# SLOT-B CSA1687:DATA WAREHOUSING AND DATA MINING FOR MEDICAL APPLICATIONS

## DAY-1

**LIST OF PROGRAMS:**

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

1-5. 200

5-15 450

15-20 300

20-50 1500

50-80 700

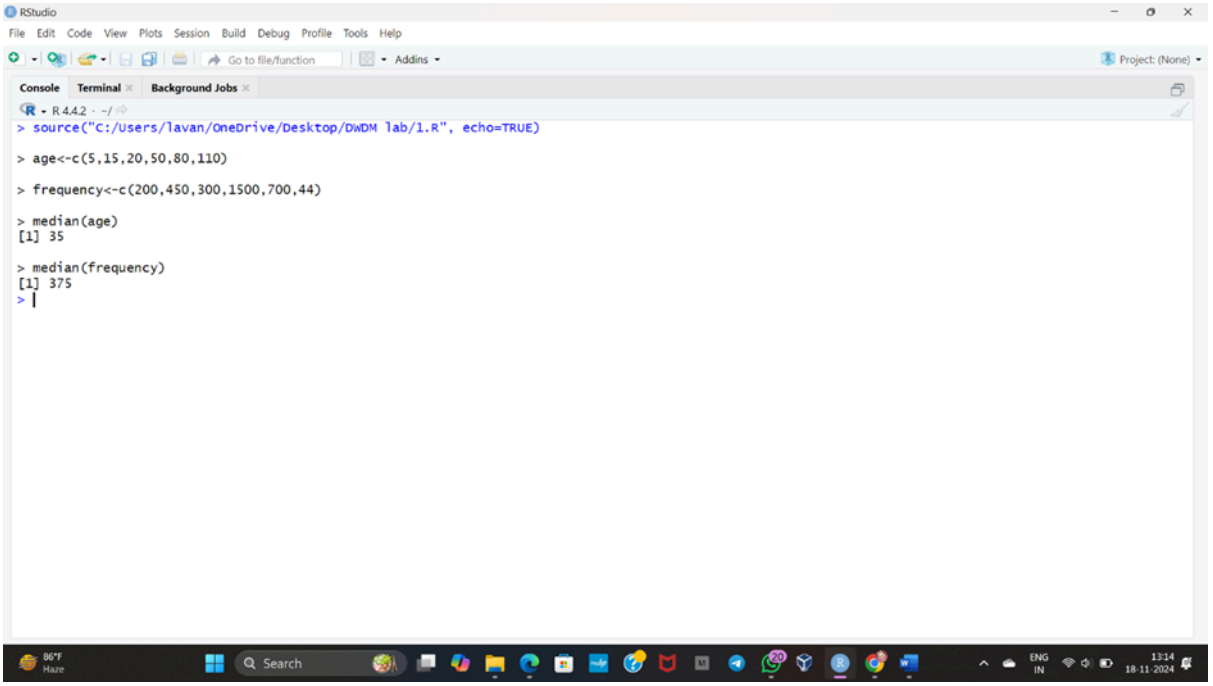
80-110 44

Compute an approximate median value for the data

**INPUT**

```
#age, frequency

age<-c(5,15,20,50,80,110)

frequency<-c(200,450,300,1500,700,44)

median(age)

median(frequency)
```

**OUTPUT**




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

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

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

(c) What is the midrange of the data?

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

**INPUT**

#mean,median,mode,quatile

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)

median(age)

mode_age<-names(table(age))[table(age)==max(table(age))]

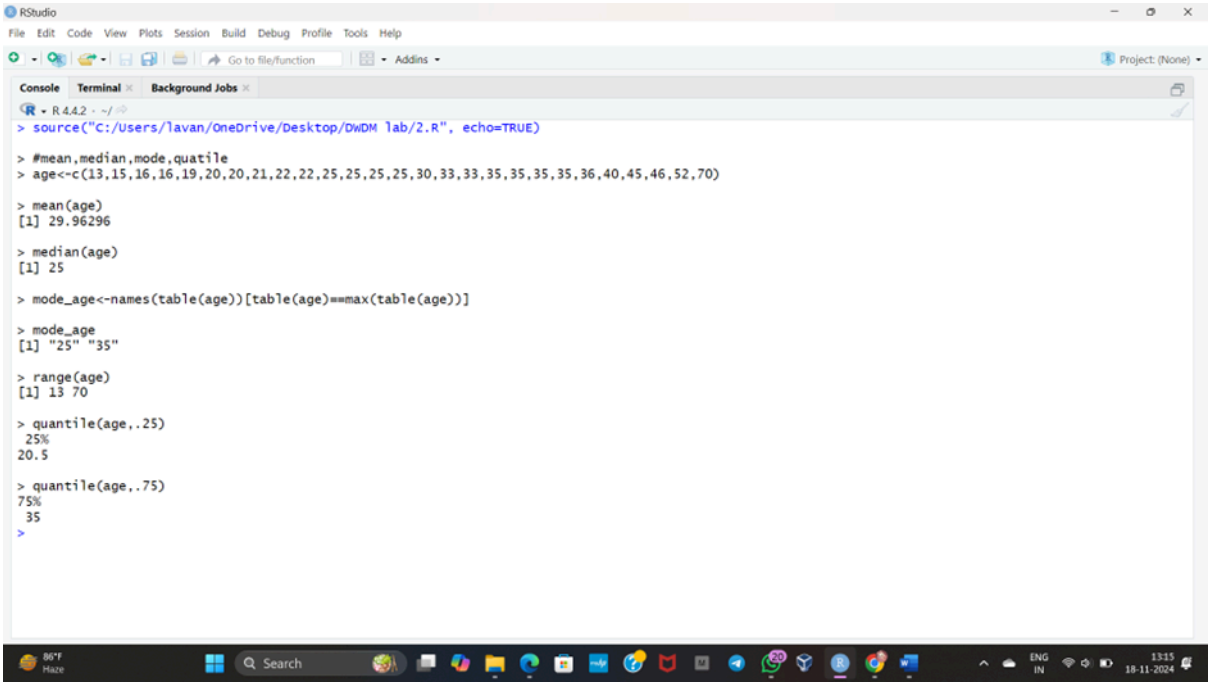
mode_age

range(age)

quantile(age,.25)

quantile(age,.75)

**OUTPUT**




**3.**Data Preprocessing :Reduction and Transformation

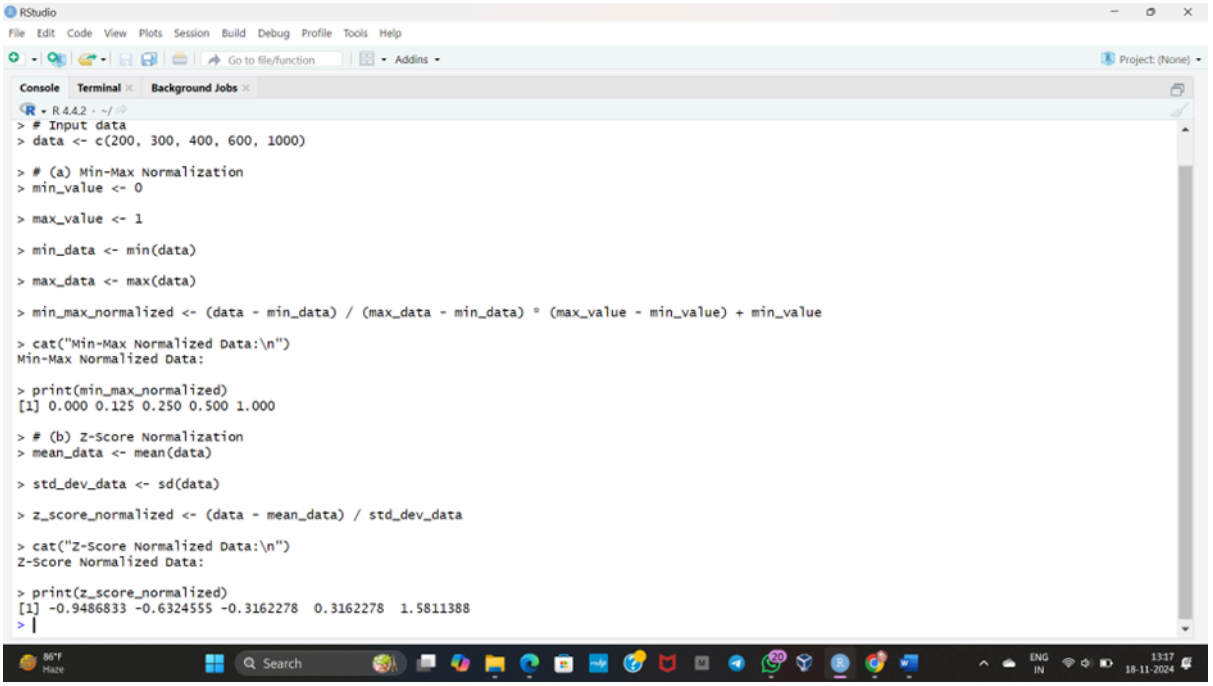
Use the two methods below to normalize the following group of data: 200, 300, 400, 600, 1000 (a) min-max normalization by setting min = 0 and max = 1 (b) z-score normalization

**INPUT**

# Input data

```
data <- c(200, 300, 400, 600, 1000)

# (a) Min-Max Normalization
min_value <- 0
max_value <- 1

min_data <- min(data)
max_data <- max(data)

min_max_normalized <- (data - min_data) / (max_data - min_data) *
(max_value - min_value) + min_value
cat("Min-Max Normalized Data:\n")
print(min_max_normalized)

# (b) Z-Score Normalization
mean_data <- mean(data)
std_dev_data <- sd(data)

z_score_normalized <- (data - mean_data) / std_dev_data
cat("Z-Score Normalized Data:\n")
print(z_score_normalized)
```

**OUTPUT**




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

a) Smoothing by bin mean

b) Smoothing by bin median

c) Smoothing by bin boundaries

**INPUT**

```
data <- c(11,13,13,15,15,16,19,20,20,20,21,21,22,23,24,30,40,45,45,45,71,72,73,75)

bins <- 5

bin_indices <- cut(data, bins)

mean_smooth <- tapply(data, bin_indices, mean)

print(mean_smooth)
```

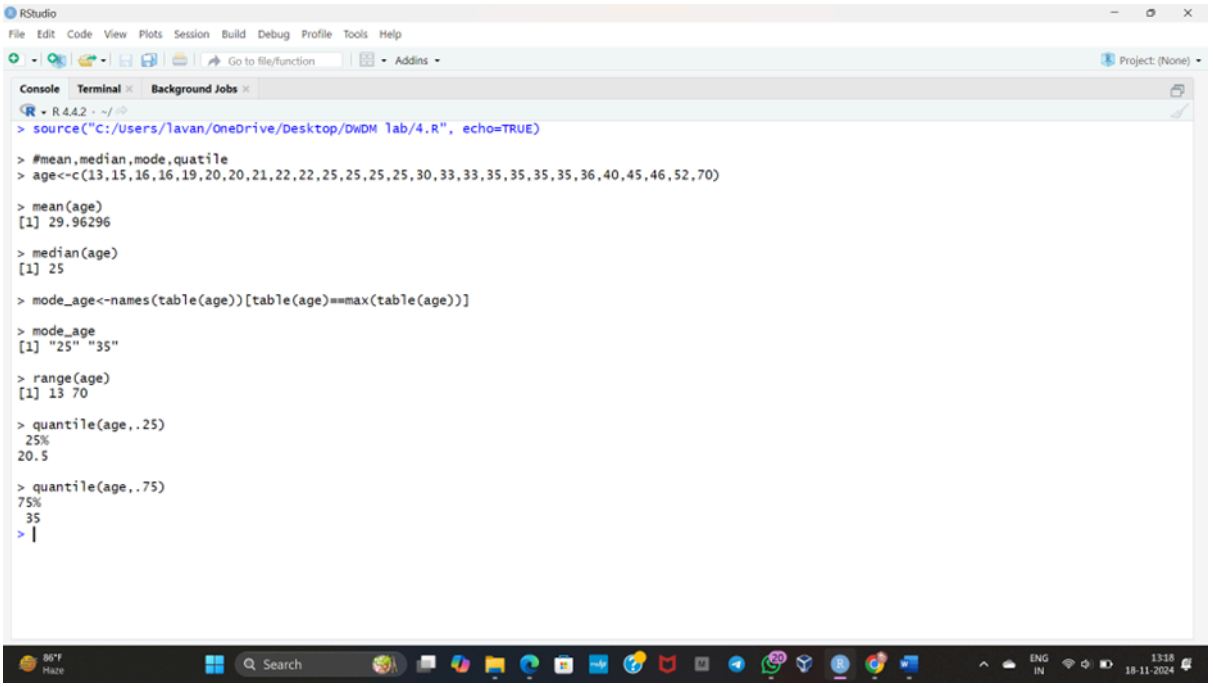
median_smooth <- tapply(data, bin_indices, median)

median_smooth

min_max_smooth <- tapply(data, bin_indices, function(x) c(min(x), max(x)))

print(min_max_smooth)

**OUTPUT**




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

(a) Calculate the mean, median, and standard deviation of age and %fat.

(b) Draw the boxplots for age and %fat.

(c) Draw a scatter plot and a q-q plot based on these two variables.

**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)

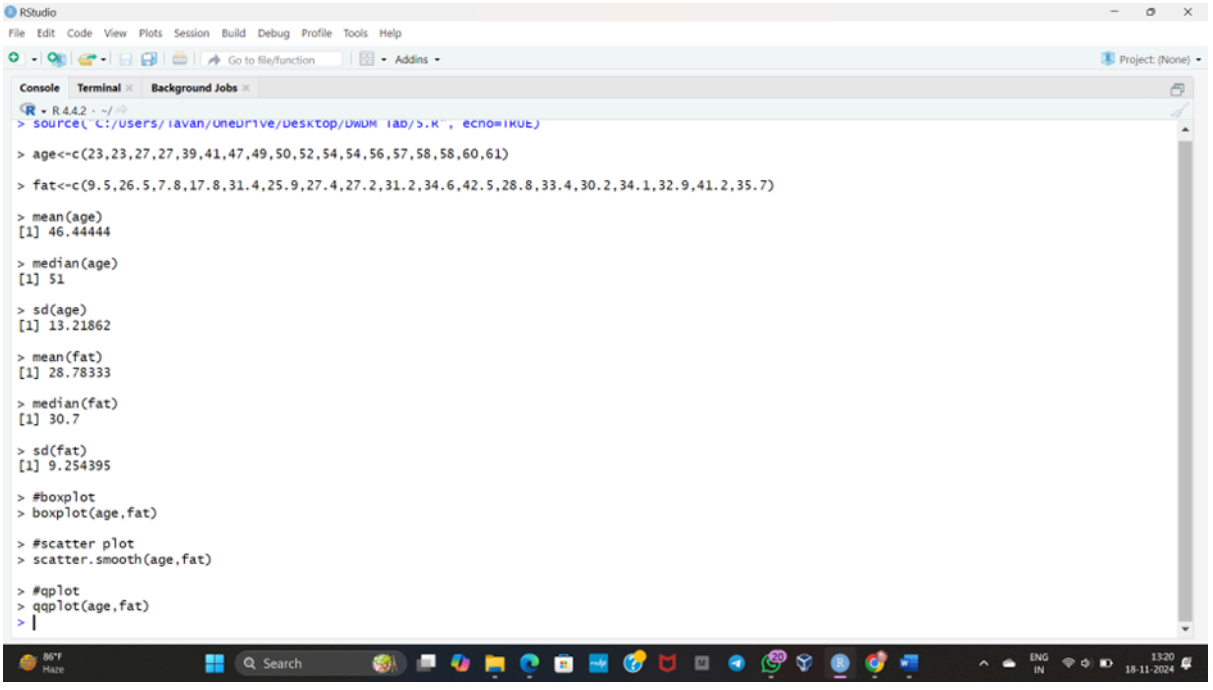
#scatter plot

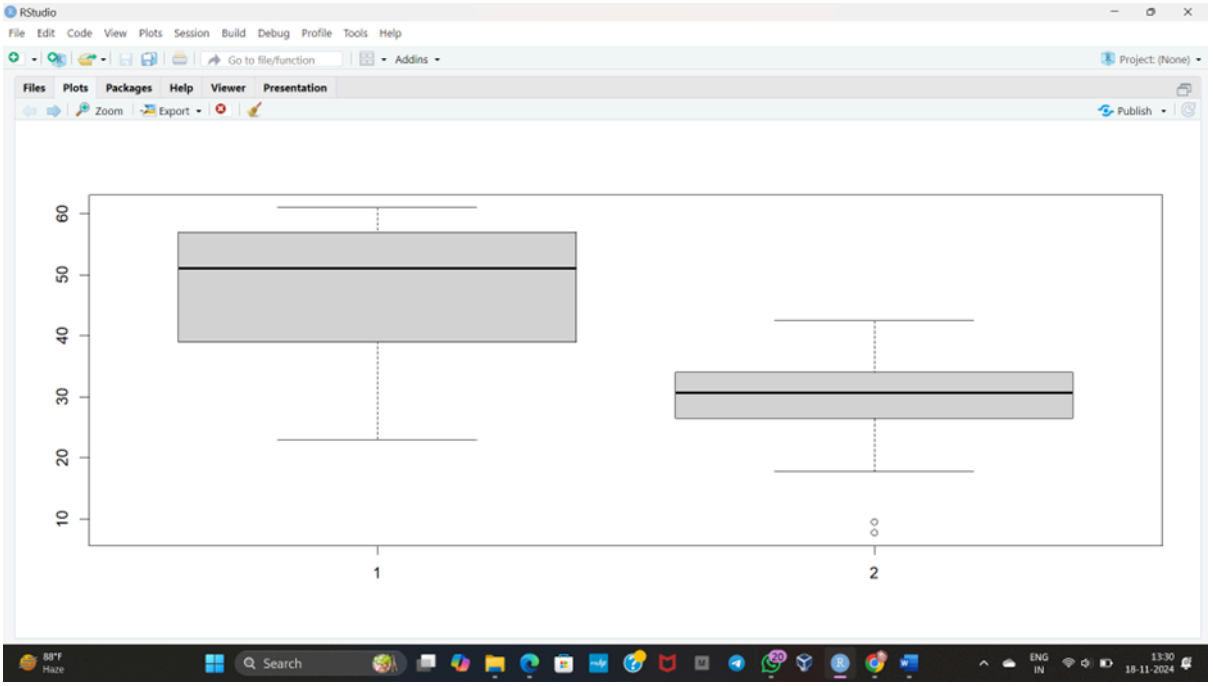
scatter.smooth(age,fat)

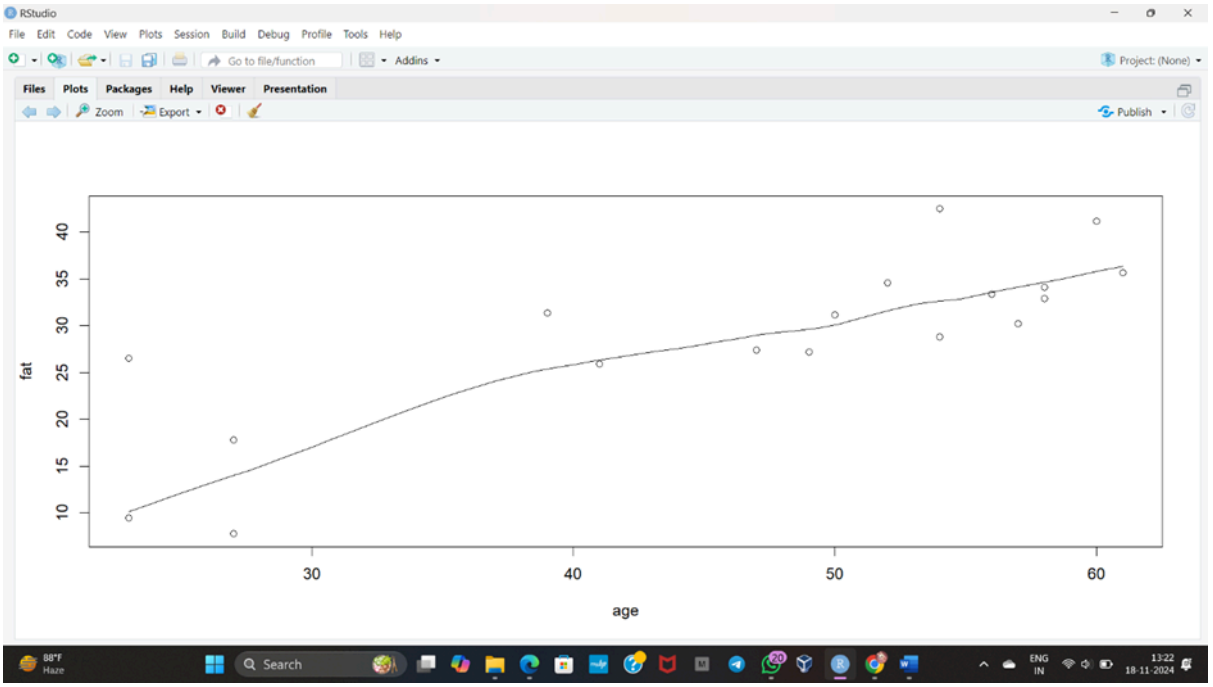
#qplot

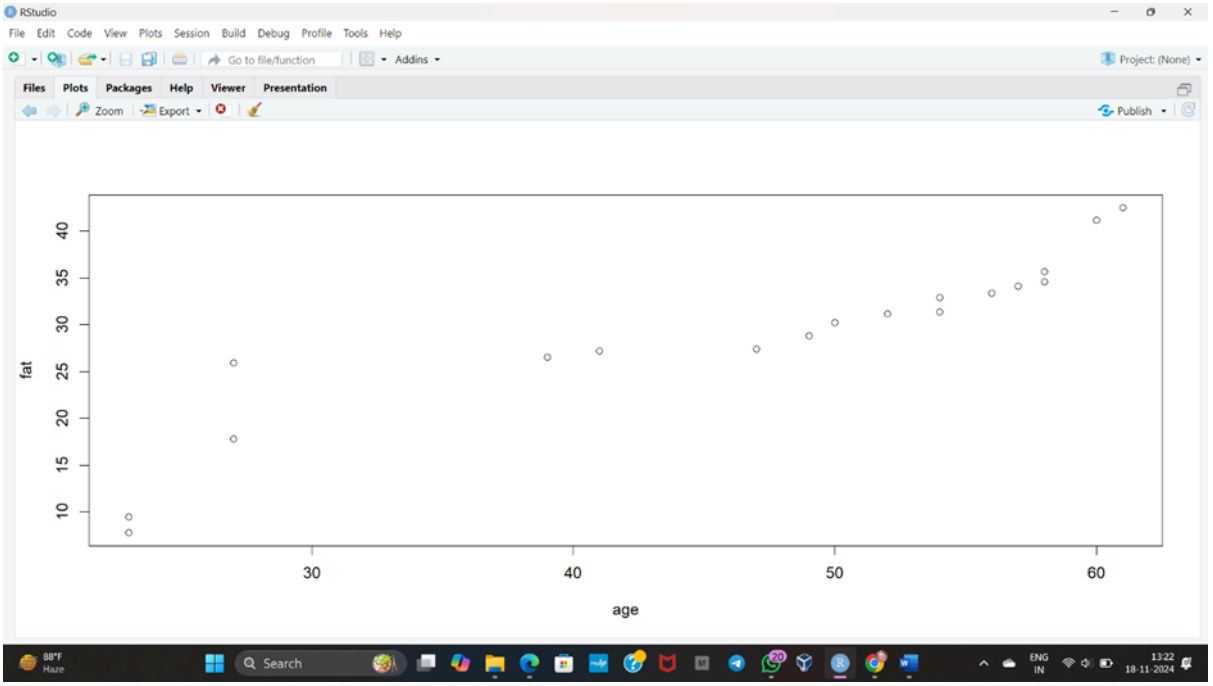
qqplot(age,fat)

**OUTPUT**




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Files Plots Packages Help Viewer Presentation
Zoom
Export
Publish
60
50
40
30
20
10
1
2
88°F
Haze
Search
ENG
IN
13:30
18-11-2024


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Files Plots Packages Help Viewer Presentation
Zoom
Export
Publish
fat
40
35
30
25
20
15
10
30
40
50
60
age
88°F
Haze
Search
ENG
IN
13:22
18-11-2024




