7. PREPROCESSING   OF DATA USING WEKA

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

se None   Apply   Stop

relation

on: weather   Attributes: 5
ses: 14   Sum of weights: 14

All	None	Invert	Pattern

Name
1 ☐ outlook
2 ☐ temperature
3 ☐ humidity
4 ☐ windy
5 ☐ play
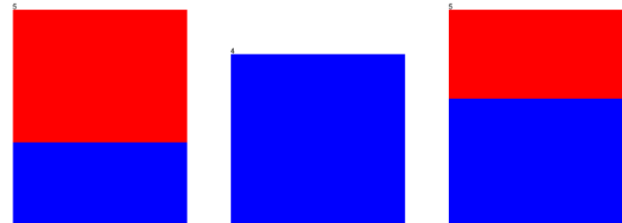
Selected attribute
Name: outlook   Distinct: 3   Type: Nominal
Missing: 0 (0%)   Unique: 0 (0%)

No.	Label	Count	Weight
1	sunny	5	5
2	overcast	4	4
3	rainy	5	5

Class: play (Nom)   Visualize A

Remove

8. K-MEANS CLUSTERING BY WEKA

se **SimpleKMeans** -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 3 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10

mode   Clusterer output

training set

plied test set   Set...

centage split   %   66

ses to clusters evaluation

om) class

re clusters for visualization

Ignore attributes

Start	Stop

st (right-click for options)

5 - SimpleKMeans
4 - SimpleKMeans

```
Attribute        Full Data        0          1          2
                  (150.0)      (61.0)     (50.0)     (39.0)
=====================================================================
sepallength        5.8433      5.8885     5.006      6.8462
sepalwidth         3.054       2.7377     3.418      3.0821
petallength        3.7587      4.3967     1.464      5.7026
petalwidth         1.1987      1.418      0.244      2.0795



Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0        61 ( 41%)
1        50 ( 33%)
2        39 ( 26%)


Class attribute: class
Classes to Clusters:

  0  1  2  <-- assigned to cluster
  0 50  0 | Iris-setosa
 47  0  3 | Iris-versicolor
 14  0 36 | Iris-virginica

Cluster 0 <-- Iris-versicolor
Cluster 1 <-- Iris-setosa
Cluster 2 <-- Iris-virginica

Incorrectly clustered instances :      17.0     11.3333 %
```

OUTPUT INFORMATION OF SIMPLE K MEANS CLUSTERING –CLUSTERER OUTPUT

K MEANS CLUSTERING IN WEKAA USING IRIS INFORMATION – PLOTTING ..

