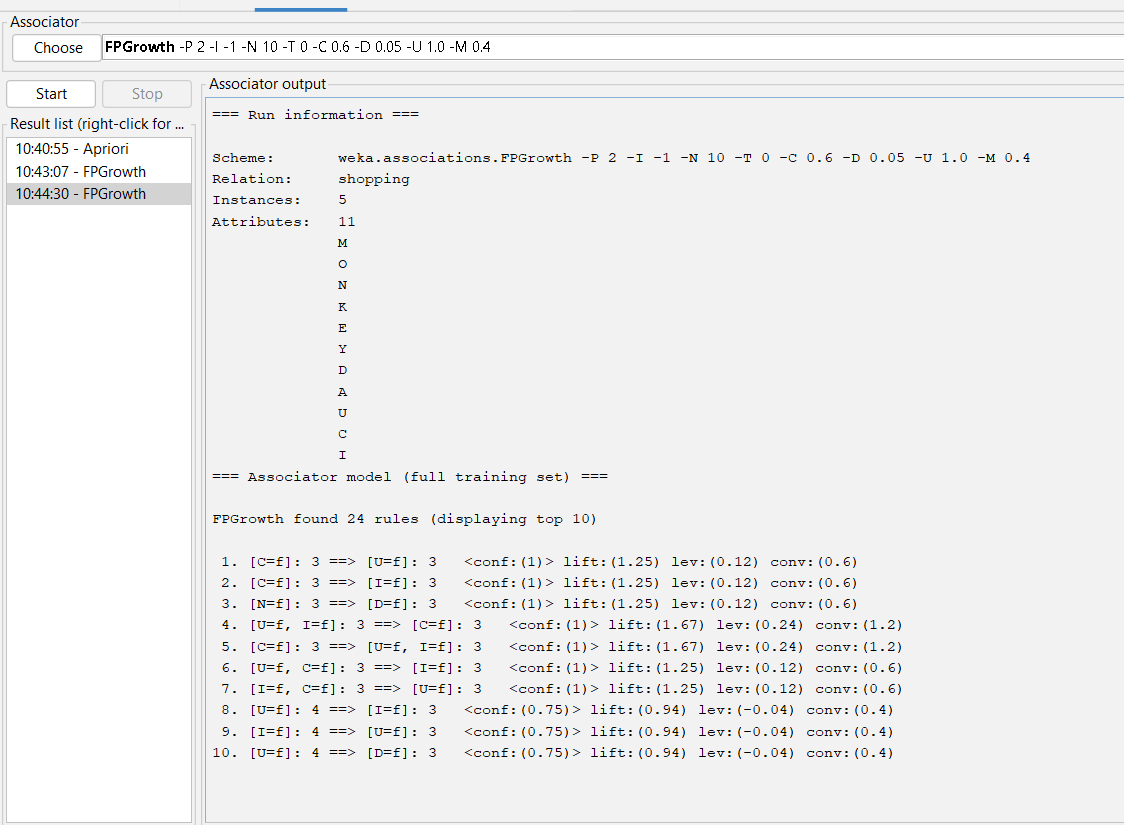
t,f,f,t,f,t,f,f,t,t,f % T400

f,t,f,t,t,f,f,f,f,t,t % T500

**lowerBoundMinSupport = 0.4**

**minMetric = 0.6**





40.

@relation credit

@attribute MaritalStatus {married,single,divorced}

@attribute Gender {male,female}

@attribute Age {18\_30,30\_50,50\_65,65\_plus}

@attribute Income {10\_25K,25\_50K,50\_65K,65\_100K,100K\_plus}

@attribute Class {Good,Bad}

@data

married,male,30\_50,50\_65K,Good

single,female,18\_30,10\_25K,Bad

married,female,30\_50,65\_100K,Good

divorced,male,50\_65,25\_50K,Bad

single,male,18\_30,10\_25K,Bad

married,male,50\_65,100K\_plus,Good

married,female,65\_plus,65\_100K,Good

single,female,30\_50,25\_50K,Bad

divorced,female,50\_65,50\_65K,Bad

married,male,30\_50,25\_50K,Good

single,male,18\_30,10\_25K,Bad

married,female,50\_65,65\_100K,Good

divorced,male,65\_plus,25\_50K,Bad

married,male,65\_plus,100K\_plus,Good