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

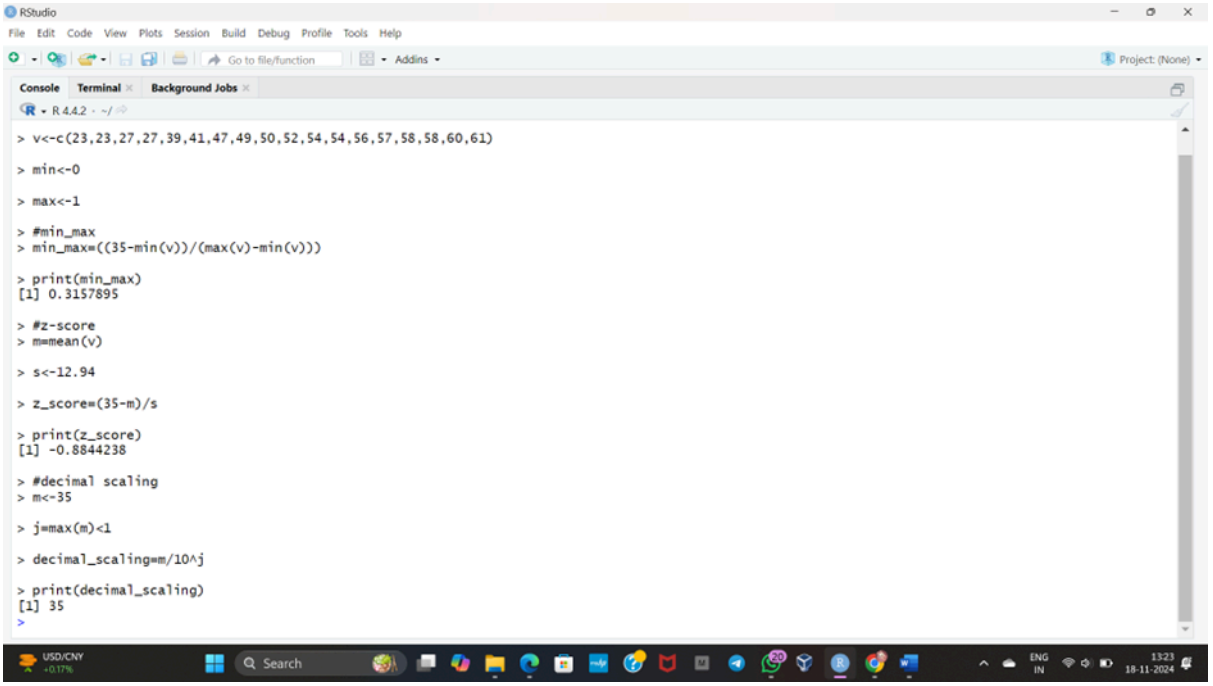
(i) Use min-max normalization to transform the value 35 for age onto the range [0.0, 1.0].
(ii) Use z-score normalization to transform the value 35 for age, where the standard deviation of age is 12.94 years.
(iii) Use normalization by decimal scaling to transform the value 35 for age. Perform the above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

**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**



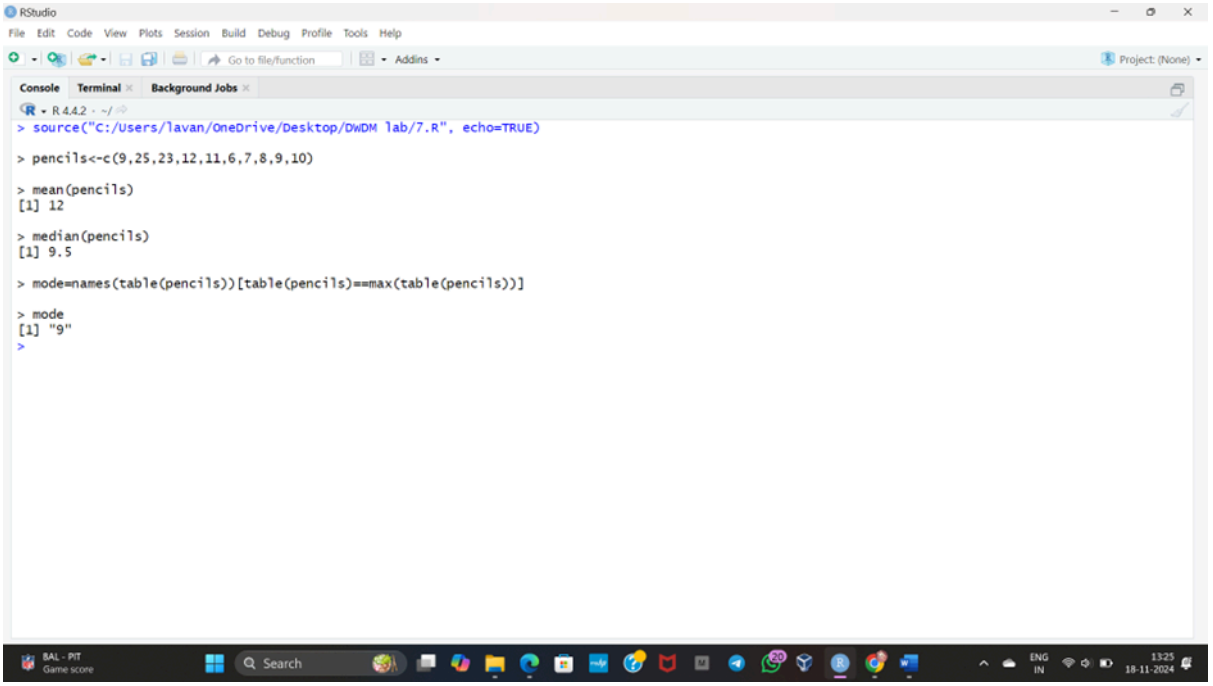

7.The following values are the number of pencils available in the different boxes. Create a vector and find out the mean, median and mode values of set of pencils in the given data.

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

**INPUT**

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

mean(pencils)

median(pencils)

mode=names(table(pencils))[table(pencils)==max(table(pencils))]

mode
```

**OUTPUT**




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

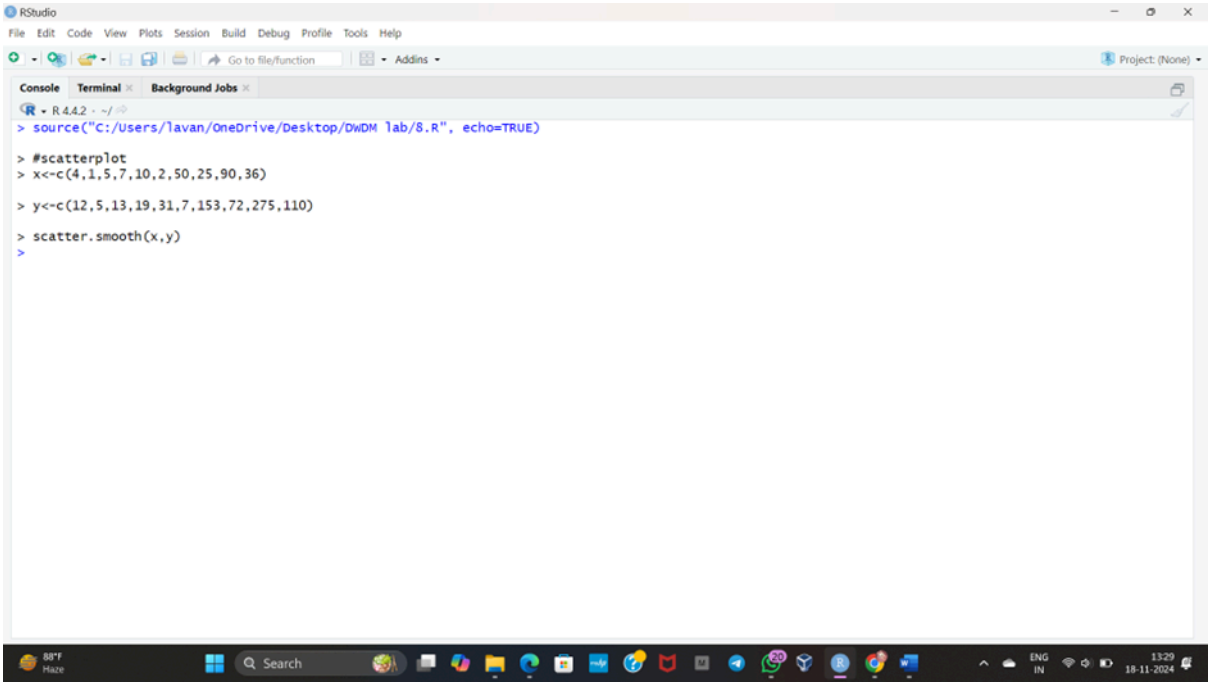
x :4 1 5 7 10 2 50 25 90 36

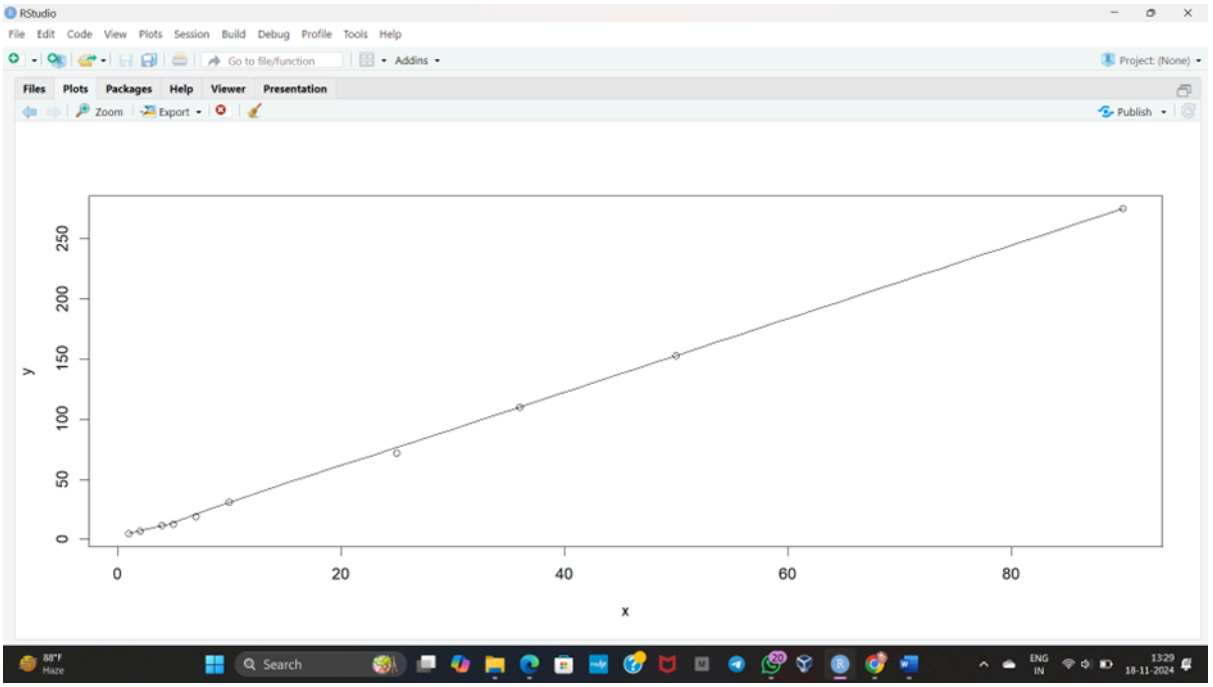
y :12 5 13 19 31 7 153 72 275 110

**INPUT**

```
#scatterplot

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

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

scatter.smooth(x,y)
```

**OUTPUT**




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

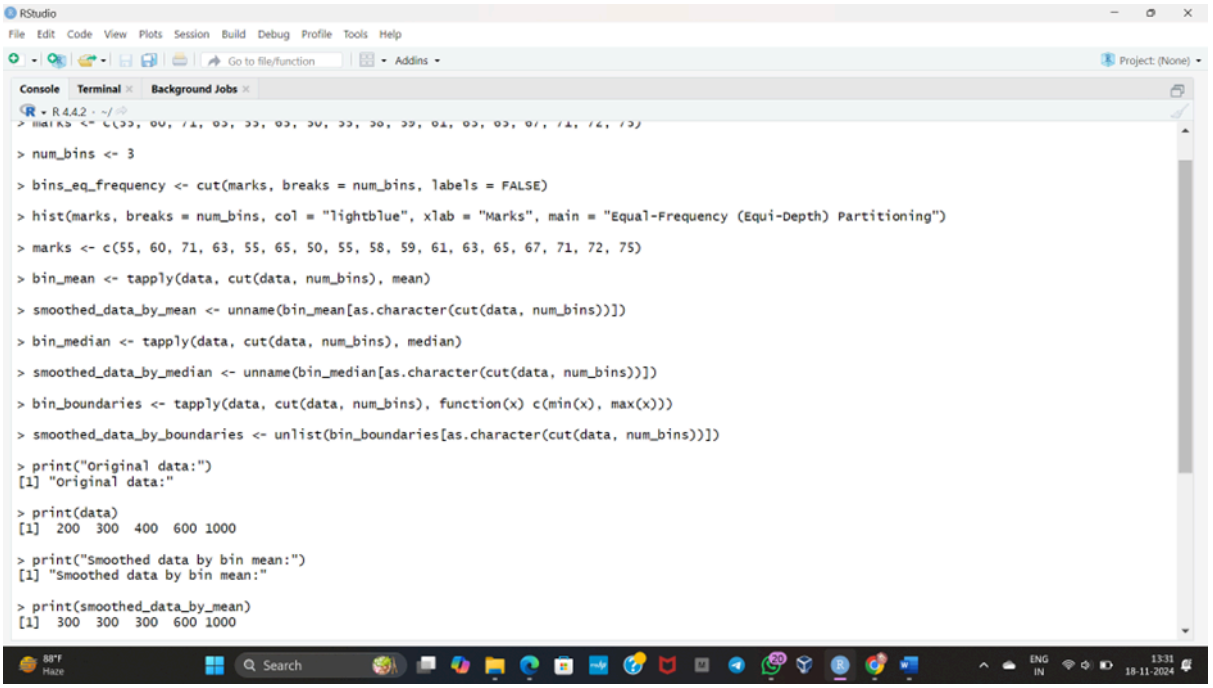
(a) equal-frequency (equi-depth) partitioning (b) equal-width partitioning

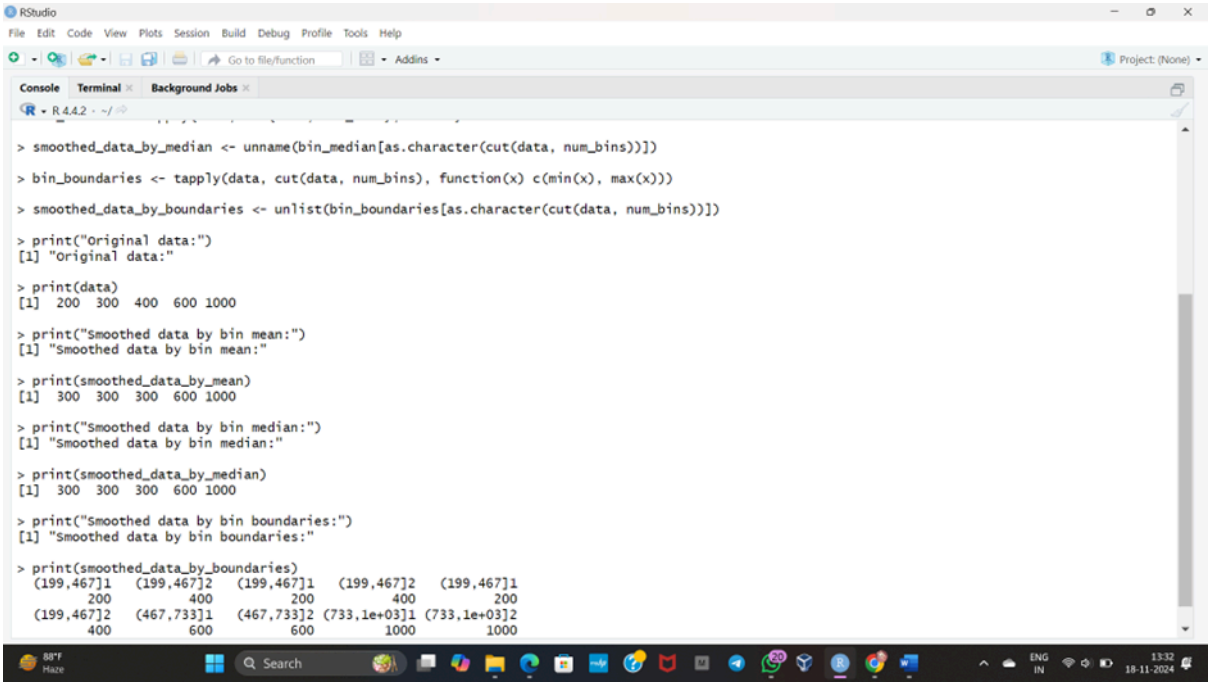
**INPUT**

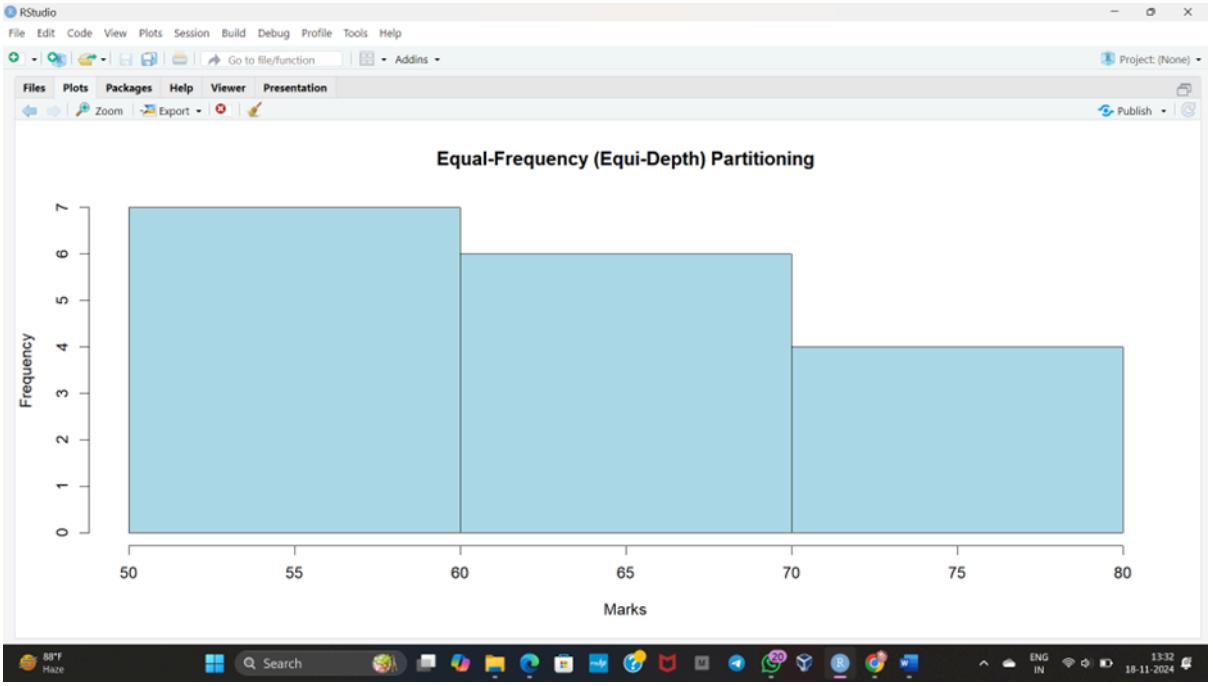
```
marks <- c(55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65, 67, 71, 72, 75)

num_bins <- 3

bins_eq_frequency <- cut(marks, breaks = num_bins, labels = FALSE)

hist(marks, breaks = num_bins, col = "lightblue", xlab = "Marks", main =
"Equal-Frequency (Equi-Depth) Partitioning")

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

bin_mean <- tapply(data, cut(data, num_bins), mean)

smoothed_data_by_mean <- unname(bin_mean[as.character(cut(data,
num_bins))])

bin_median <- tapply(data, cut(data, num_bins), median)

smoothed_data_by_median <- unname(bin_median[as.character(cut(data,
num_bins))])

bin_boundaries <- tapply(data, cut(data, num_bins), function(x) c(min(x),
max(x)))

smoothed_data_by_boundaries <- unlist(bin_boundaries[as.character(cut(data,
num_bins))])

print("Original data:")

print(data)

print("Smoothed data by bin mean:")

print(smoothed_data_by_mean)

print("Smoothed data by bin median:")

print(smoothed_data_by_median)

print("Smoothed data by bin boundaries:")
```

print(smoothed_data_by_boundaries)

## OUTPUT






**10.** Suppose that the speed car is mentioned in different driving style.

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

Calculate the Inter quantile and standard deviation of the given data.

**INPUT**

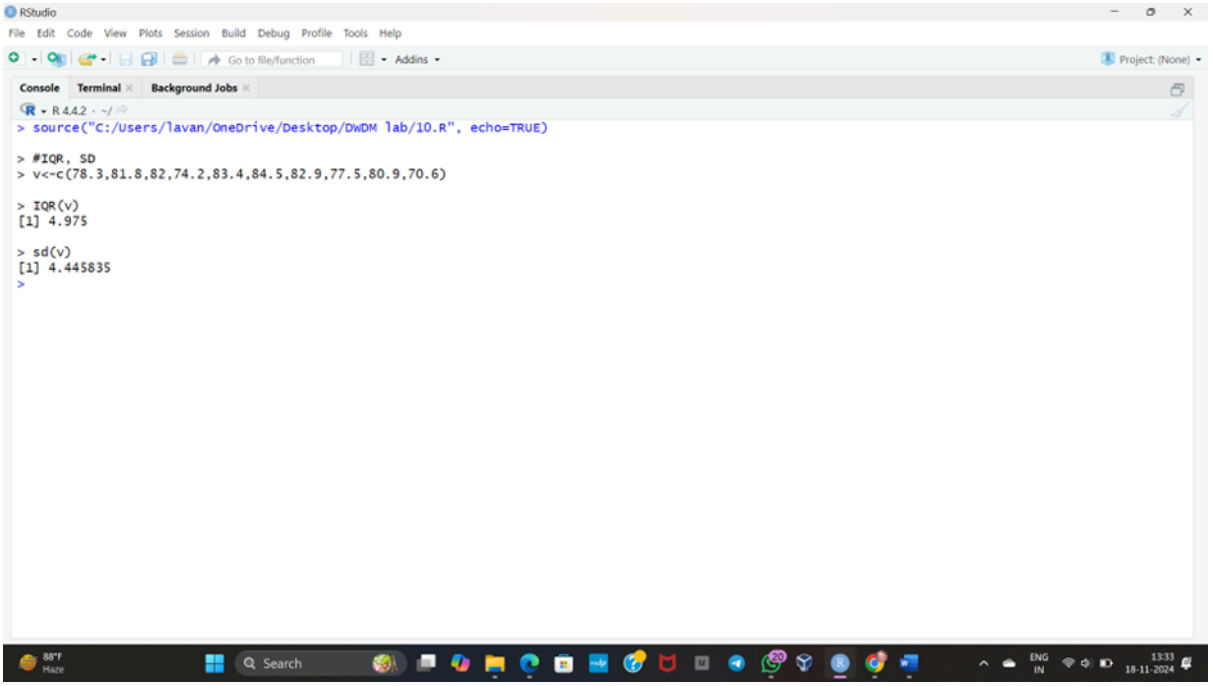
#IQR, SD

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

IQR(v)

sd(v)

**OUTPUT**




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

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

**INPUT**

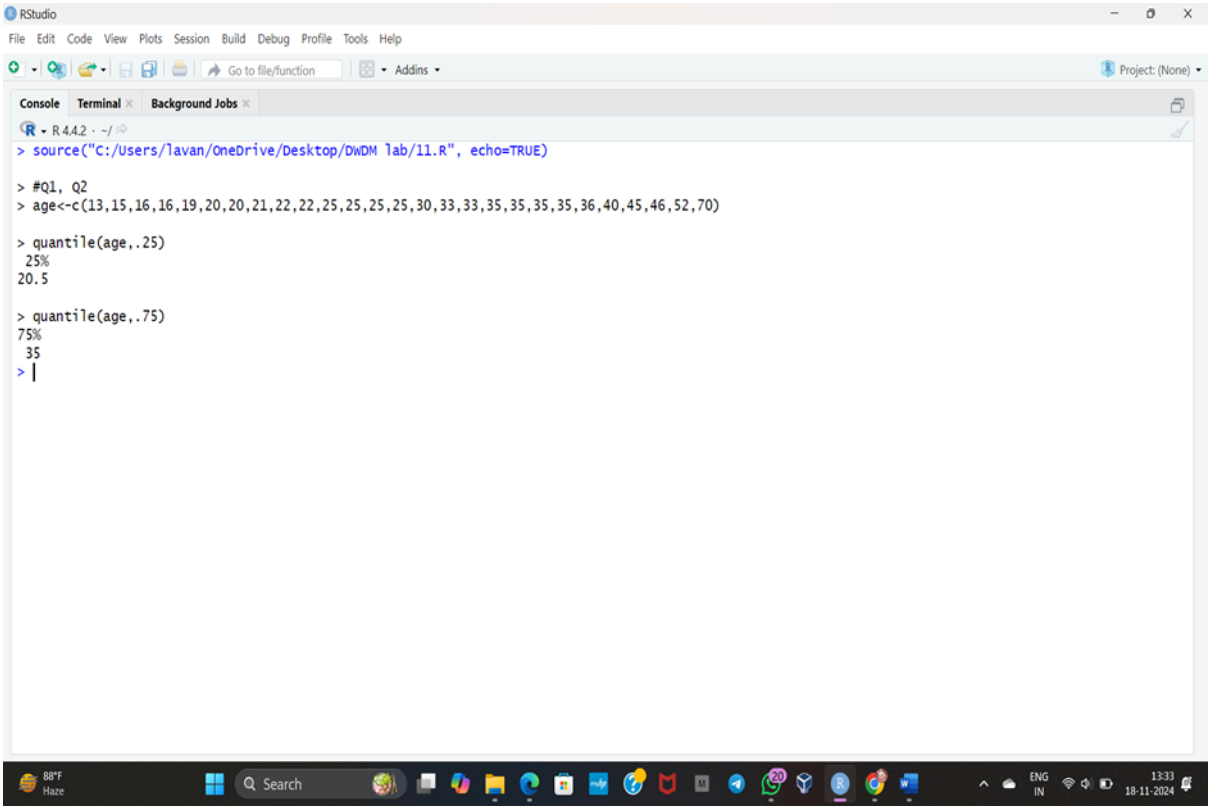
#Q1, Q2

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**




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

(a) equal-frequency (equi depth) partitioning

(b) equal-width partitioning

(c) clustering

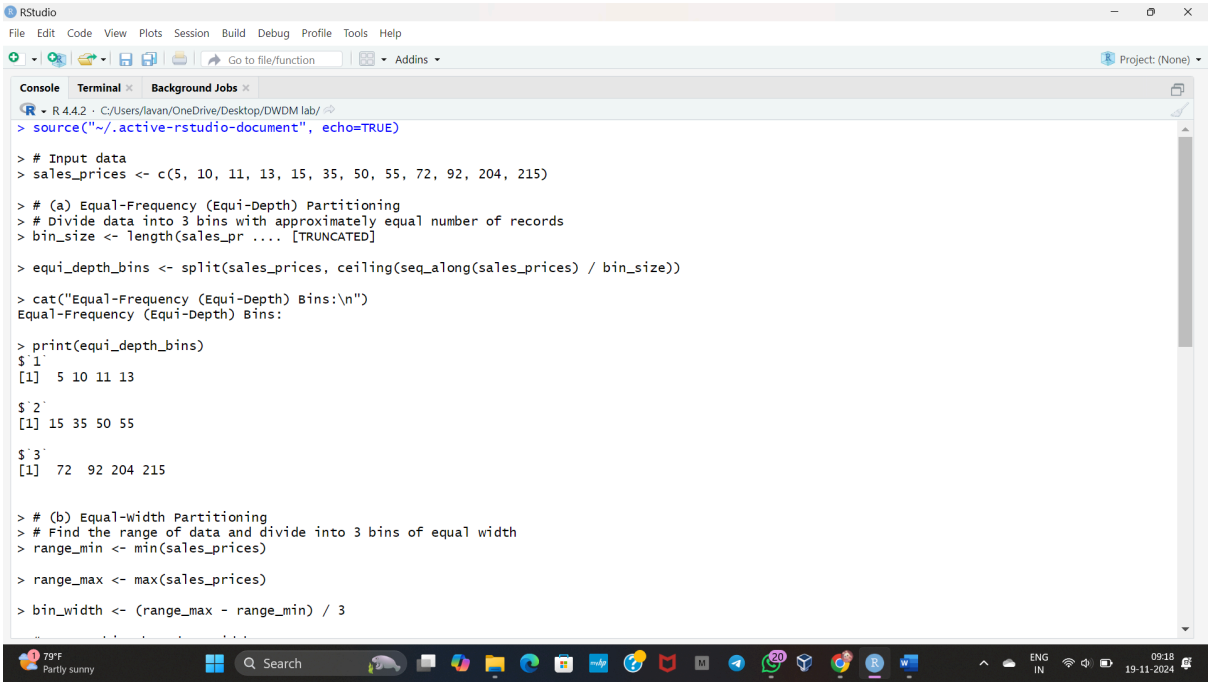
## INPUT

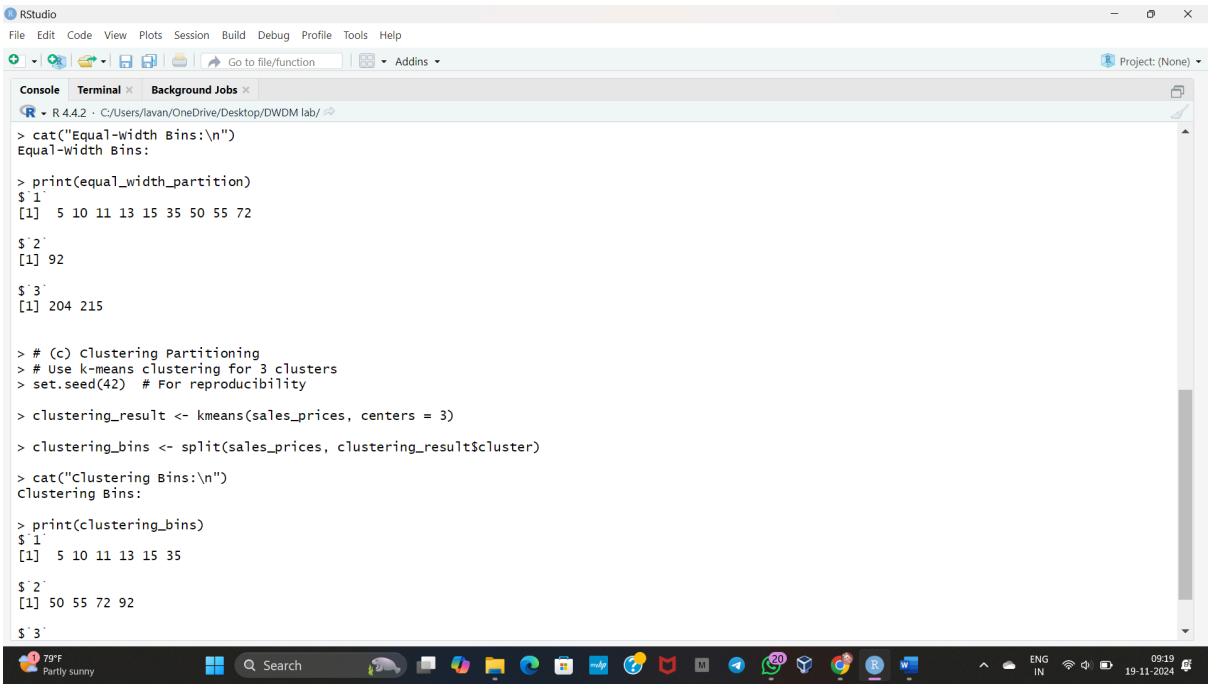
```
# Input data