9. DATA ANALYSIS BY EXPECTION MAXIMISATION ALGORITHM USING WEKA
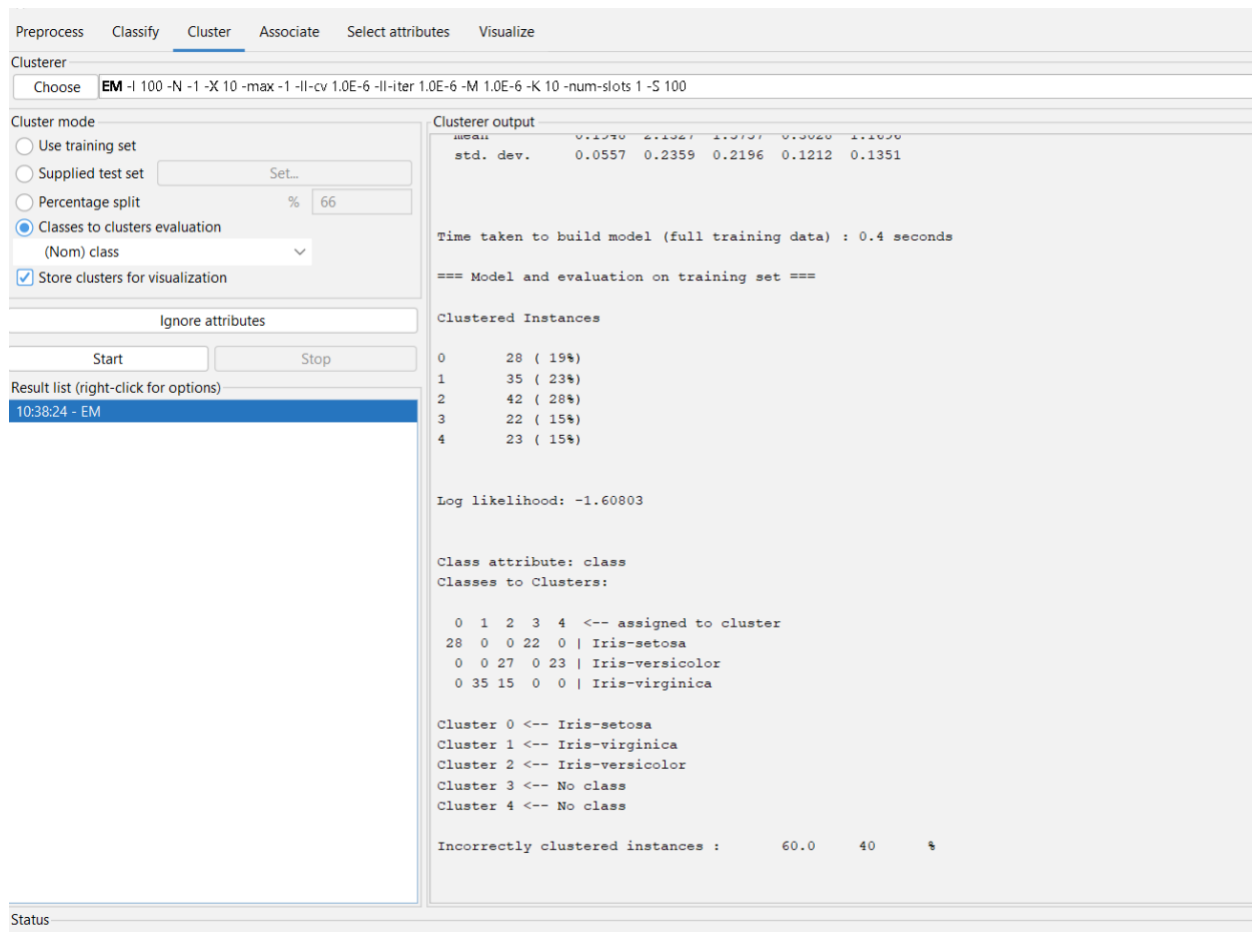
DATA ANALYSIS BY EXPECTATION MAXIMISATION ALGORITHM – OUTPUT INNFO OF IRIS
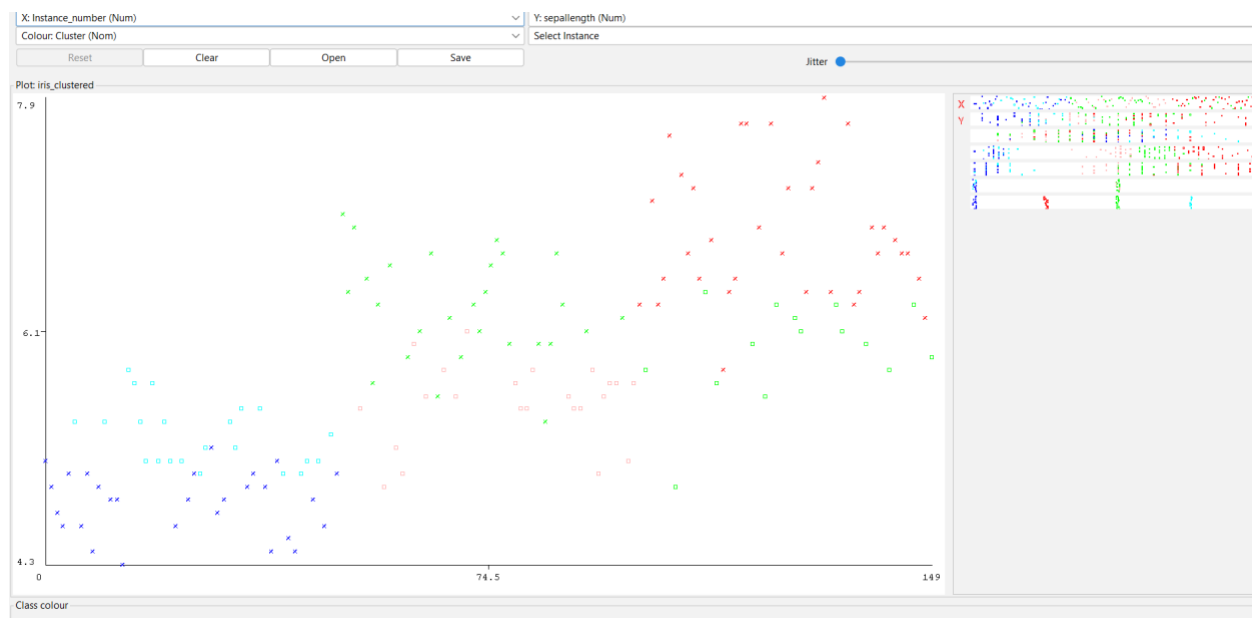
DATA ANALYSIS BY EXPECTATION MAXIMISATION ALGORITHM—PLOT DIAGRAM OF IRIS INFO