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



# (a) Equal-Frequency (Equi-Depth) Partitioning

# Divide data into 3 bins with approximately equal number of records

bin_size <- length(sales_prices) / 3
```

```
equi_depth_bins <- split(sales_prices, ceiling(seq_along(sales_prices) / bin_size))

cat("Equal-Frequency (Equi-Depth) Bins:\n")

print(equi_depth_bins)


# (b) Equal-Width Partitioning

# Find the range of data and divide into 3 bins of equal width

range_min <- min(sales_prices)

range_max <- max(sales_prices)

bin_width <- (range_max - range_min) / 3


# Create bins based on width

bin_edges <- seq(range_min, range_max, by = bin_width)

equal_width_bins <- cut(sales_prices, breaks = bin_edges, include.lowest = TRUE, labels =
FALSE)

equal_width_partition <- split(sales_prices, equal_width_bins)

cat("Equal-Width Bins:\n")

print(equal_width_partition)


# (c) Clustering Partitioning

# Use k-means clustering for 3 clusters

set.seed(42)  # For reproducibility

clustering_result <- kmeans(sales_prices, centers = 3)

clustering_bins <- split(sales_prices, clustering_result$cluster)

cat("Clustering Bins:\n")
```

print(clustering_bins)

**OUTPUT**




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

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

(b) z-score normalization

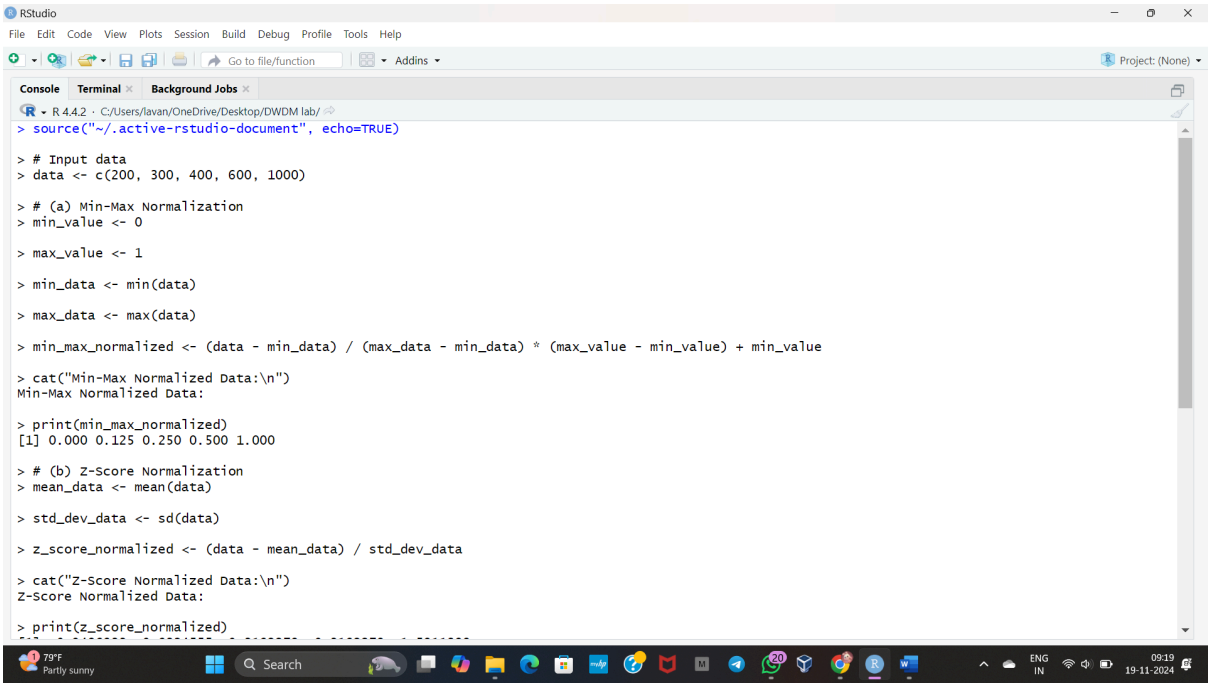
(c) (c) z-score normalization using the mean absolute deviation instead of standard deviation (d) normalization by decimal scaling

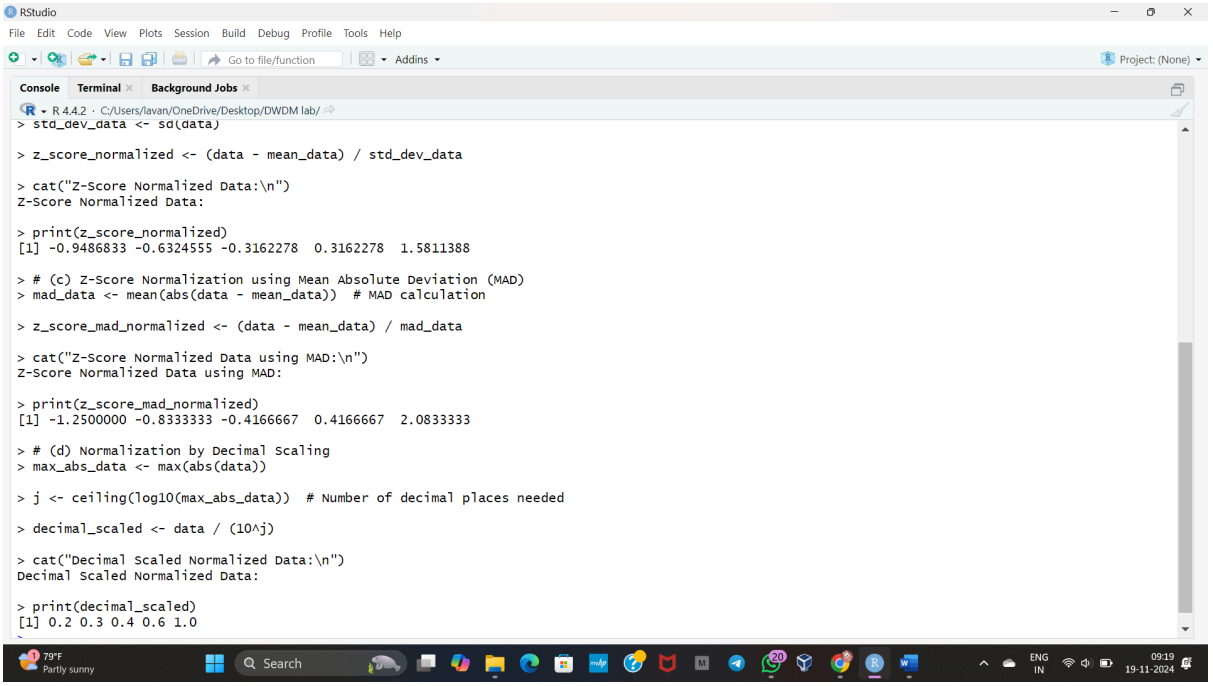
**INPUT**

```
# Input data

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


# (a) Min-Max Normalization

min_value <- 0

max_value <- 1


min_data <- min(data)

max_data <- max(data)


min_max_normalized <- (data - min_data) / (max_data - min_data) * (max_value - min_value) + min_value

cat("Min-Max Normalized Data:\n")

print(min_max_normalized)


# (b) Z-Score Normalization

mean_data <- mean(data)

std_dev_data <- sd(data)


z_score_normalized <- (data - mean_data) / std_dev_data

cat("Z-Score Normalized Data:\n")

print(z_score_normalized)
```

```
# (c) Z-Score Normalization using Mean Absolute Deviation (MAD)
mad_data <- mean(abs(data - mean_data))  # MAD calculation
z_score_mad_normalized <- (data - mean_data) / mad_data
cat("Z-Score Normalized Data using MAD:\n")
print(z_score_mad_normalized)

# (d) Normalization by Decimal Scaling
max_abs_data <- max(abs(data))
j <- ceiling(log10(max_abs_data))  # Number of decimal places needed
decimal_scaled <- data / (10^j)
cat("Decimal Scaled Normalized Data:\n")
print(decimal_scaled)
```

**OUTPUT**






## DAY-2

**LIST OF PROGRAMS:**

**14**.Covariance and correlation

Children of three ages are asked to indicate their preference for three photographs of adults. Do the data suggest that there is a significant relationship between age and photograph preference? What is wrong with this study?

| | **Photograph:** | | |
|---|---|---|---|
| **Age of child** | A | B | C |
| 5-6 years: | 18 | 22 | 20 |

7-8 years: 2 28
40

9-10 years: 20 10
40

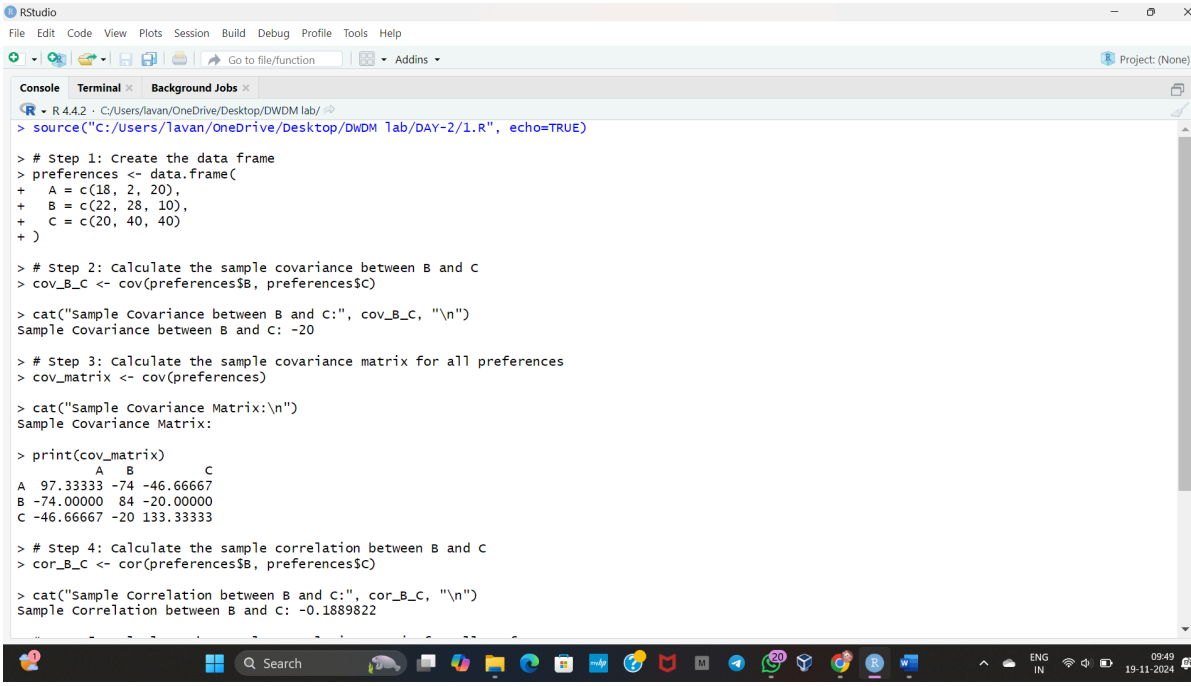
1. Use cov() to calculate the sample covariance between B and C.
2. Use another call to cov() to calculate the sample covariance matrix for the preferences.
3. Use cor() to calculate the sample correlation between B and C.
4. Use another call to cor() to calculate the sample correlation matrix for the preferences.

**INPUT**

```
# Step 1: Create the data frame
preferences <- data.frame(
  A = c(18, 2, 20),
  B = c(22, 28, 10),
  C = c(20, 40, 40)
)

# Step 2: Calculate the sample covariance between B and C
cov_B_C <- cov(preferences$B, preferences$C)
cat("Sample Covariance between B and C:", cov_B_C, "\n")
```

```
# Step 3: Calculate the sample covariance matrix for all
    preferences

cov_matrix <- cov(preferences)

cat("Sample Covariance Matrix:\n")

print(cov_matrix)


# Step 4: Calculate the sample correlation between B and C

cor_B_C <- cor(preferences$B, preferences$C)

cat("Sample Correlation between B and C:", cor_B_C, "\n")


# Step 5: Calculate the sample correlation matrix for all
    preferences

cor_matrix <- cor(preferences)

cat("Sample Correlation Matrix:\n")

print(cor_matrix)
```

**OUTPUT**




**15.**Imagine that you have selected data from the All Electronics data warehouse for analysis. The data set will be huge! The following data are a list of All Electronics prices for commonly sold items (rounded to the nearest dollar). The numbers have been sorted: 1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18,

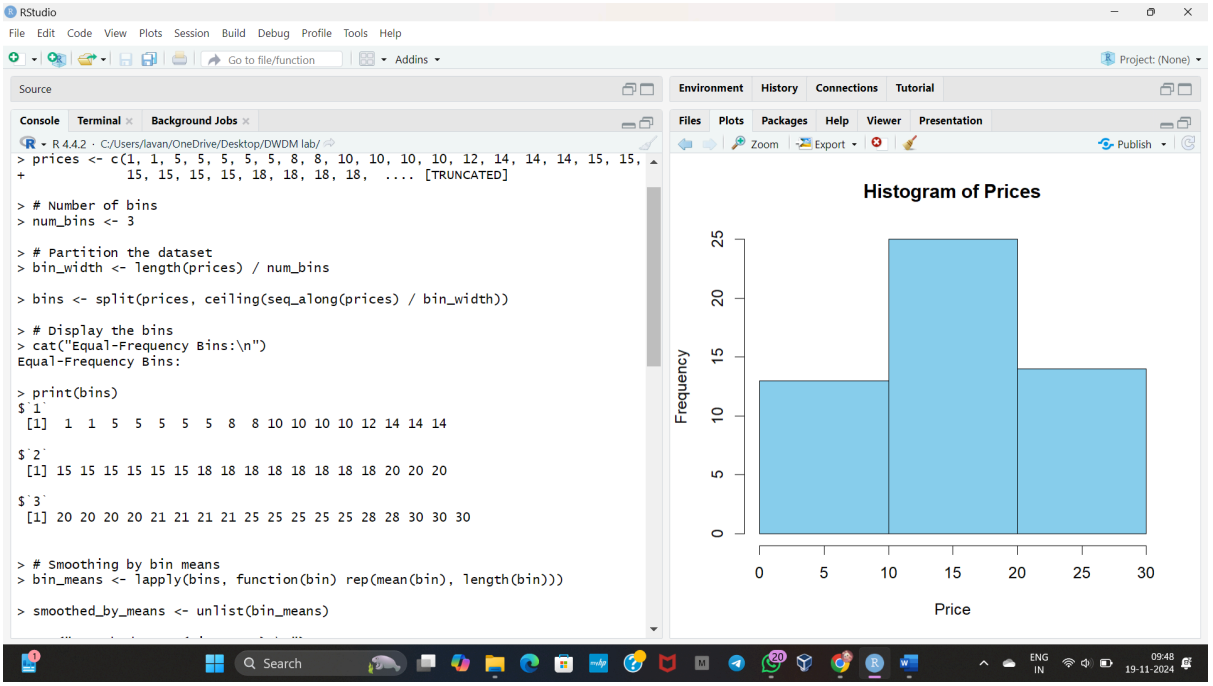
| |
|---|
| 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, |
| 30, 30.<br><br>(i) Partition the dataset using an equal-frequency partitioning method with bin equal to 3 (ii) apply data smoothing using bin means and bin boundary.<br>(iii) Plot Histogram for the above frequency division |

**INPUT**

```
# Input the dataset
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, 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)

# Number of bins
num_bins <- 3

# Partition the dataset
bin_width <- length(prices) / num_bins
bins <- split(prices, ceiling(seq_along(prices) / bin_width))

# Display the bins
cat("Equal-Frequency Bins:\n")
print(bins)

# Smoothing by bin means
bin_means <- lapply(bins, function(bin) rep(mean(bin), length(bin)))
smoothed_by_means <- unlist(bin_means)
cat("Smoothed Data (Bin Means):\n")
print(smoothed_by_means)
```

```
# Smoothing by bin boundaries
bin_boundaries <- lapply(bins, function(bin) {
  lower <- min(bin)
  upper <- max(bin)
  sapply(bin, function(x) ifelse(abs(x - lower) <= abs(x - upper), lower, upper))
})
smoothed_by_boundaries <- unlist(bin_boundaries)
cat("Smoothed Data (Bin Boundaries):\n")
print(smoothed_by_boundaries)

# Plot histogram
hist(prices, breaks=num_bins, col="skyblue", main="Histogram of Prices",
     xlab="Price", ylab="Frequency", border="black")
```

**OUTPUT**




**16.**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

Class B: 51, 56, 84, 60, 59, 70, 63, 66, 5051,56,84,60,59,70,63,66,50

**INPUT**

# Class A and Class B scores

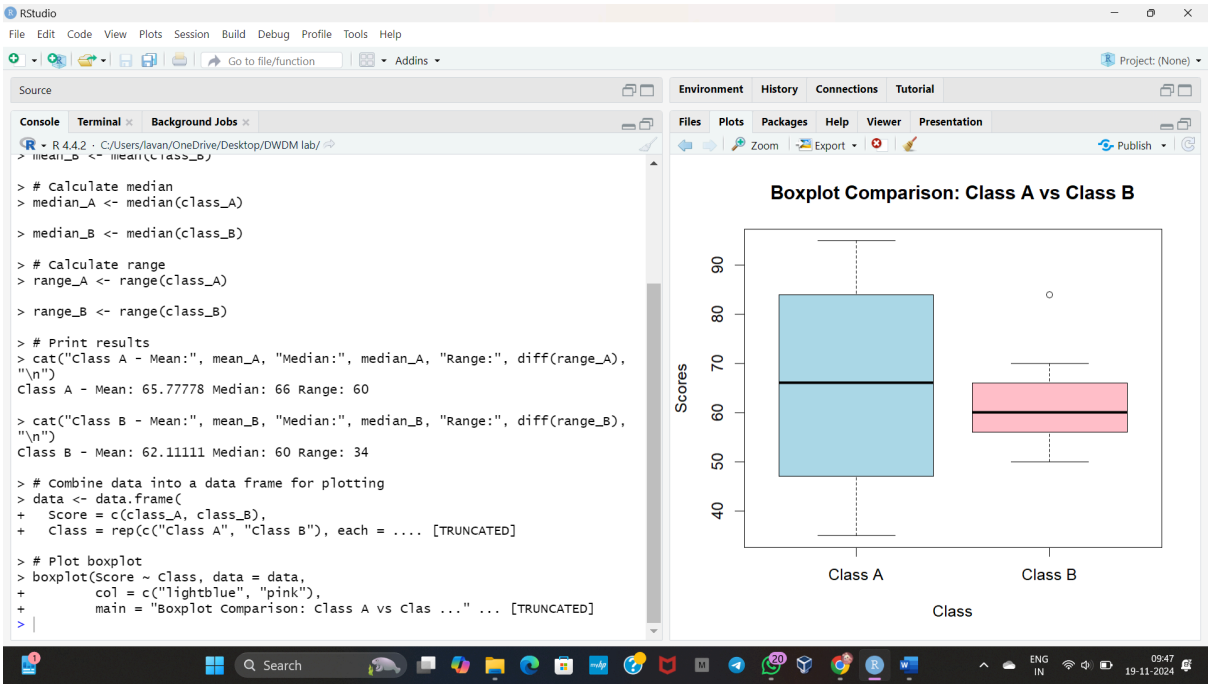
class_A <- c(76, 35, 47, 64, 95, 66, 89, 36, 84)

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

```
# Calculate mean

mean_A <- mean(class_A)

mean_B <- mean(class_B)


# Calculate median

median_A <- median(class_A)

median_B <- median(class_B)


# Calculate range

range_A <- range(class_A)

range_B <- range(class_B)


# Print results

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

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


# Combine data into a data frame for plotting

data <- data.frame(

  Score = c(class_A, class_B),

  Class = rep(c("Class A", "Class B"), each = length(class_A))

)
```

```
# Plot boxplot

boxplot(Score ~ Class, data = data,

        col = c("lightblue", "pink"),

        main = "Boxplot Comparison: Class A vs Class B",

        xlab = "Class",

        ylab = "Scores")
```

**OUTPUT**




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

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

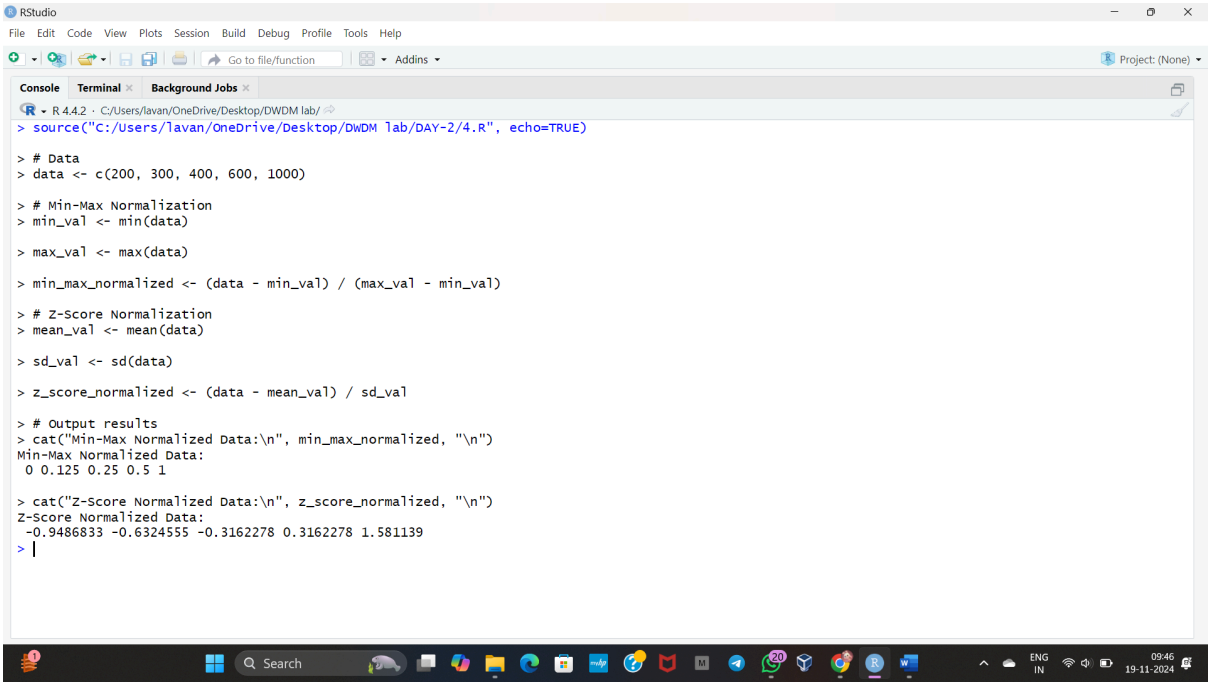
(a) min-max normalization by setting min = 0 and max = 1

(b) z-score normalization

**INPUT**

```
# Data
data <- c(200, 300, 400, 600, 1000)


# Min-Max Normalization
min_val <- min(data)
max_val <- max(data)
min_max_normalized <- (data - min_val) / (max_val - min_val)


# Z-Score Normalization
mean_val <- mean(data)
sd_val <- sd(data)
z_score_normalized <- (data - mean_val) / sd_val


# Output results
cat("Min-Max Normalized Data:\n", min_max_normalized, "\n")
cat("Z-Score Normalized Data:\n", z_score_normalized, "\n")
```

**OUTPUT**




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

**INPUT**

# Load the AirPassengers dataset

data <- AirPassengers

# Inspect the data

summary(data)

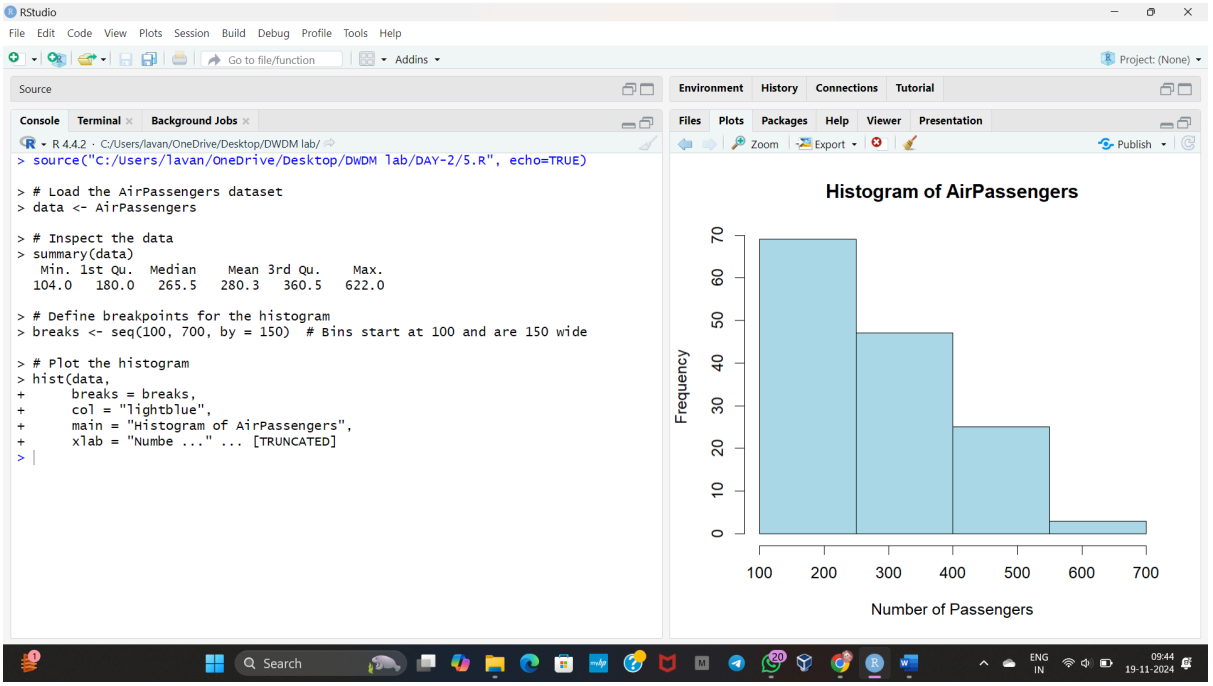
# Define breakpoints for the histogram

breaks <- seq(100, 700, by = 150)  # Bins start at 100 and are 150 wide

```
# Plot the histogram
hist(data,
     breaks = breaks,
     col = "lightblue",
     main = "Histogram of AirPassengers",
     xlab = "Number of Passengers",
     ylab = "Frequency",
     xlim = c(100, 700),
     border = "black")

# Step 1: Install and load the water dataset from a link
install.packages("datasets.load")
library(datasets.load)

# Load the "water" dataset
data <- datasets.load::load_data("water")

# Step 2: Check the dataset structure
str(data)

# Step 3: Visualize the relationship
plot(data$hardness, data$mortality,
```

```
    xlab = "Hardness",
    ylab = "Mortality",
    main = "Mortality vs Hardness",
    col = "blue", pch = 16)

# Step 4: Fit the linear regression model
model <- lm(mortality ~ hardness, data = data)
summary(model)

# Step 5: Predict mortality for hardness = 88
new_data <- data.frame(hardness = 88)
predicted_mortality <- predict(model, newdata = new_data)
print(paste("Predicted mortality for hardness = 88 is:", predicted_mortality))

# Add regression line to the plot
abline(model, col = "red", lwd = 2)
```

**OUTPUT**




**19.**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.

**INPUT**

# Step 1: Install and load the water dataset from a link

install.packages("datasets.load")

library(datasets.load)

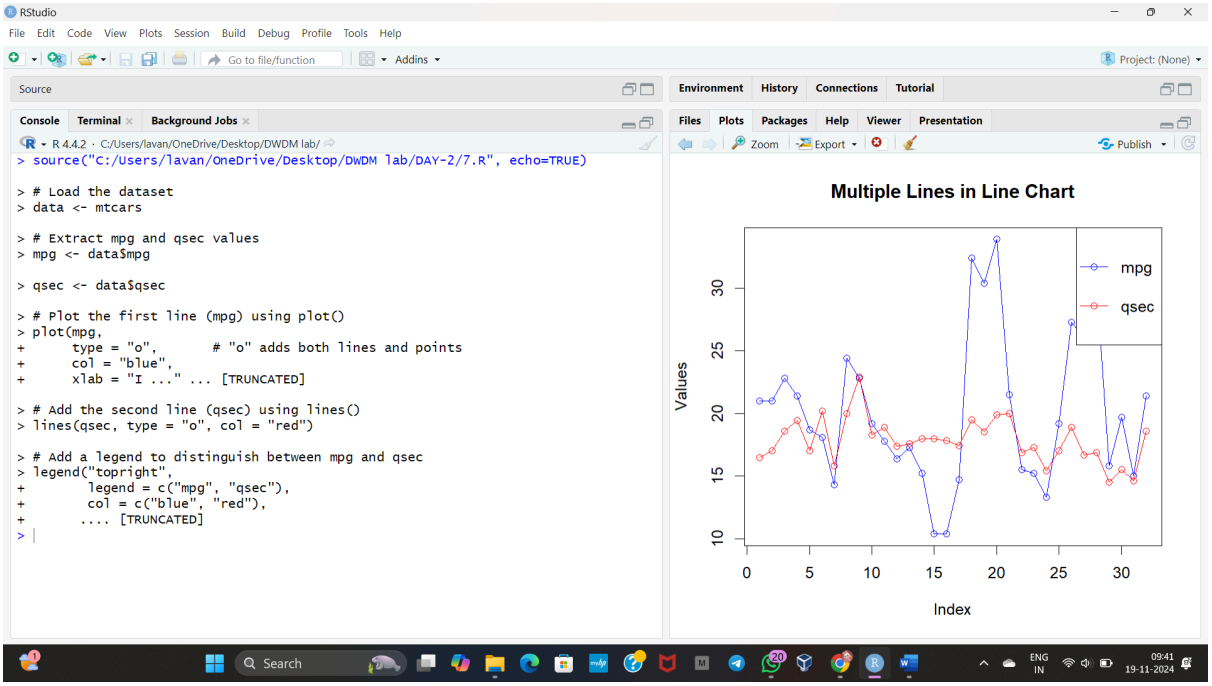
# Load the "water" dataset

data <- datasets.load::load_data("water")

# Step 2: Check the dataset structure

```
str(data)

# Step 3: Visualize the relationship
plot(data$hardness, data$mortality,
     xlab = "Hardness",
     ylab = "Mortality",
     main = "Mortality vs Hardness",
     col = "blue", pch = 16)

# Step 4: Fit the linear regression model
model <- lm(mortality ~ hardness, data = data)
summary(model)

# Step 5: Predict mortality for hardness = 88
new_data <- data.frame(hardness = 88)
predicted_mortality <- predict(model, newdata = new_data)
print(paste("Predicted mortality for hardness = 88 is:", predicted_mortality))

# Add regression line to the plot
abline(model, col = "red", lwd = 2)
```

**OUTPUT**



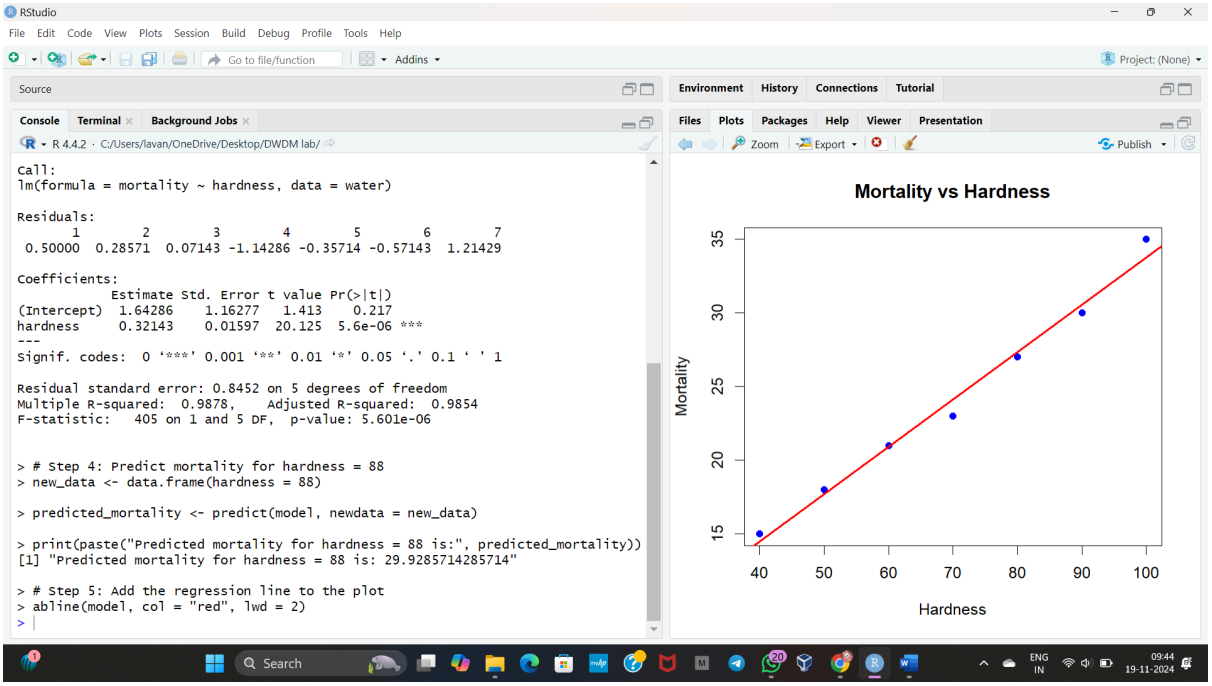

**20.**Obtain Multiple Lines in Line Chart using a single Plot Function in R.Use attributes“mpg”and“qsec”of the dataset “mtcars”

**INPUT**

```
# Step 1: Create a custom dataset

water <- data.frame(

  hardness = c(40, 50, 60, 70, 80, 90, 100),  # Example hardness values

  mortality = c(15, 18, 21, 23, 27, 30, 35)   # Example mortality values

)


# Check the structure

str(water)


# Step 2: Visualize the relationship between 'hardness' and 'mortality'
```

```
plot(water$hardness, water$mortality,
     xlab = "Hardness",
     ylab = "Mortality",
     main = "Mortality vs Hardness",
     col = "blue", pch = 16)

# Step 3: Fit the linear regression model
model <- lm(mortality ~ hardness, data = water)

# Display model summary
summary(model)

# Step 4: Predict mortality for hardness = 88
new_data <- data.frame(hardness = 88)
predicted_mortality <- predict(model, newdata = new_data)
print(paste("Predicted mortality for hardness = 88 is:", predicted_mortality))

# Step 5: Add the regression line to the plot
abline(model, col = "red", lwd = 2)
```

**OUTPUT**



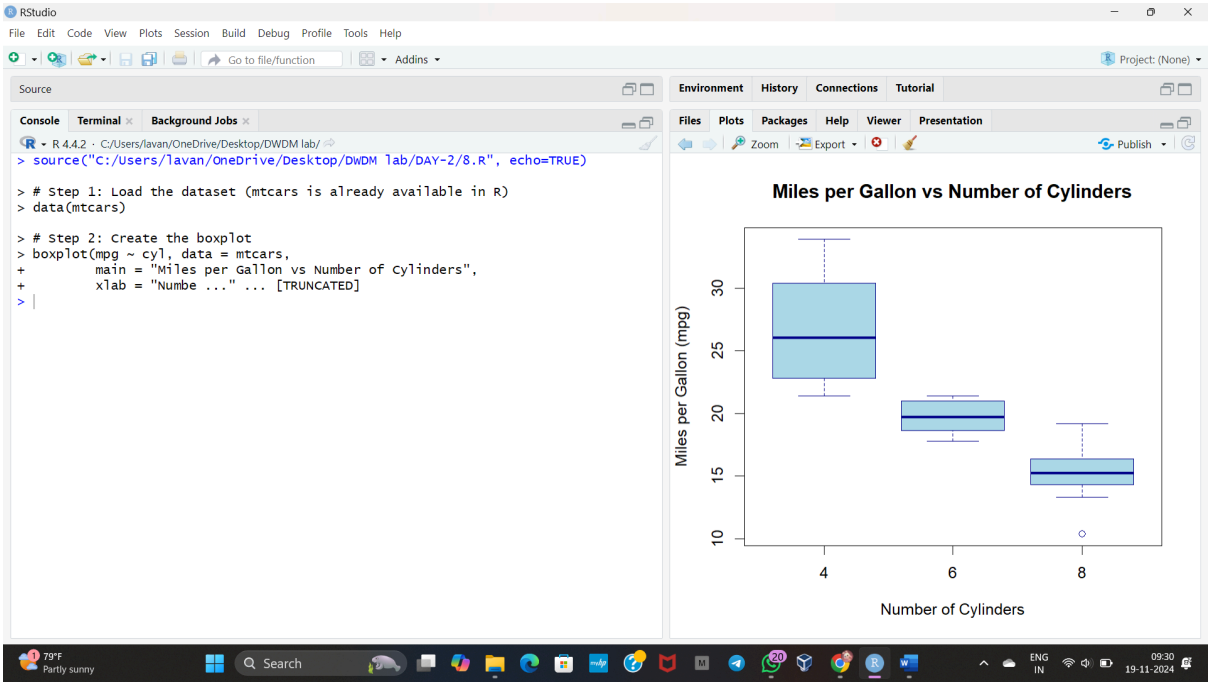

**21.**Create a Boxplot graph for the relation between "mpg"(miles per galloon) and "cyl"(number of Cylinders) for the dataset "mtcars" available in R Environment.

**INPUT**

```
# Step 1: Load the dataset (mtcars is already available in R)

data(mtcars)


# Step 2: Create the boxplot

boxplot(mpg ~ cyl, data = mtcars,

    main = "Miles per Gallon vs Number of Cylinders",

    xlab = "Number of Cylinders",

    ylab = "Miles per Gallon (mpg)",

    col = "lightblue",
```

border = "darkblue")

**OUTPUT**




**22.** Assume the Tennis coach wants to determine if any of his team players are scoring

outliers. To visualize the distribution of points scored by his players, then how can he

decide to develop the box plot? Give suitable example using Boxplot visualization

Technique.

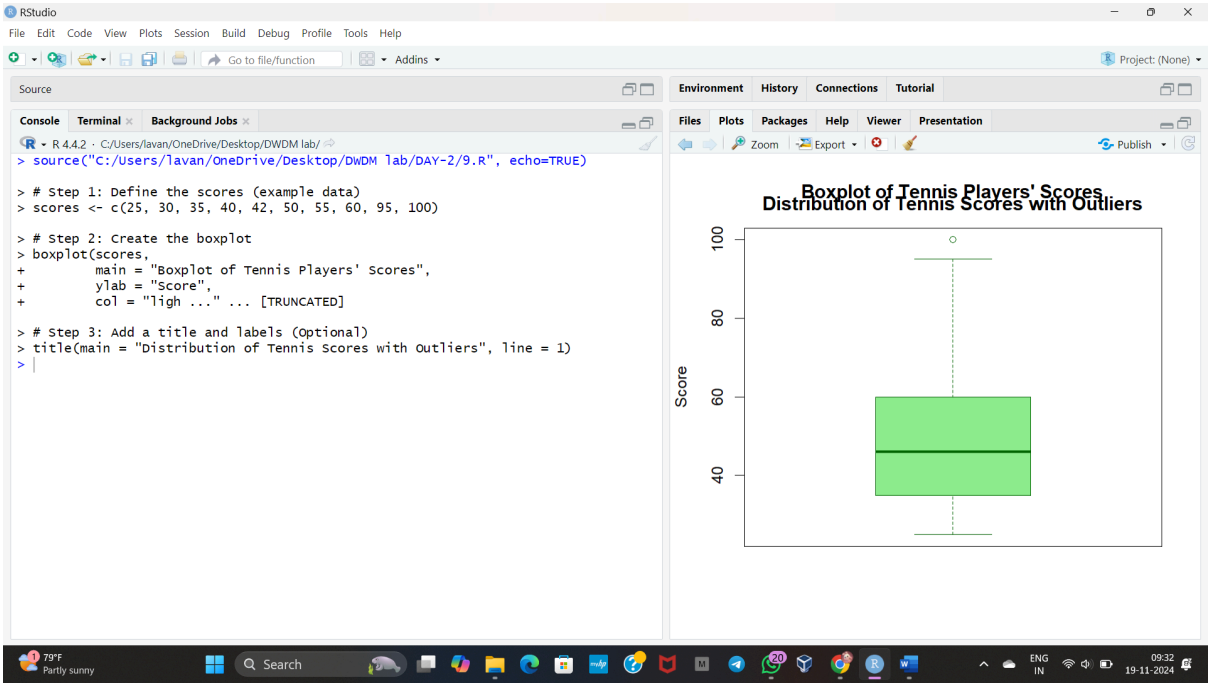
**INPUT**

# Step 1: Define the scores (example data)

scores <- c(25, 30, 35, 40, 42, 50, 55, 60, 95, 100)

```
# Step 2: Create the boxplot
boxplot(scores,
        main = "Boxplot of Tennis Players' Scores",
        ylab = "Score",
        col = "lightgreen",
        border = "darkgreen")

# Step 3: Add a title and labels (Optional)
title(main = "Distribution of Tennis Scores with Outliers", line = 1)
```

**OUTPUT**




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

**INPUT**

```
 # Load libraries
library(ggplot2)


# Manually create dataset
diabetes_data <- data.frame(
  Age = c(50, 60, 45, 70, 55, 65, 40),
  BloodPressure = c(80, 120, 85, 130, 100, 110, 90)
)


# Scatter plot: Blood Pressure vs Age
ggplot(diabetes_data, aes(x = Age, y = BloodPressure)) +
  geom_point() + labs(title = "Blood Pressure vs Age")


# Bar chart: Average Blood Pressure by Age Group
diabetes_data$AgeGroup <- cut(diabetes_data$Age, breaks = c(0, 30, 40, 50, 60, 70, 80, 100), labels = c("0-30", "31-40", "41-50", "51-60", "61-70", "71-80", "81-100"))
ggplot(diabetes_data, aes(x = AgeGroup, y = BloodPressure, fill = AgeGroup)) +
  geom_bar(stat = "summary", fun = "mean", color = "black") +
  labs(title = "Average Blood Pressure by Age Group")
```

**OUTPUT**

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

Source

Console | Terminal | Background Jobs

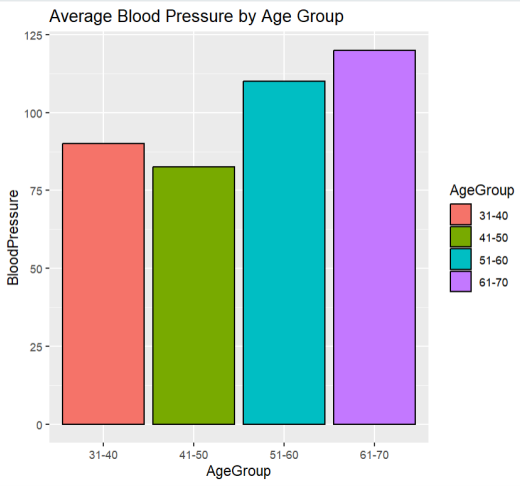
R 4.4.2 · C:/Users/lavan/OneDrive/Desktop/DWDM lab/

```
> source("C:/Users/lavan/OneDrive/Desktop/DWDM lab/DAY-2/10.R", echo=TRUE)

> # Load libraries
> library(ggplot2)

> # Manually create dataset
> diabetes_data <- data.frame(
+   Age = c(50, 60, 45, 70, 55, 65, 40),
+   BloodPressure = c(80, 120, 85, 130, 100, 110,  .... [TRUNCATED]

> # Scatter plot: Blood Pressure vs Age
> ggplot(diabetes_data, aes(x = Age, y = BloodPressure)) +
+   geom_point() + labs(title = "Blood Pressure vs  ..." ... [TRUNCATED]

> # Bar chart: Average Blood Pressure by Age Group
> diabetes_data$AgeGroup <- cut(diabetes_data$Age, breaks = c(0, 30, 40, 50, 60, 7
0, 80, 100), labe .... [TRUNCATED]

> ggplot(diabetes_data, aes(x = AgeGroup, y = BloodPressure, fill = AgeGroup)) +
+   geom_bar(stat = "summary", fun = "mean", color = "black") +
+   l .... [TRUNCATED]
>
```

Environment | History | Connections | Tutorial

Files | Plots | Packages | Help | Viewer | Presentation

Zoom | Export | Publish




79°F Partly sunny | Search | ENG IN | 09:33 19-11-2024

# DAY-3

**LIST OF PROGRAMS:**

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

| Customer ID | Transaction ID | Items Bought |
|---|---|---|
| 1 | 0001 | $\{a, d, e\}$ |
| 1 | 0024 | $\{a, b, c, e\}$ |
| 2 | 0012 | $\{a, b, d, e\}$ |
| 2 | 0031 | $\{a, c, d, e\}$ |
| 3 | 0015 | $\{b, c, e\}$ |
| 3 | 0022 | $\{b, d, e\}$ |
| 4 | 0029 | $\{c, d\}$ |
| 4 | 0040 | $\{a, b, c\}$ |
| 5 | 0033 | $\{a, d, e\}$ |
| 5 | 0038 | $\{a, b, e\}$ |

**INPUT**

**@relation transactions**

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

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

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

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

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

**@data**

**true,false,false,true,true**

**true,true,true,false,true**

**true,true,false,true,true**

**true,false,true,true,true**

**false,true,true,false,true**

**false,true,false,true,true**

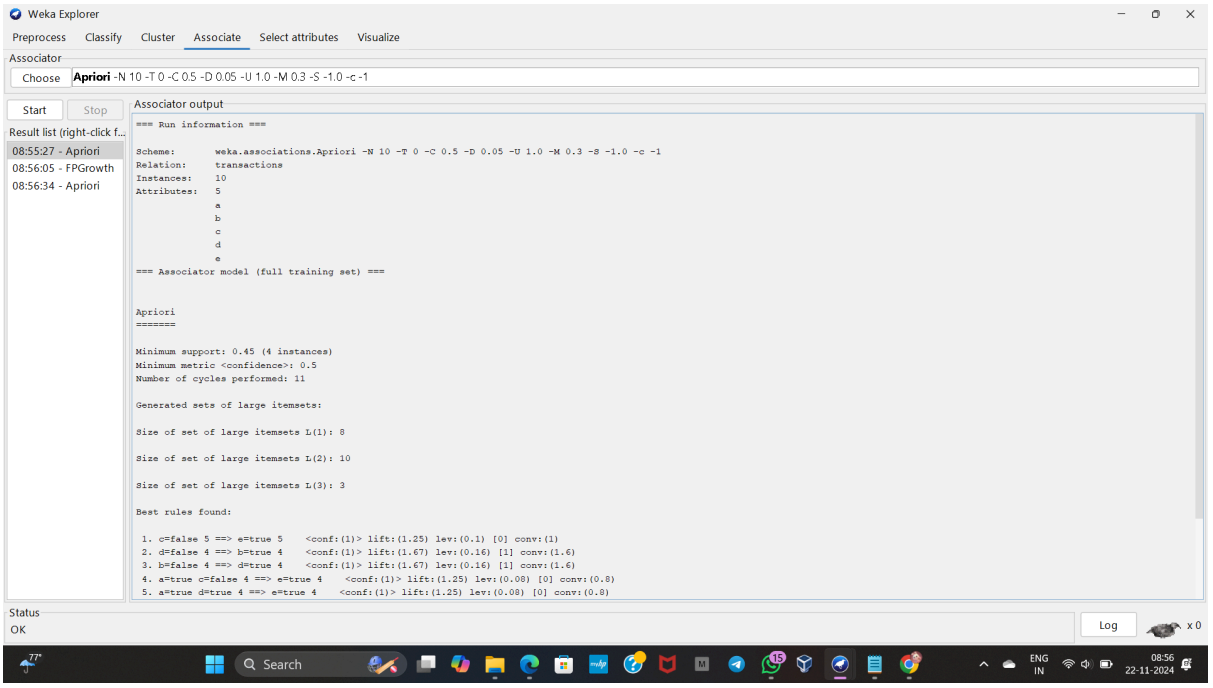
**false,false,true,true,false**

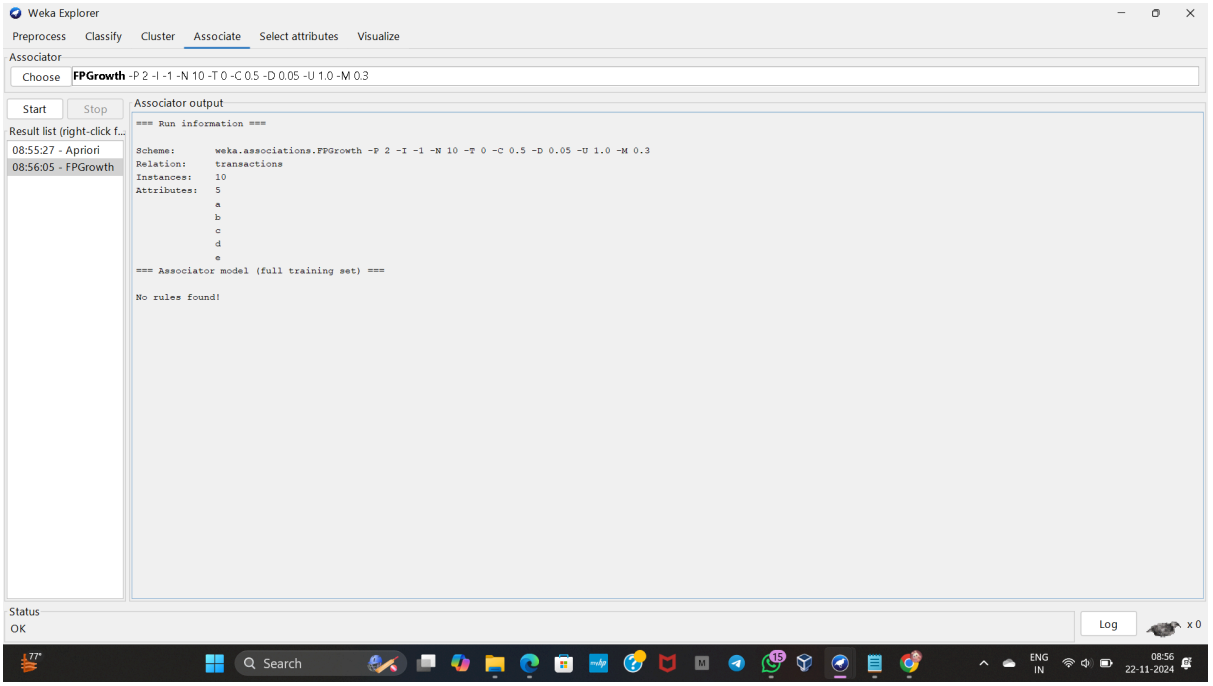
**true,true,true,false,false**

**true,false,false,true,true**

**True,true,false,false,true**

**OUTPUT**






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

**INPUT**

**@relation market_basket**

**@attribute Milk {Yes, No}**

**@attribute Beer {Yes, No}**

**@attribute Diapers {Yes, No}**

**@attribute Bread {Yes, No}**

**@attribute Butter {Yes, No}**

**@attribute Cookies {Yes, No}**

**@data**

**Yes, Yes, Yes, No, No, No**

**Yes, Yes, No, Yes, Yes, No**

**Yes, No, Yes, Yes, No, Yes**

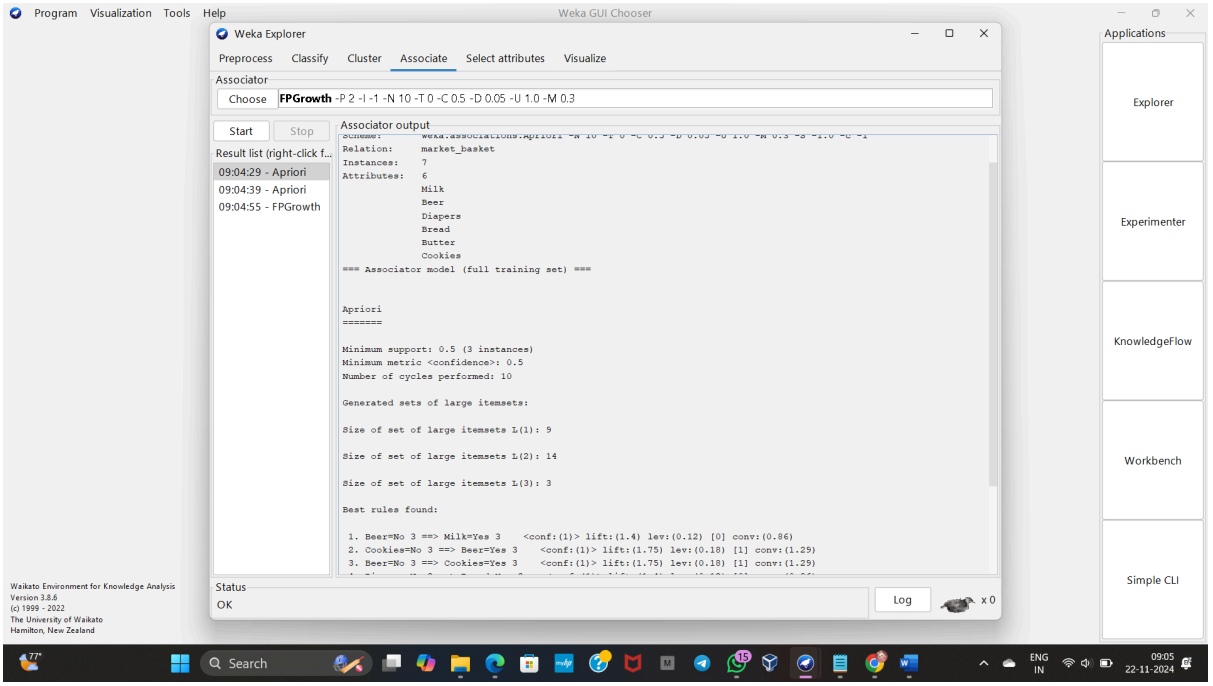
**Yes, No, No, Yes, Yes, Yes**

**No, Yes, Yes, Yes, Yes, No**

**Yes, No, Yes, No, Yes, Yes**

**No, Yes, No, Yes, Yes, Yes**

**OUTPUT**




**3.Consider the market basket transactions shown in the above table.**

**(a) What is the maximum number of association rules that can be extracted**

**from this data (including rules that have zero support)?**

**(b) What is the maximum size of frequent itemsets that can be extracted**

**INPUT**

**(assuming minsup > 0)?**

**@relation market_basket**

**@attribute Milk {Yes, No}**

**@attribute Beer {Yes, No}**

**@attribute Diapers {Yes, No}**

**@attribute Bread {Yes, No}**

**@attribute Butter {Yes, No}**

**@attribute Cookies {Yes, No}**

**@data**

**Yes, Yes, Yes, No, No, No**

**Yes, Yes, No, Yes, Yes, No**

**Yes, No, Yes, Yes, No, Yes**

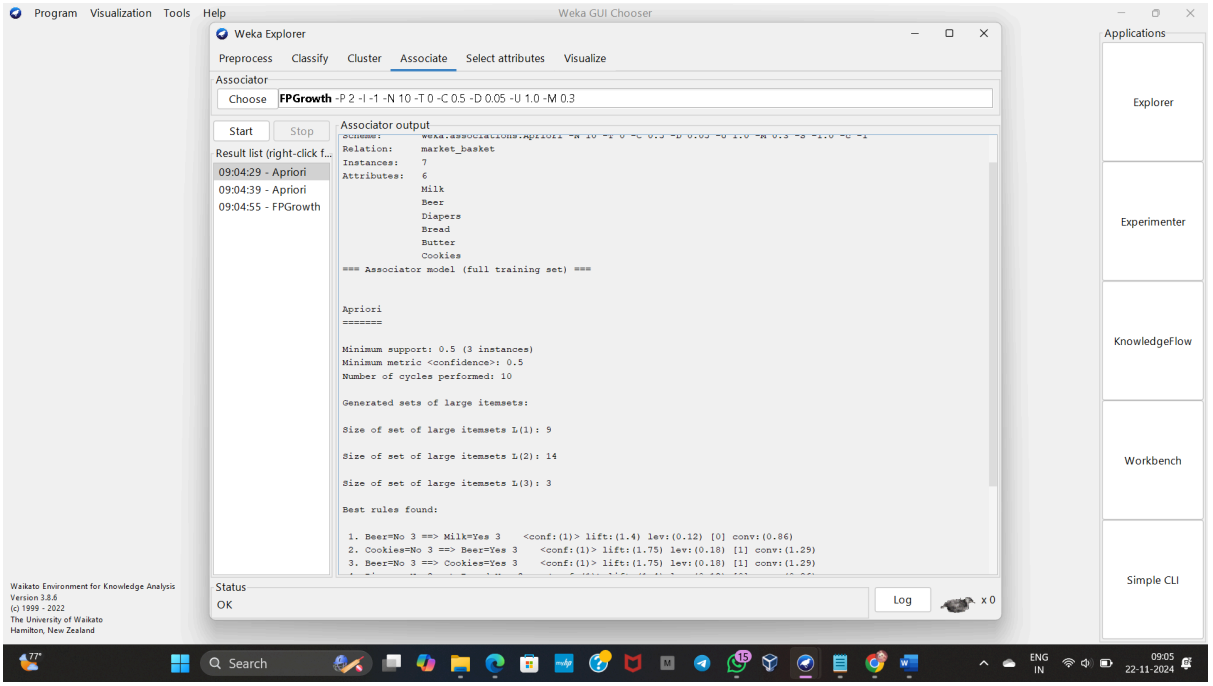
**Yes, No, No, Yes, Yes, Yes**

**No, Yes, Yes, Yes, Yes, No**

**Yes, No, Yes, No, Yes, Yes**

**No, Yes, No, Yes, Yes, Yes**

**OUTPUT**




## 4.Bayes classification and descion tree (using training and test data)

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

**INPUT**

**@relation buys_computer**

**@attribute age {<=30, 31...40, >40}**

**@attribute income {low, medium, high}**

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

**@attribute credit_rating {fair, excellent}**

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

**@data**

**<=30, high, no, fair, no**

**<=30, high, no, excellent, no**

**31...40, high, no, fair, yes**

**>40, medium, no, fair, yes**

**>40, low, yes, fair, yes**

**>40, low, yes, excellent, no**

**31...40, low, yes, excellent, yes**

**<=30, medium, no, fair, no**

**<=30, low, yes, fair, yes**

**>40, medium, yes, fair, yes**

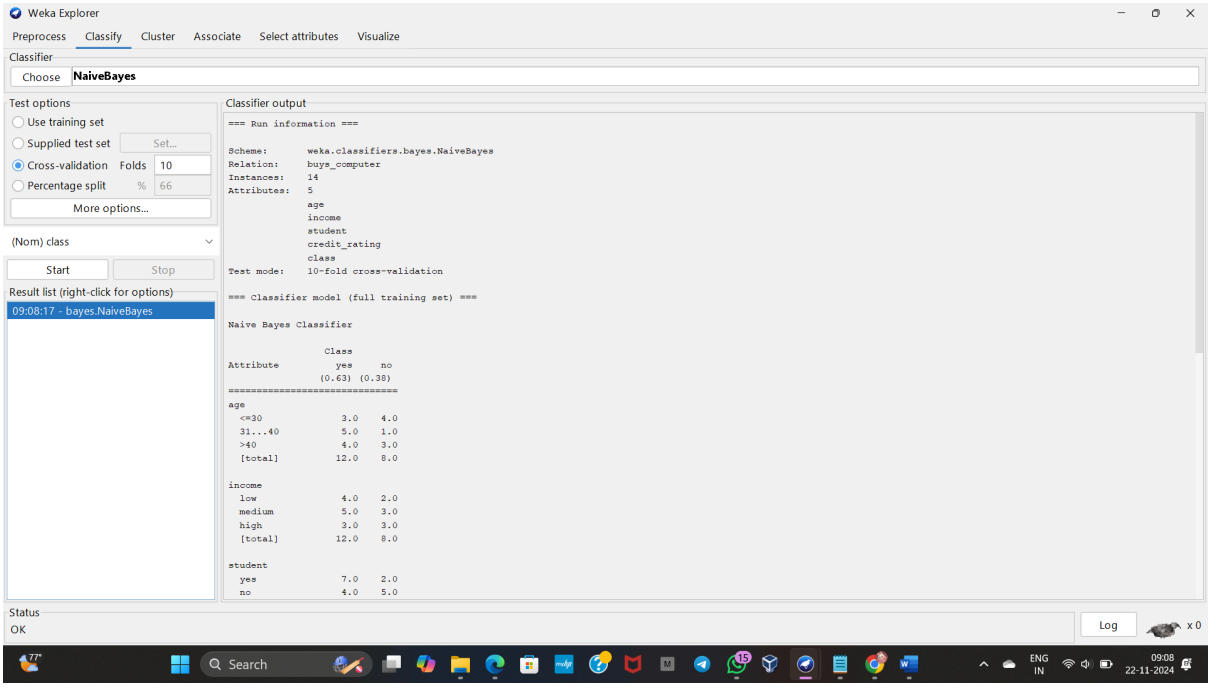
**<=30, medium, yes, excellent, yes**

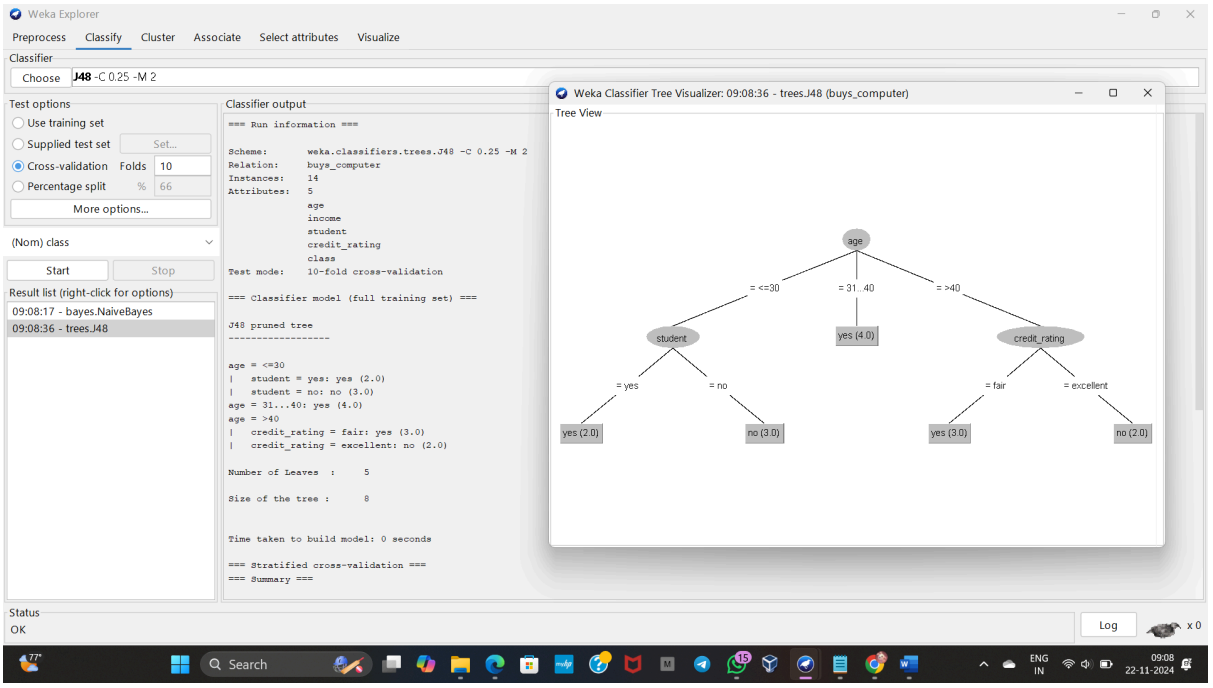
**31...40, medium, no, excellent, yes**

**31...40, high, yes, fair, yes**

**>40, medium, no, excellent, no**

**OUTPUT**




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

| Transactions | Items |
|---|---|
| T1 | (M, O, N, K, E, Y) |
| T2 | (D, O, N, K, E, Y) |
| T3 | (M, A, K, E) |

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

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

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

**INPUT**

**@relation transactions**

**@attribute Item1 {M, O, N, K, E, Y, D, A, U, C, I}**

**@attribute Item2 {M, O, N, K, E, Y, D, A, U, C, I}**

**@attribute Item3 {M, O, N, K, E, Y, D, A, U, C, I}**

**@attribute Item4 {M, O, N, K, E, Y, D, A, U, C, I}**

**@attribute Item5 {M, O, N, K, E, Y, D, A, U, C, I}**

**@data**

**M,O,N,K,E**

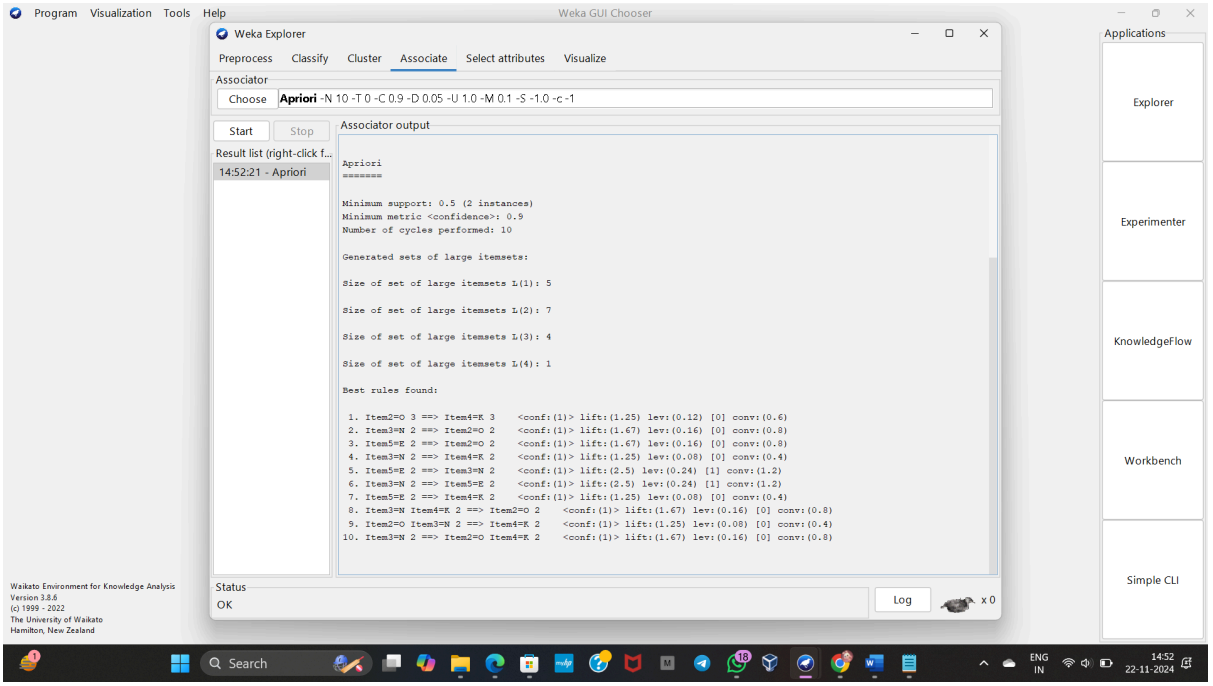
**D,O,N,K,E**

**M,A,K,E,?**

**M,U,C,K,Y**

**C,O,O,K,I**

**OUTPUT**

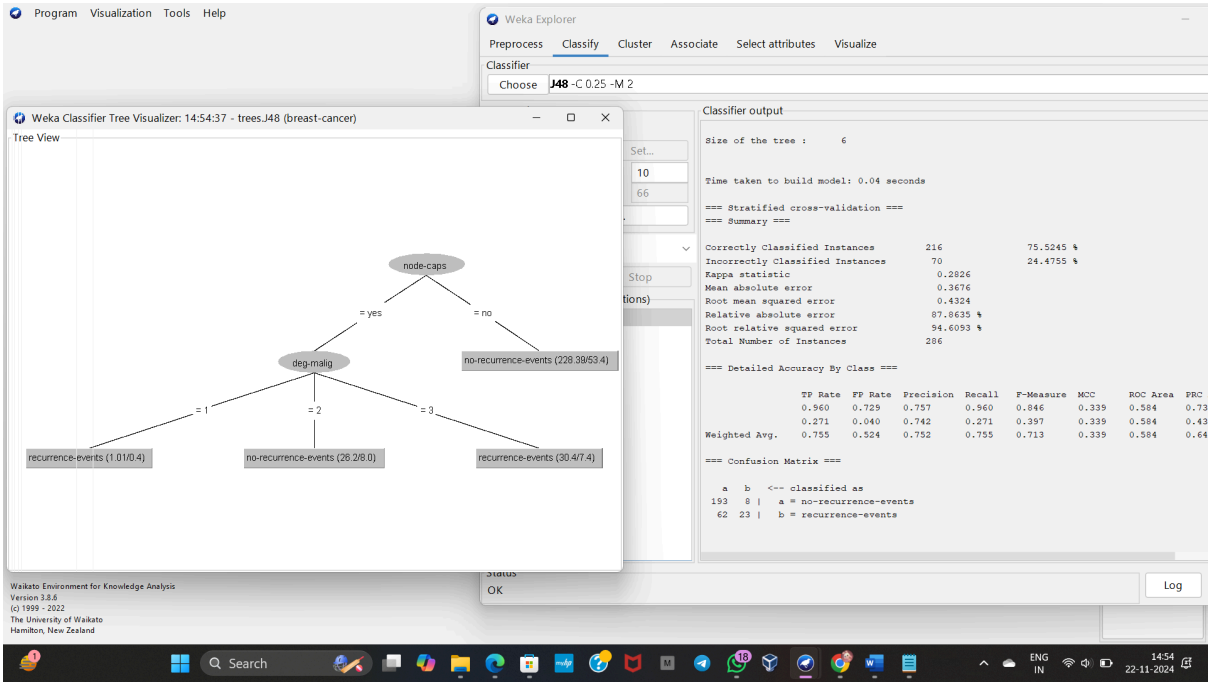


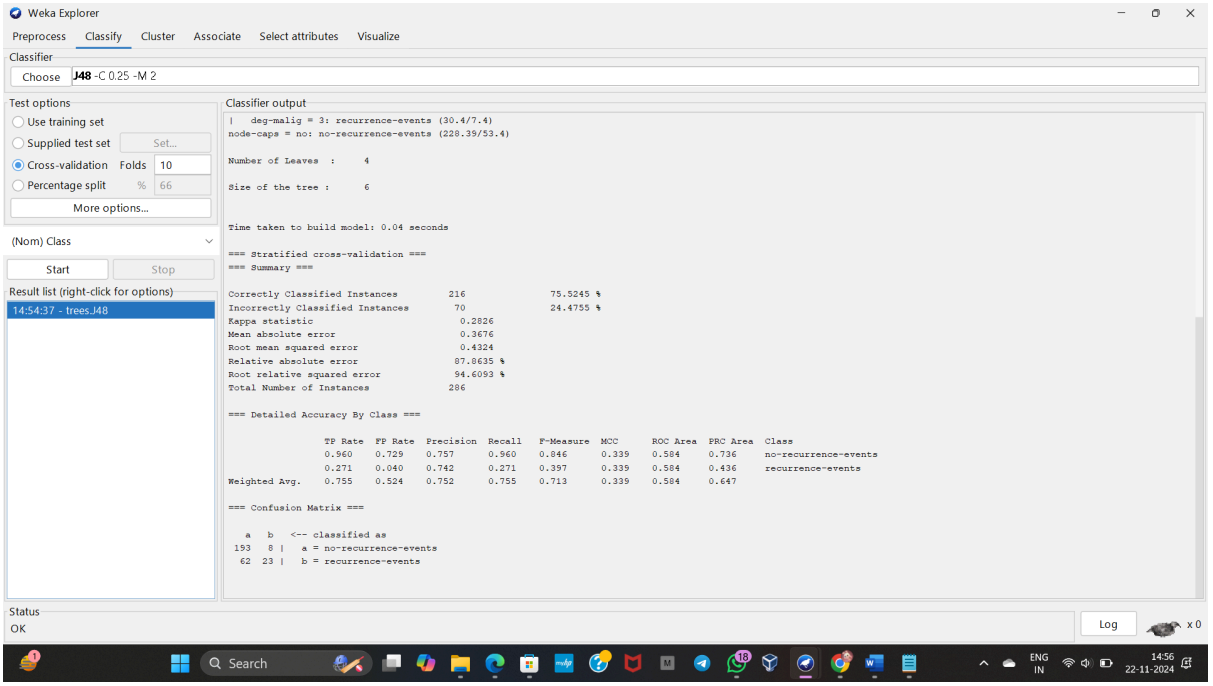

## 6.Prediction of Categorical Data using Decision Tree Algorithm through WEKA using any datasets. a) Tree b) Preprocess c) Logistic

## INPUT

## BREAT CANCER DATASET

## OUTPUT






**7.**

**Create the dataset using ARFF file format:**

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

**b.List the various rule generated by apriori and FP tree algorthim ,mention wheather accepted or rejcted.**

**INPUT**

**@relation transactions**

**@attribute item1 {yes, no}**

**@attribute item2 {yes, no}**

**@attribute item3 {yes, no}**

**@data**

**yes, yes, no**

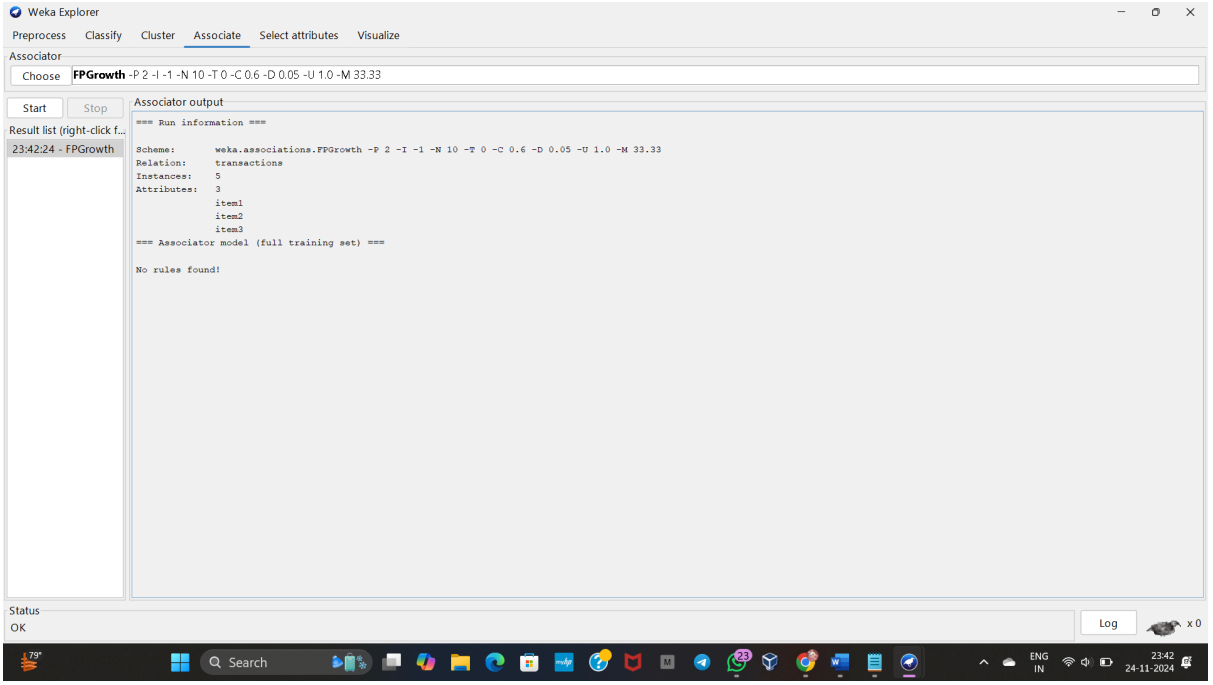
**yes, no, yes**

**no, yes, yes**

**yes, yes, yes**

**no, no, yes**

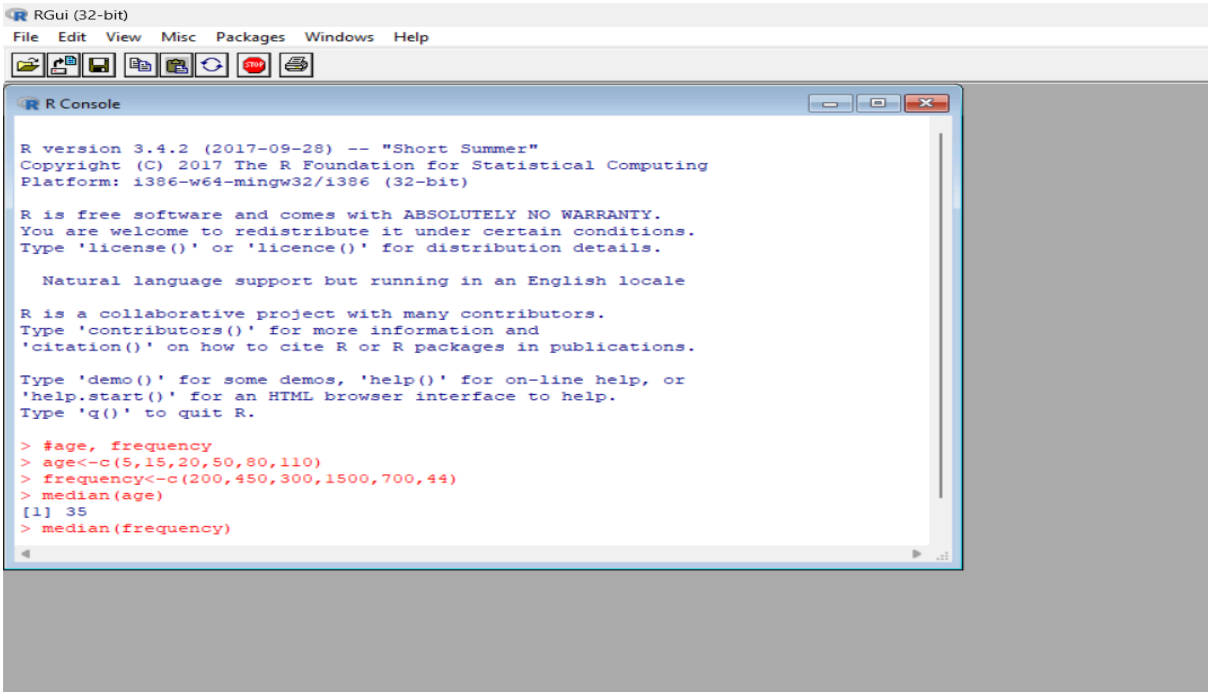
## OUTPUT



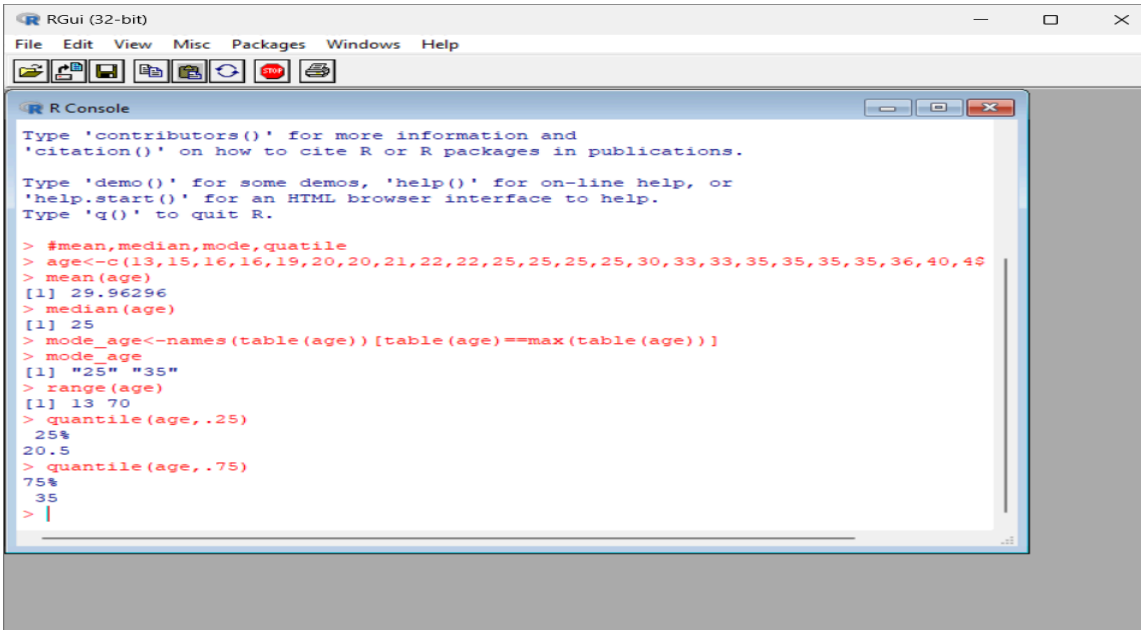

# DAY-4

## LIST OF PROGRAMS:

1


RGui (32-bit)
File Edit View Misc Packages Windows Help
R Console
R version 3.4.2 (2017-09-28) -- "Short Summer"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)
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)
> median(age)
[1] 35
> median(frequency)

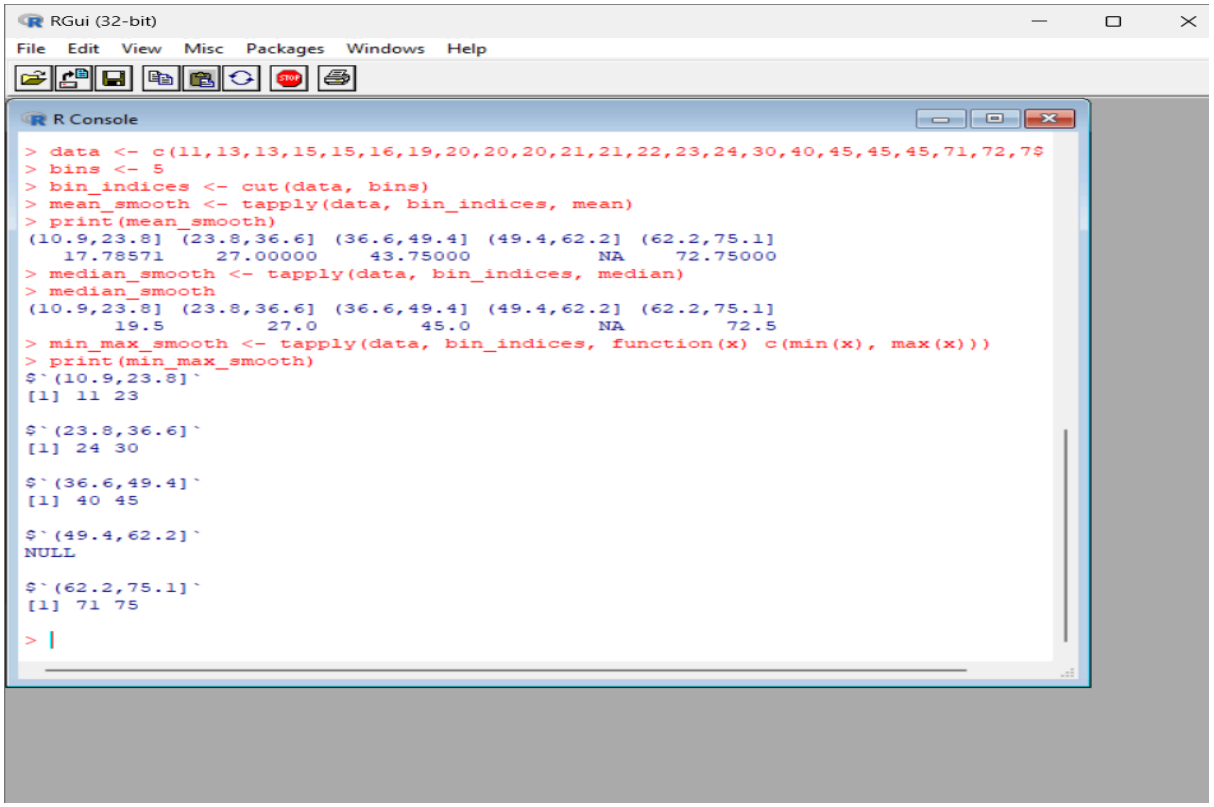
2


RGui (32-bit)
File Edit View Misc Packages Windows Help
R Console
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.
> #mean,median,mode,quatile
> age<-c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,4$
> mean(age)
[1] 29.96296
> median(age)
[1] 25
> 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
>

3

```
> data <- c(11,13,13,15,15,16,19,20,20,20,21,21,22,23,24,30,40,45,45,45,71,72,73,75)
> bins <- 5
> bin_indices <- cut(data, bins)
> mean_smooth <- tapply(data, bin_indices, mean)
> print(mean_smooth)
(10.9,23.8] (23.8,36.6] (36.6,49.4] (49.4,62.2] (62.2,75.1]
   17.78571    27.00000    43.75000          NA    72.75000
> median_smooth <- tapply(data, bin_indices, median)
> median_smooth
(10.9,23.8] (23.8,36.6] (36.6,49.4] (49.4,62.2] (62.2,75.1]
       19.5        27.0        45.0          NA        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

> q
function (save = "default", status = 0, runLast = TRUE)
.Internal(quit(save, status, runLast))
<bytecode: 0x000002913c309348>
<environment: namespace:base>
```

4.




5

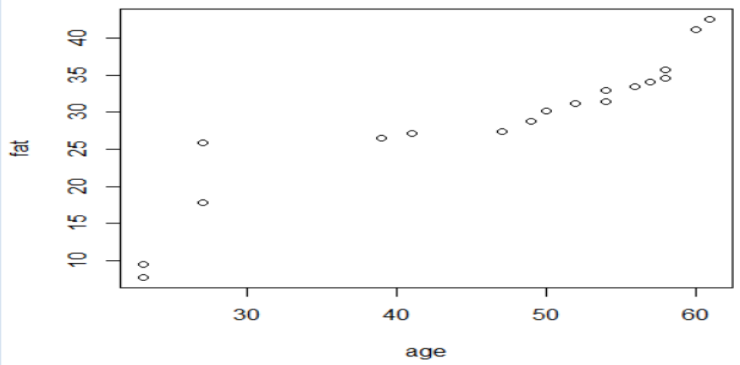
RGui (32-bit)

File History Resize Windows

R Console

```
[1] 40 45

$`(49.4,62.2]`
NULL

$`(62.2,75.1]`
[1] 71 75

> 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$
> 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] 9.254395
> #boxplot
> boxplot(age,fat)
> #scatter plot
> scatter.smooth(age,fat)
> #qplot
> qqplot(age,fat)
>
```

R Graphics: Device 2 (ACTIVE)



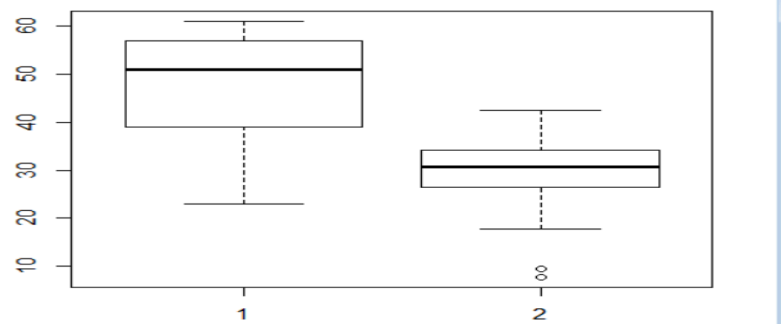

RGui (32-bit)

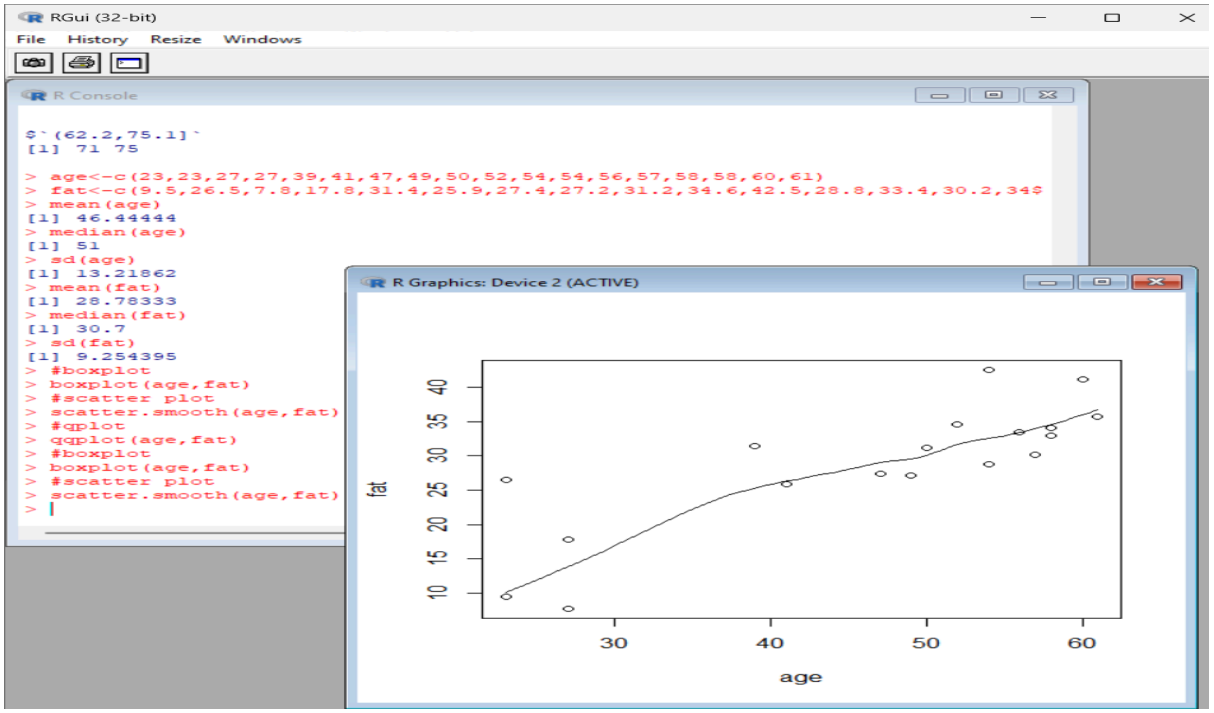
File History Resize Windows

R Console

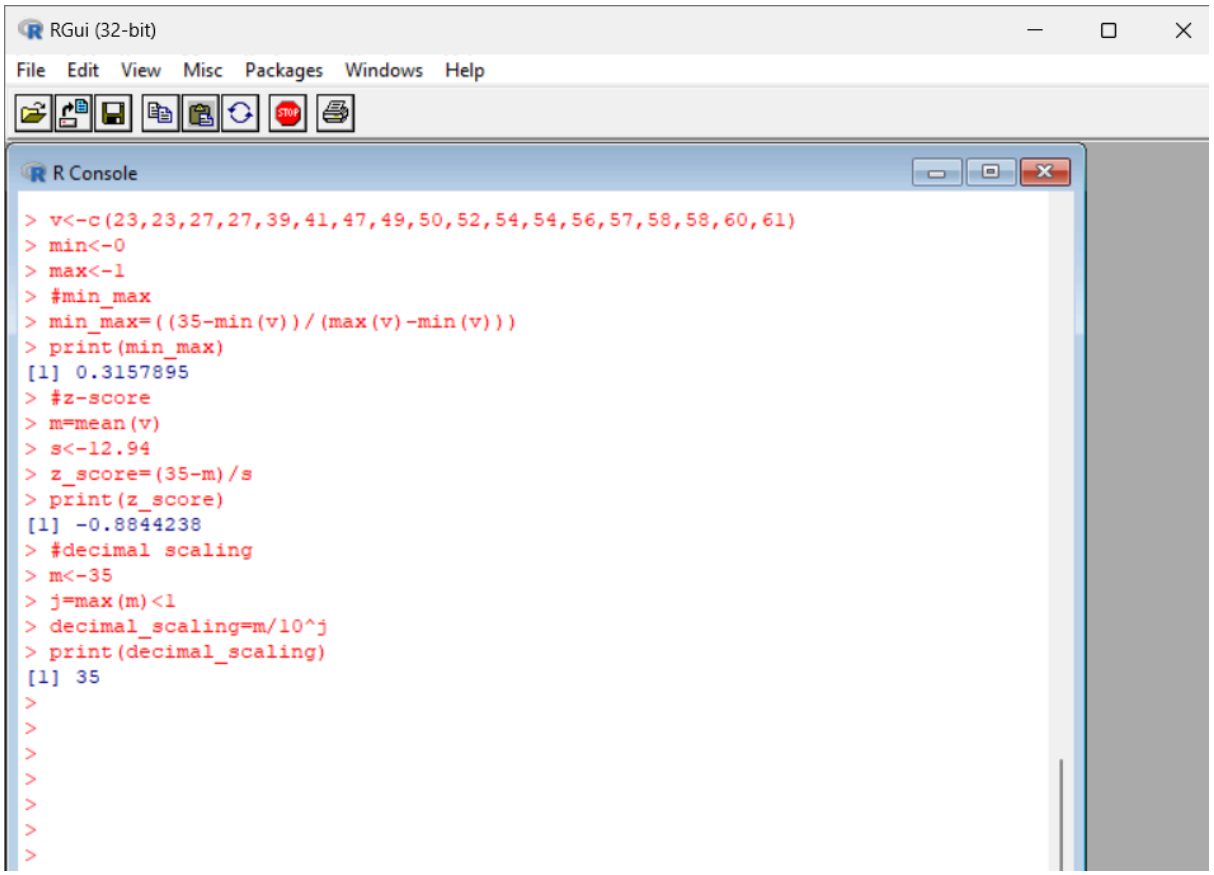
```
$`(49.4,62.2]`
NULL

$`(62.2,75.1]`
[1] 71 75

> 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$
> 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] 9.254395
> #boxplot
> boxplot(age,fat)
> #scatter plot
> scatter.smooth(age,fat)
> #qplot
> qqplot(age,fat)
> #boxplot
> boxplot(age,fat)
>
```

R Graphics: Device 2 (ACTIVE)




RGui (32-bit)
File History Resize Windows
R Console
$`(62.2,75.1]`
[1] 71 75
> age<-c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
> fat<-c(9.5,26.5,7.8,17.8,31.4,25.9,27.4,27.2,31.2,34.6,42.5,28.8,33.4,30.2,34$
> 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] 9.254395
> #boxplot
> boxplot(age,fat)
> #scatter plot
> scatter.smooth(age,fat)
> #qplot
> qqplot(age,fat)
> #boxplot
> boxplot(age,fat)
> #scatter plot
> scatter.smooth(age,fat)
R Graphics: Device 2 (ACTIVE)
fat
age
10
15
20
25
30
35
40
30
40
50
60

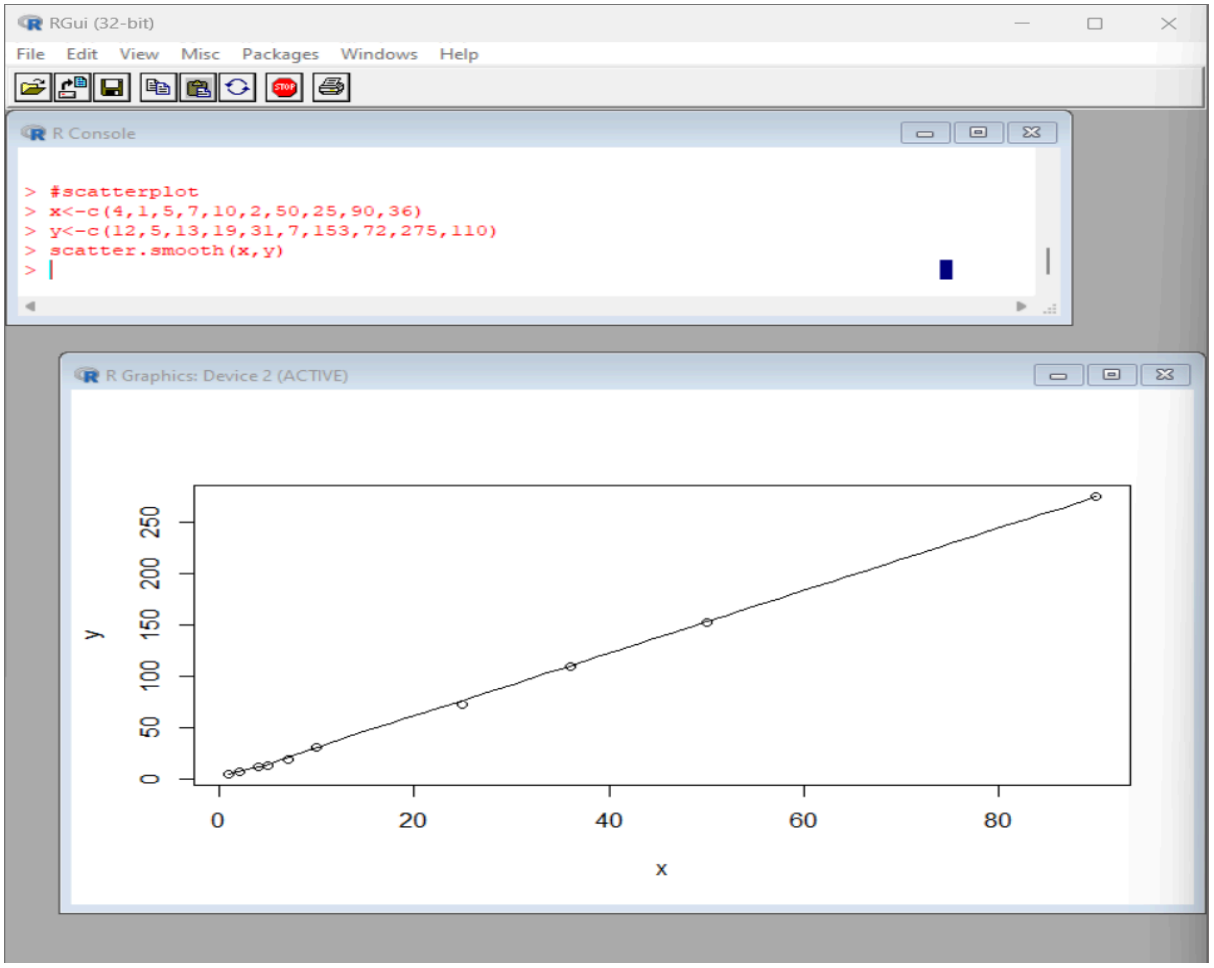
6


RGui (32-bit)
File Edit View Misc Packages Windows Help
R Console
> 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)
[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

7

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

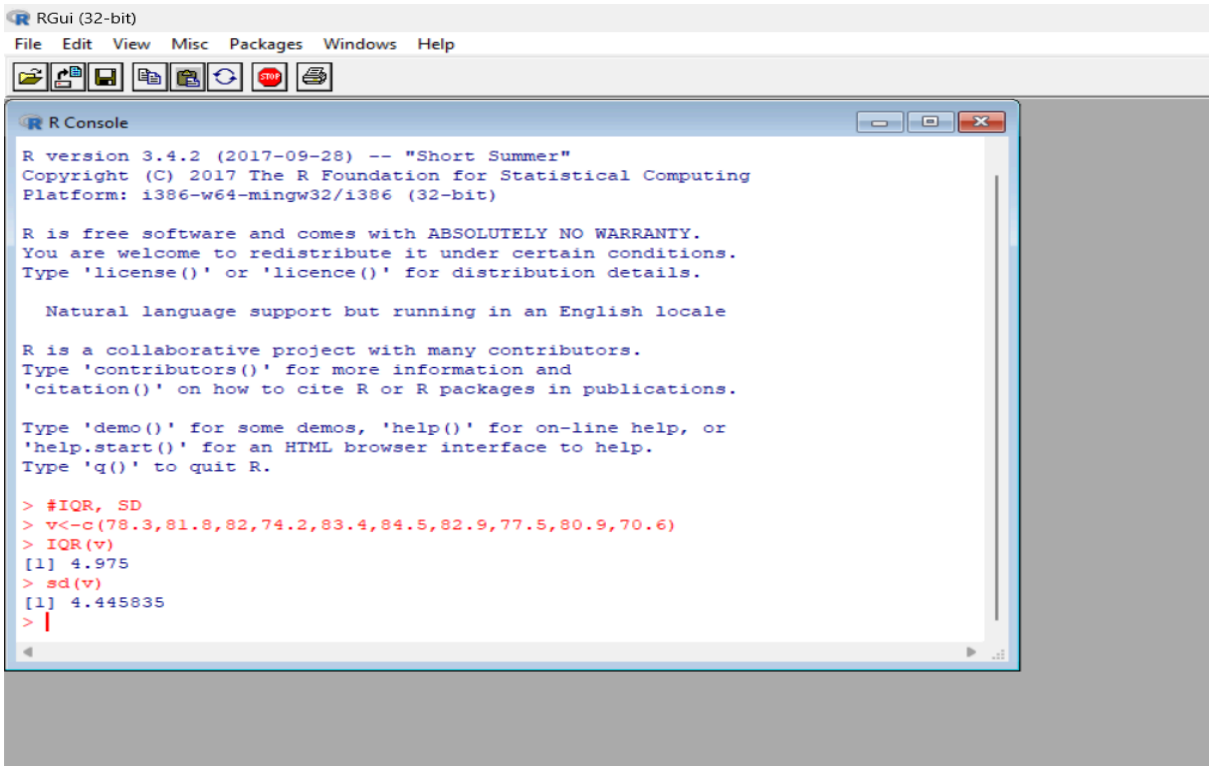
8



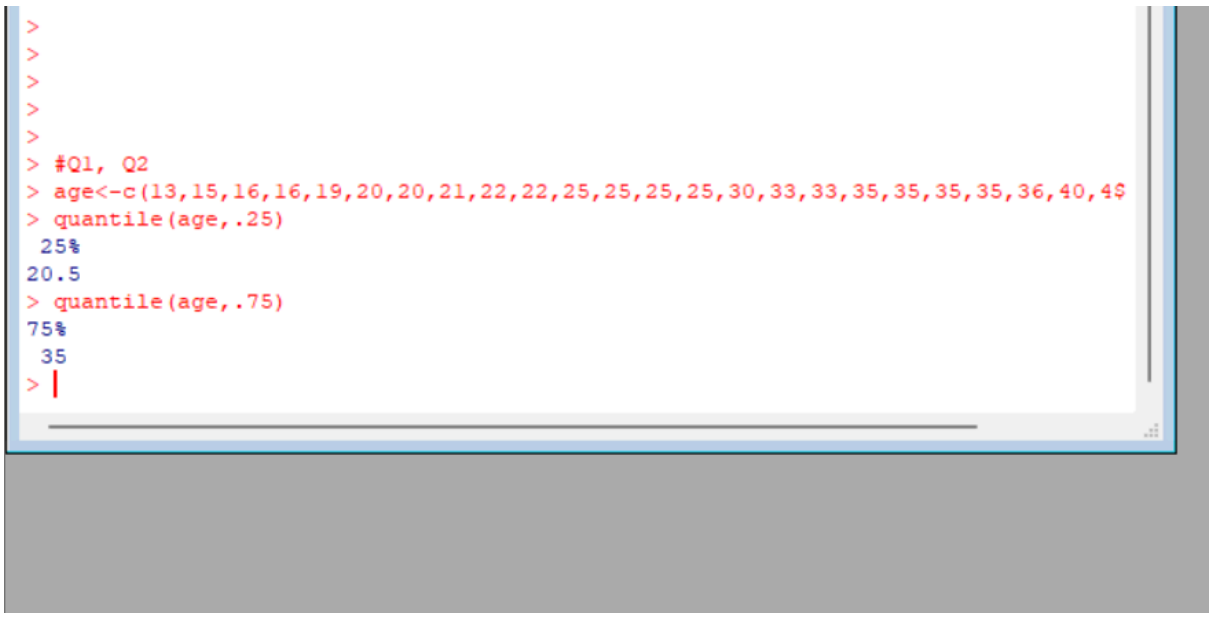

9

```
<environment: namespace:base>
> #IQR, SD
> v<-c(78.3,81.8,82,74.2,83.4,84.5,82.9,77.5,80.9,70.6)
> IQR(v)
[1] 4.975
> sd(v)
[1] 4.445835
```

10




11




1.Covariance and correlation

Children of three ages are asked to indicate their preference for three photographs of adults. Do the data suggest that there is a significant relationship between age and photograph preference? What is wrong with this study?

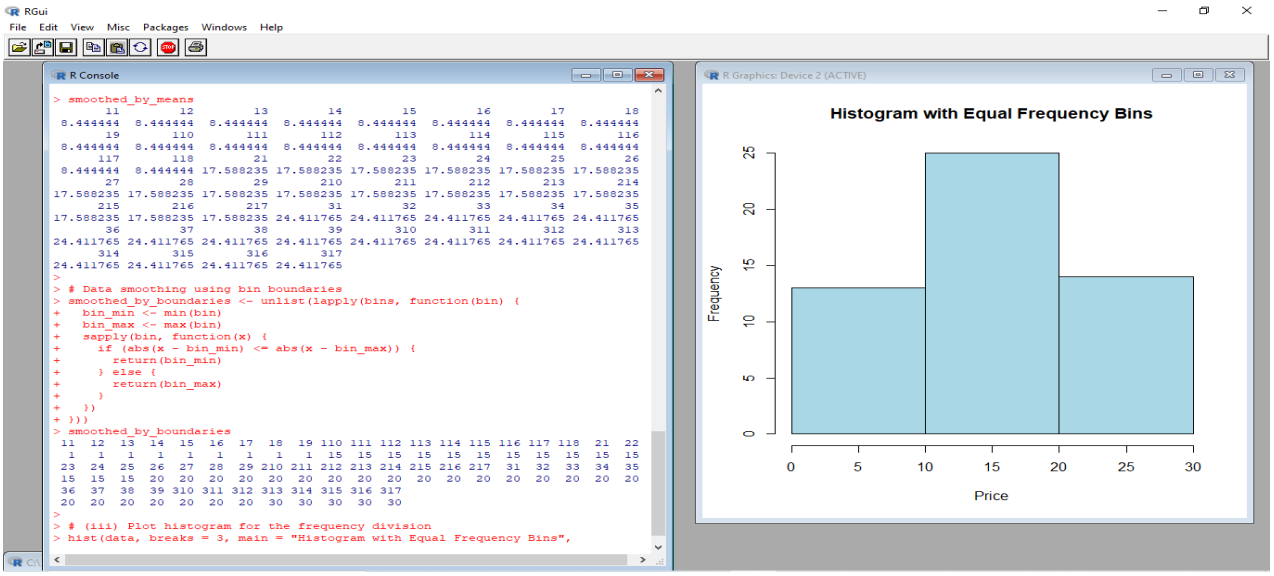
| | **Photograph:** | | |
|---|---|---|---|
| **Age of child** | A | B | C |
| 5-6 years: | 18 | 22 | 20 |
| 7-8 years: | 2 | 28 | 40 |
| 9-10 years: | 20 | 10 | 40 |

1. Use cov() to calculate the sample covariance between B and C.
2. Use another call to cov() to calculate the sample covariance matrix for the preferences.
3. Use cor() to calculate the sample correlation between B and C.
4. Use another call to cor() to calculate the sample correlation matrix for the preferences.

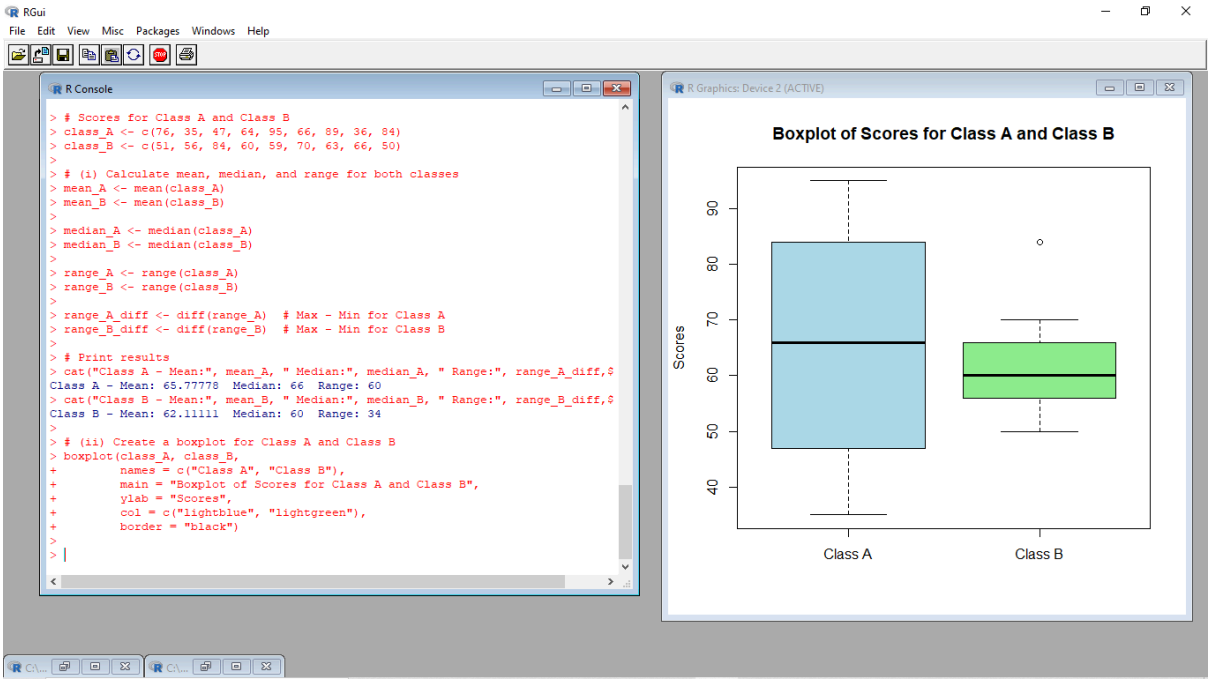
2.Imagine that you have selected data from the All Electronics data warehouse for analysis. The data set will be huge! The following data are a list of All Electronics prices for commonly sold items (rounded to the nearest dollar). The numbers have been sorted: 1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18,

| |
|---|
| 0, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, |
| dataset using an equal-frequency partitioning method with bin equal to 3 (ii) apply data smoothing using bin means and bin boundary.<br>(iii) Plot Histogram for the above frequency division |

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




exams. Their results are as follows:
Class A: 76, 35, 47, 64, 95, 66, 89, 36, 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
Class B: 51, 56, 84, 60, 59, 70, 63, 66, 5051,56,84,60,59,70,63,66,50




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

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

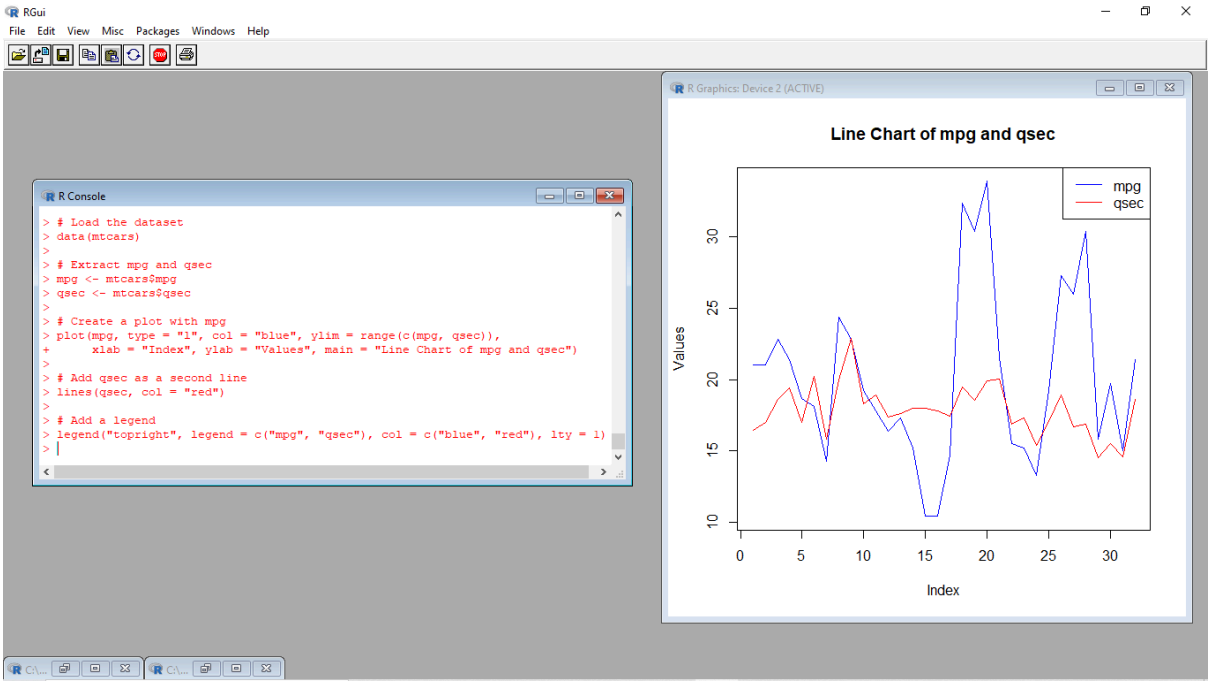
(a) min-max normalization by setting min = 0 and max = 1

(b) z-score normalization

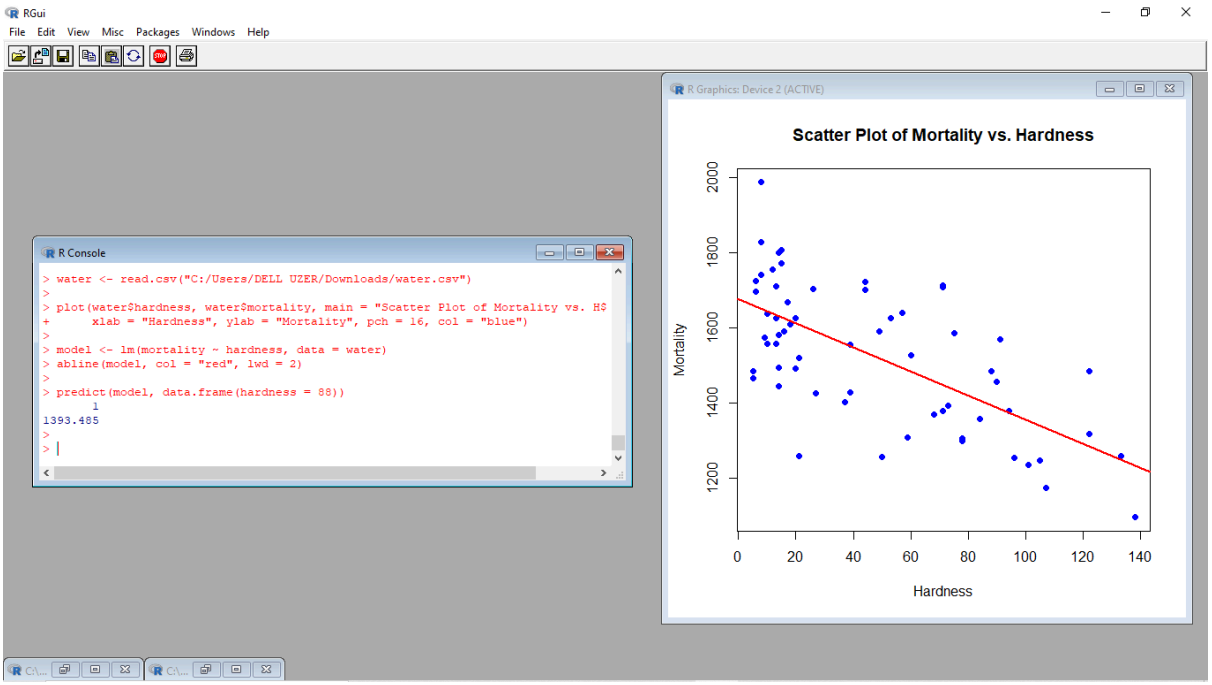
```
> # Data
> data <- c(200, 300, 400, 600, 1000)
>
> # (a) Min-Max Normalization (min = 0, max = 1)
> min_value <- min(data)
> max_value <- max(data)
>
> min_max_normalized <- (data - min_value) / (max_value - min_value)
> cat("Min-Max Normalization:\n", min_max_normalized, "\n")
Min-Max Normalization:
 0 0.125 0.25 0.5 1
>
> # (b) Z-Score Normalization
> mean_value <- mean(data)
> sd_value <- sd(data)
>
> z_score_normalized <- (data - mean_value) / sd_value
> cat("Z-Score Normalization:\n", z_score_normalized, "\n")
Z-Score Normalization:
 -0.9486833 -0.6324555 -0.3162278 0.3162278 1.581139
>
```

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

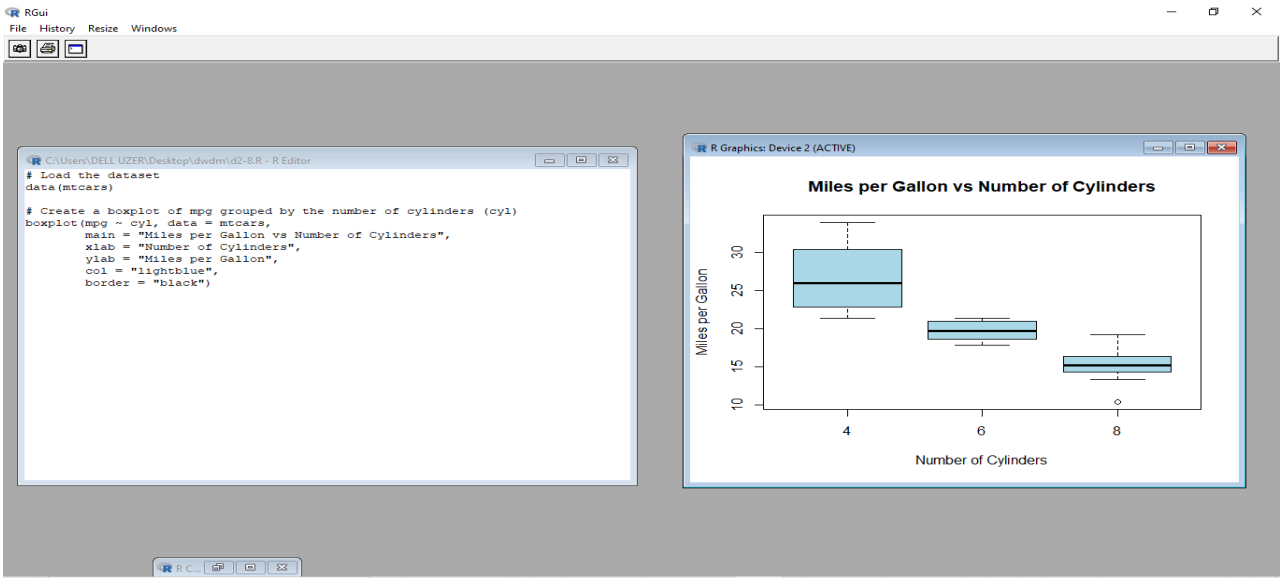

6.Obtain Multiple Lines in Line Chart using a single Plot Function in R.Use attributes“mpg”and“qsec”of the dataset “mtcars”



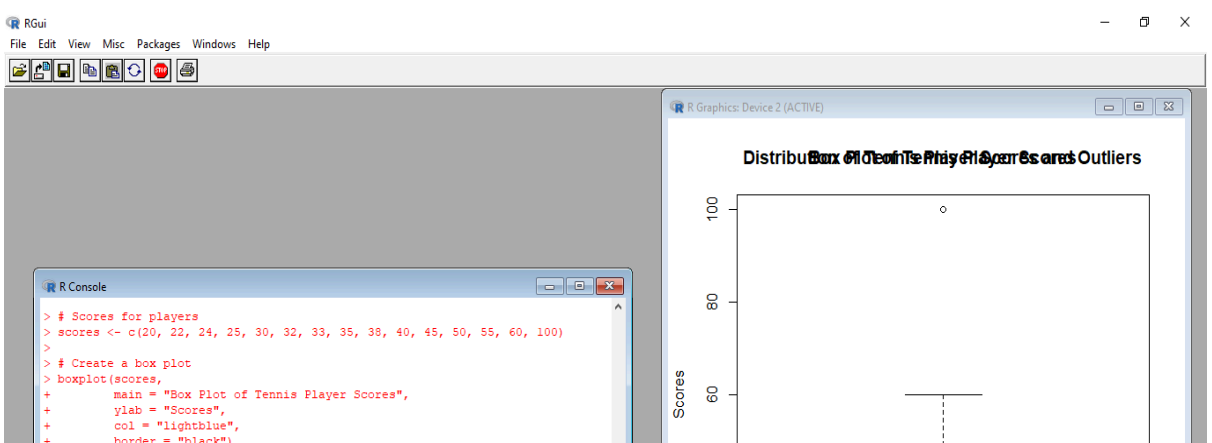

7.Download the Dataset "water" From R dataset Link.Find out whether there is a linear relation between attributes"mortality" and"hardness" by plot function.Fit the Data into the Linear Regression model.Predict the mortality for the hardness=88.



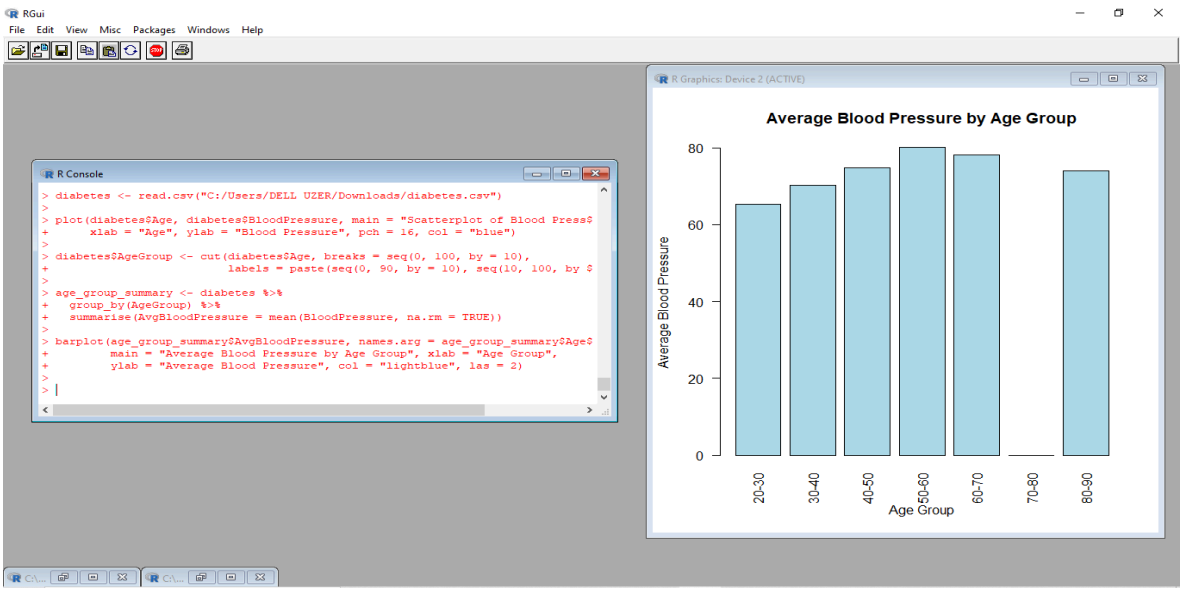

**8.Create a Boxplot graph for the relation between "mpg"(miles per galloon) and "cyl"(number of Cylinders) for the dataset "mtcars" available in R Environment.**




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



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




**1.**

@relation dataset

@attribute a{true,false}

@attribute b{true,false}

@attribute c{true,false}

@attribute d{true,false}

@attribute e{true,false}

@data

true false false true true

true true true false true

true true false true true

true false true true true

false true true false true

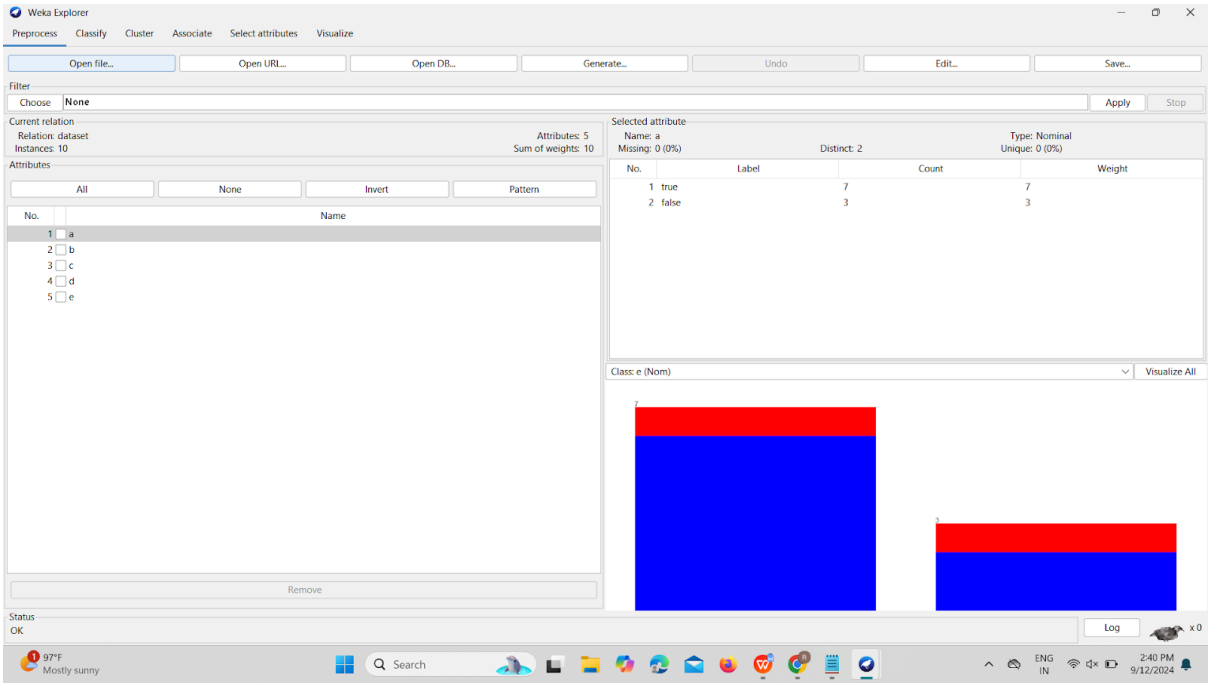
false true false true true

false false true true false

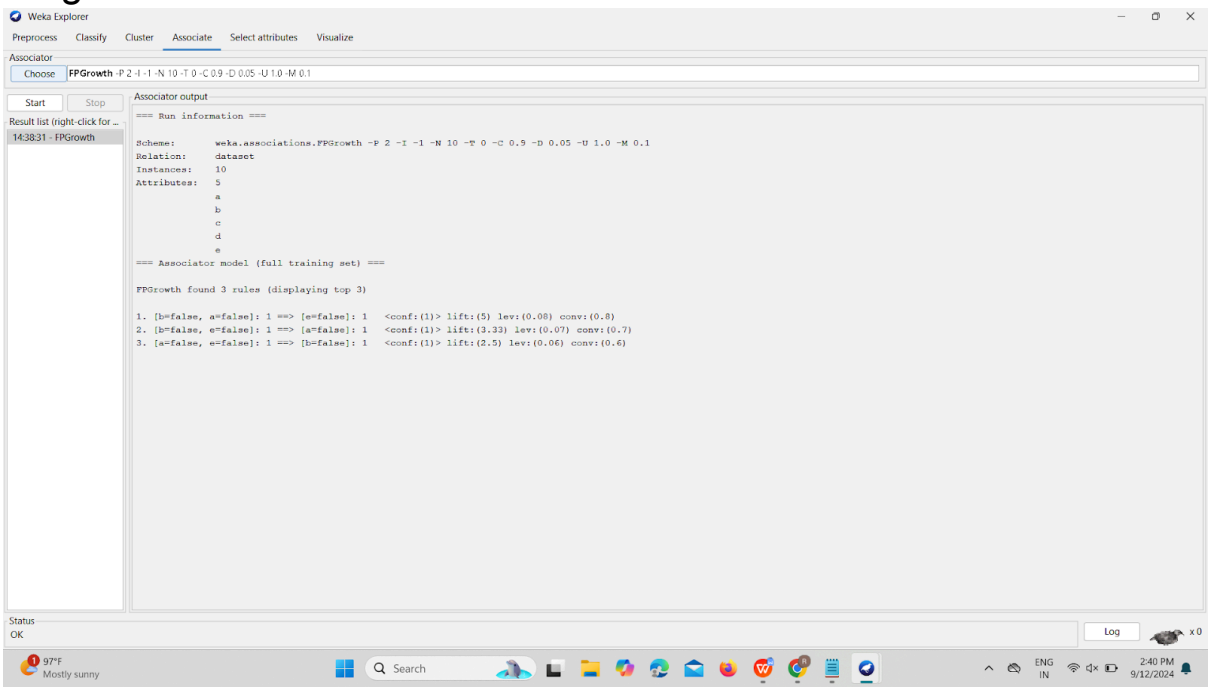
true true true false false

true false false true true

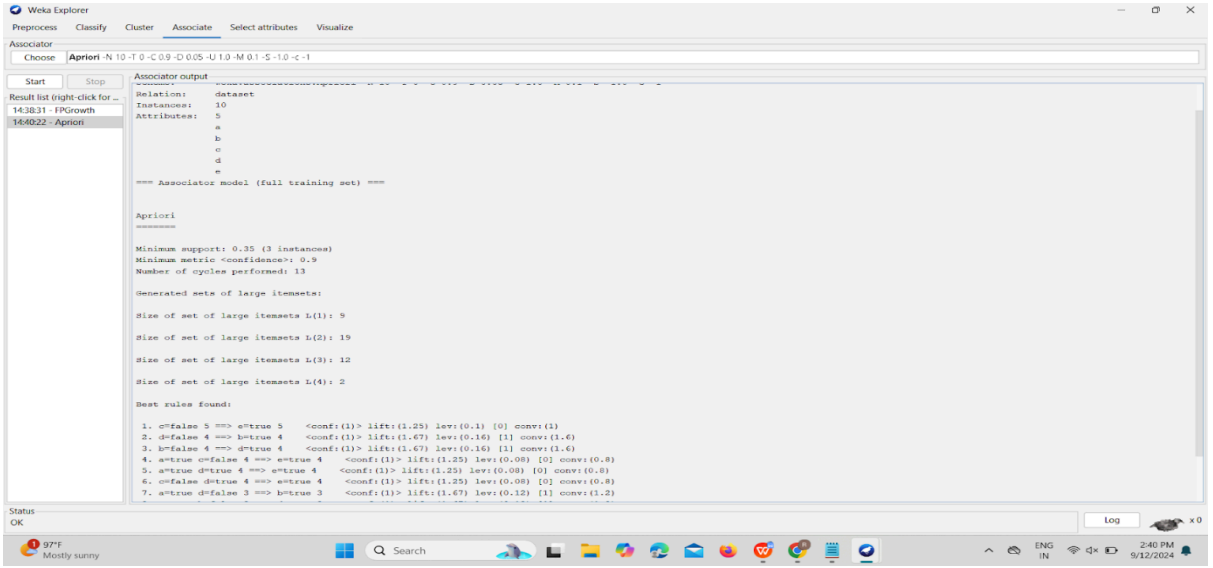
true true false false true




# FP growth:




# Apriori Algorithm:




## 2.

@relation dataset

@attribute Milk{true,false}

@attribute Bread{true,false}

@attribute Diapers{true,false}

@attribute Butter{true,false}

@attribute Beer{true,false}

@data

true false false true true

true true true false true

true true false true true

true false true true true

false true true false true

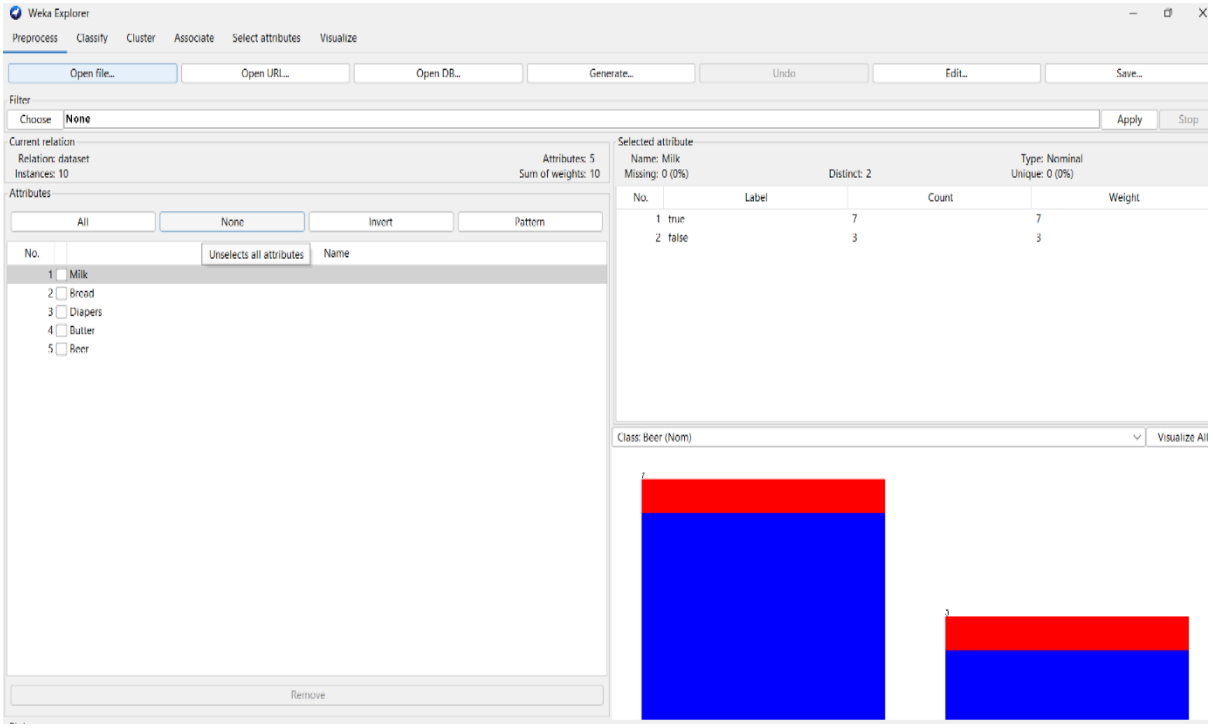
false true false true true

false false true true false

true true true false false

true false false true true

true true false false true




## Apriori Algorithm:

```
Apriori
=======

Minimum support: 0.35 (3 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 13

Generated sets of large itemsets:

Size of set of large itemsets L(1): 9

Size of set of large itemsets L(2): 19

Size of set of large itemsets L(3): 12

Size of set of large itemsets L(4): 2

Best rules found:

 1. Diapers=false 5 ==> Beer=true 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
 2. Butter=false 4 ==> Bread=true 4    <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
 3. Bread=false 4 ==> Butter=true 4    <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
 4. Milk=true Diapers=false 4 ==> Beer=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 5. Milk=true Butter=true 4 ==> Beer=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 6. Diapers=false Butter=true 4 ==> Beer=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 7. Milk=true Butter=false 3 ==> Bread=true 3    <conf:(1)> lift:(1.67) lev:(0.12) [1] conv:(1.2)
 8. Milk=true Bread=false 3 ==> Butter=true 3    <conf:(1)> lift:(1.67) lev:(0.12) [1] conv:(1.2)
 9. Bread=false Beer=true 3 ==> Milk=true 3    <conf:(1)> lift:(1.43) lev:(0.09) [0] conv:(0.9)
10. Milk=true Bread=false 3 ==> Beer=true 3    <conf:(1)> lift:(1.25) lev:(0.06) [0] conv:(0.6)
```

## FP growth:

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
              Milk
              Bread
              Diapers
              Butter
              Beer
=== Associator model (full training set) ===

FPGrowth found 3 rules (displaying top 3)

1. [Bread=false, Milk=false]: 1 ==> [Beer=false]: 1   <conf:(1)> lift:(5) lev:(0.08) conv:(0.8)
2. [Bread=false, Beer=false]: 1 ==> [Milk=false]: 1   <conf:(1)> lift:(3.33) lev:(0.07) conv:(0.7)
3. [Milk=false, Beer=false]: 1 ==> [Bread=false]: 1   <conf:(1)> lift:(2.5) lev:(0.06) conv:(0.6)
```

**4.** @relation dataset

@attribute Monkey{true,false}

@attribute donkey{true,false}

@attribute make{true,false}

@attribute mucky{true,false}

@attribute cookie{true,false}

@data

true false false true true

true true true false true

true true false true true

true false true true true

false true true false true

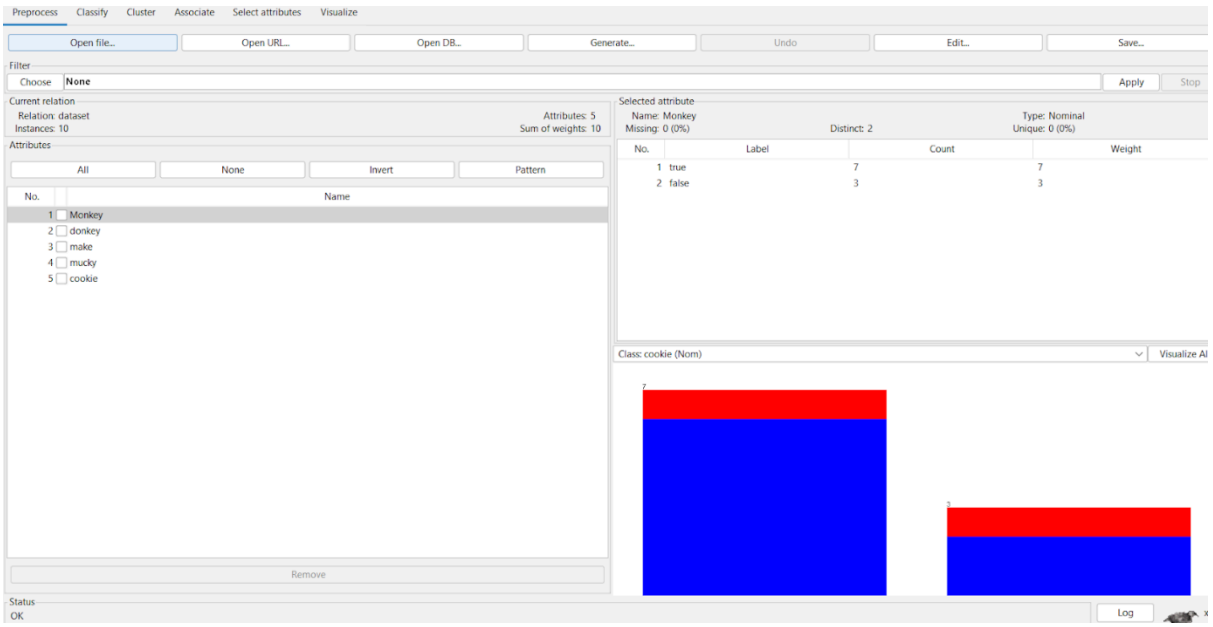
false true false true true

false false true true false

true true true false false

true false false true true

true true false false true




## Apriori Algorithm:

```
Apriori
=======

Minimum support: 0.35 (3 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 13

Generated sets of large itemsets:

Size of set of large itemsets L(1): 9

Size of set of large itemsets L(2): 19

Size of set of large itemsets L(3): 12

Size of set of large itemsets L(4): 2

Best rules found:

 1. make=false 5 ==> cookie=true 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
 2. mucky=false 4 ==> donkey=true 4    <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
 3. donkey=false 4 ==> mucky=true 4    <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
 4. Monkey=true make=false 4 ==> cookie=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 5. Monkey=true mucky=true 4 ==> cookie=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 6. make=false mucky=true 4 ==> cookie=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 7. Monkey=true mucky=false 3 ==> donkey=true 3    <conf:(1)> lift:(1.67) lev:(0.12) [1] conv:(1.2)
 8. Monkey=true donkey=false 3 ==> mucky=true 3    <conf:(1)> lift:(1.67) lev:(0.12) [1] conv:(1.2)
 9. donkey=false cookie=true 3 ==> Monkey=true 3    <conf:(1)> lift:(1.43) lev:(0.09) [0] conv:(0.9)
10. Monkey=true donkey=false 3 ==> cookie=true 3    <conf:(1)> lift:(1.25) lev:(0.06) [0] conv:(0.6)
```

## FP growth:

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
              Monkey
              donkey
              make
              mucky
              cookie
=== Associator model (full training set) ===

FPGrowth found 3 rules (displaying top 3)

1. [donkey=false, Monkey=false]: 1 ==> [cookie=false]: 1   <conf:(1)> lift:(5) lev:(0.08) conv:(0.8)
2. [donkey=false, cookie=false]: 1 ==> [Monkey=false]: 1   <conf:(1)> lift:(3.33) lev:(0.07) conv:(0.7)
3. [Monkey=false, cookie=false]: 1 ==> [donkey=false]: 1   <conf:(1)> lift:(2.5) lev:(0.06) conv:(0.6)
```

**5.**@relation dataset

@attribute hotdogs{true,false}

@attribute buns{true,false}

@attribute coke{true,false}

@attribute chip{true,false}

@attribute ketchup{true,false}

@data

true false false true true

true true true false true

true true false true true

true false true true true

false true true false true

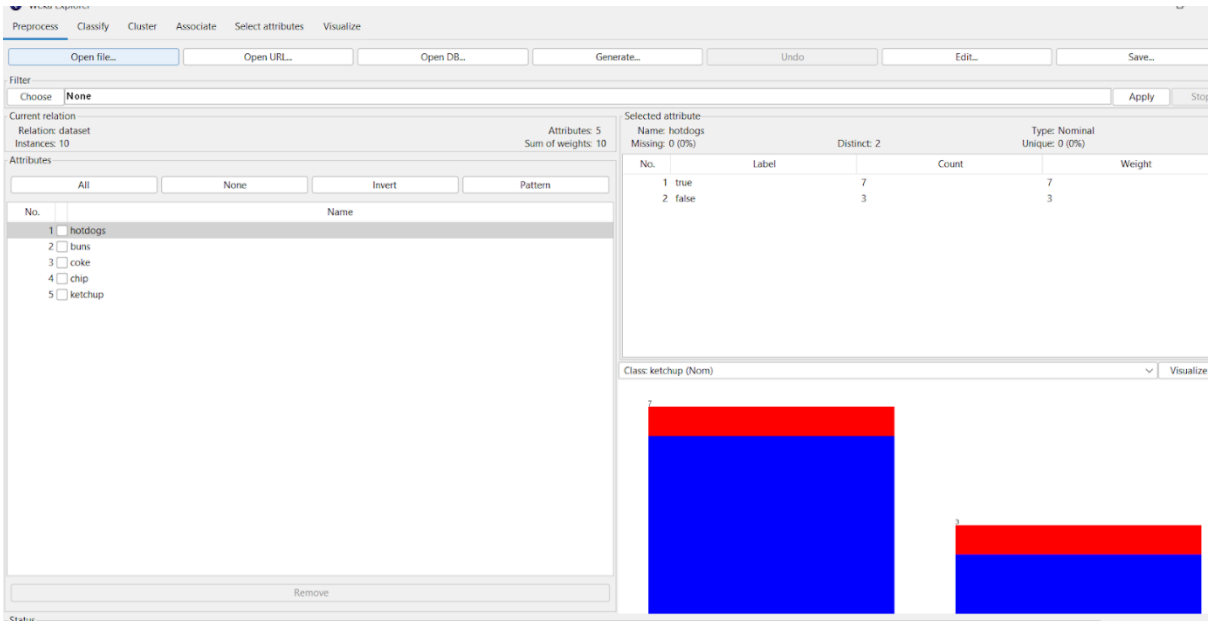
false true false true true

false false true true false

true true true false false

true false false true true

true true false false true




# Apriori algorithm:

```
Apriori
=======

Minimum support: 0.35 (3 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 13

Generated sets of large itemsets:

Size of set of large itemsets L(1): 9

Size of set of large itemsets L(2): 19

Size of set of large itemsets L(3): 12

Size of set of large itemsets L(4): 2

Best rules found:

 1. coke=false 5 ==> ketchup=true 5     <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
 2. chip=false 4 ==> buns=true 4     <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
 3. buns=false 4 ==> chip=true 4     <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
 4. hotdogs=true coke=false 4 ==> ketchup=true 4     <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 5. hotdogs=true chip=true 4 ==> ketchup=true 4     <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 6. coke=false chip=true 4 ==> ketchup=true 4     <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
 7. hotdogs=true chip=false 3 ==> buns=true 3     <conf:(1)> lift:(1.67) lev:(0.12) [1] conv:(1.2)
 8. hotdogs=true buns=false 3 ==> chip=true 3     <conf:(1)> lift:(1.67) lev:(0.12) [1] conv:(1.2)
 9. buns=false ketchup=true 3 ==> hotdogs=true 3     <conf:(1)> lift:(1.43) lev:(0.09) [0] conv:(0.9)
10. hotdogs=true buns=false 3 ==> ketchup=true 3     <conf:(1)> lift:(1.25) lev:(0.06) [0] conv:(0.6)
```

## FP growth:

```
=== 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
              hotdogs
              buns
              coke
              chip
              ketchup
=== Associator model (full training set) ===

FPGrowth found 3 rules (displaying top 3)

1. [buns=false, hotdogs=false]: 1 ==> [ketchup=false]: 1   <conf:(1)> lift:(5) lev:(0.08) conv:(0.8)
2. [buns=false, ketchup=false]: 1 ==> [hotdogs=false]: 1   <conf:(1)> lift:(3.33) lev:(0.07) conv:(0.7)
3. [hotdogs=false, ketchup=false]: 1 ==> [buns=false]: 1   <conf:(1)> lift:(2.5) lev:(0.06) conv:(0.6)
```