10.DATA ANALYSIS BY HIERARCHICAL CLUSTERING. IN WEKA


```
reprocess    Classify    Cluster    Associate    Select attributes    Visualize

usterer
  Choose   HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"

uster mode                                    Clusterer output
  Use training set                              Ignored:
  Supplied test set          Set...                          play
  Percentage split        %  66                Test mode:    Classes to clusters evaluation on training data
  Classes to clusters evaluation
    (Nom) play                          v       === Clustering model (full training set) ===
  Store clusters for visualization
                                                Cluster 0
              Ignore attributes                 (((((1.0:0.69805,1.0:0.69805):0.12095,1.0:0.81901):0.18604,(1.0:0.3674,1.0:0.3674):0.63765):0,(1.0:0.52484,(1.0:0.33333,1.0:0.33333):0.19151):0.480
      Start                    Stop
                                                Cluster 1
esult list (right-click for options)            (0.0:0.68769,0.0:0.68769)
0:49:18 - HierarchicalClusterer

                                                Time taken to build model (full training data) : 0 seconds

                                                === Model and evaluation on training set ===

                                                Clustered Instances

                                                0      12 ( 86%)
                                                1       2 ( 14%)

                                                Class attribute: play
                                                Classes to Clusters:

                                                 0 1  <-- assigned to cluster
                                                 8 1 | yes
                                                 4 1 | no

                                                Cluster 0 <-- yes
                                                Cluster 1 <-- no

                                                Incorrectly clustered instances :      5.0      35.7143 %
```
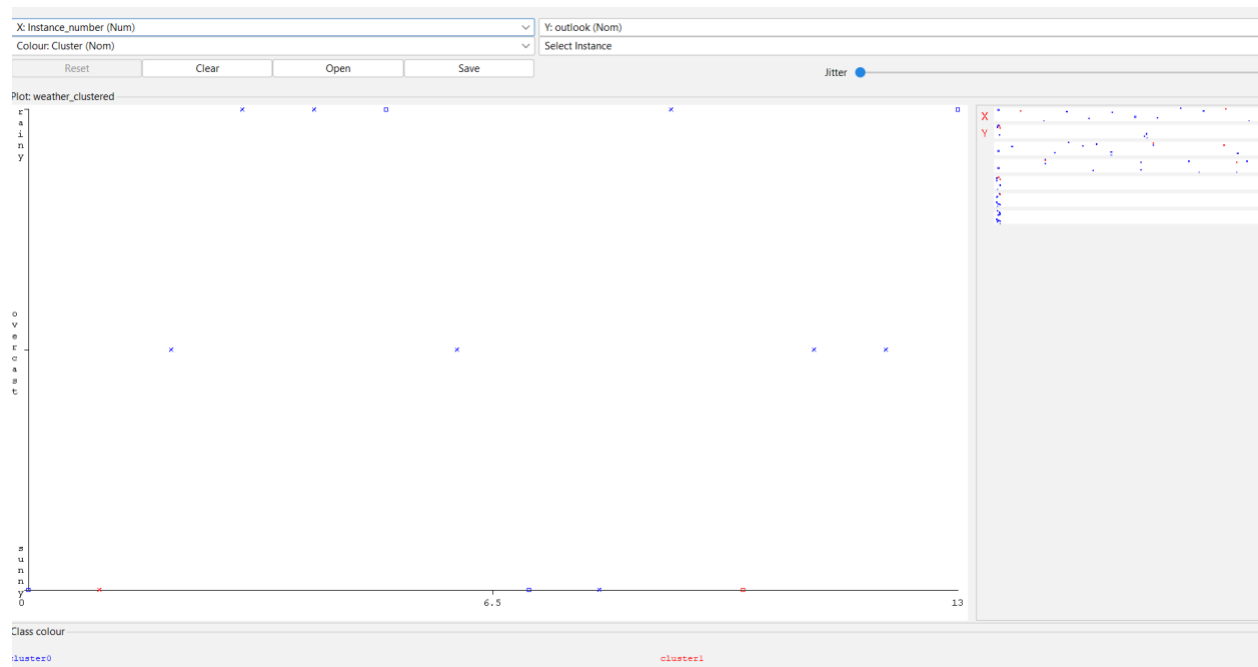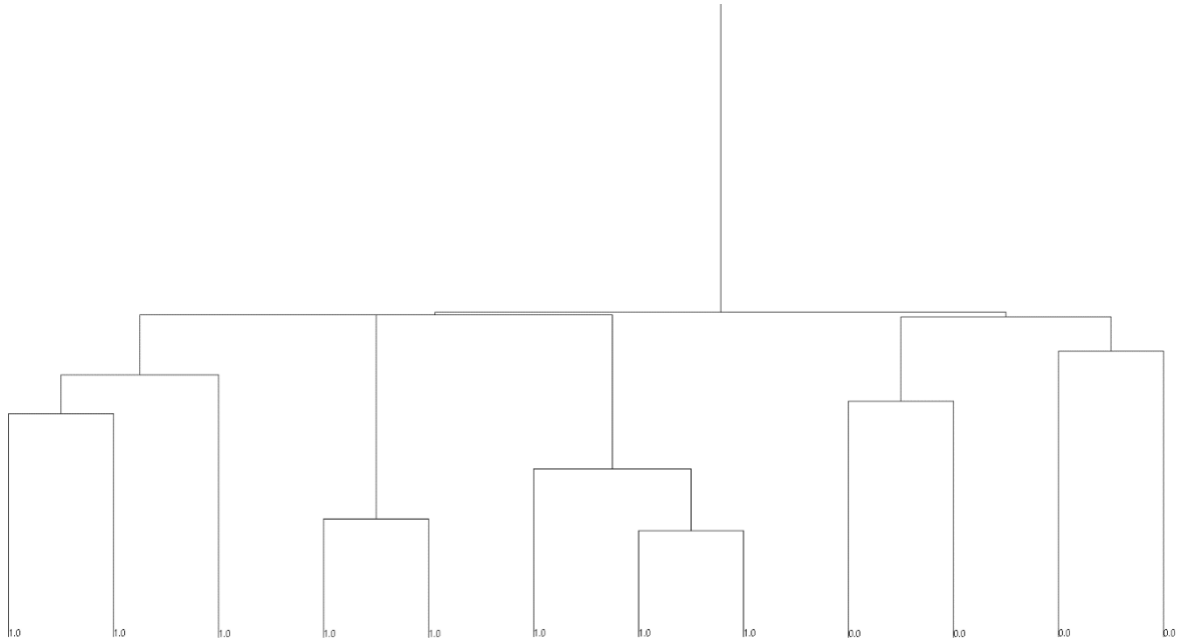
OUTPUT INFORMATION


```
X: Instance_number (Num)                    v   Y: outlook (Nom)
Colour: Cluster (Nom)                       v   Select Instance

    Reset        Clear        Open        Save                    Jitter

Plot: weather_clustered
```

VISUALISING THE PLOT GRAPH OF CLUSTER ASSIGNMENTS

HIERARCHICAL TREE FOR DATA ANALYSIS.

11.KNOWLEDGE MINING FOR ASSOCIATION RULES USING WEKA


```
                    temperature
                    humidity
                    windy
                    play
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.15 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 12

Size of set of large itemsets L(2): 47

Size of set of large itemsets L(3): 39

Size of set of large itemsets L(4): 6

Best rules found:

 1. outlook=overcast 4 ==> play=yes 4      <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 2. temperature=cool 4 ==> humidity=normal 4     <conf:(1)> lift:(2) lev:(0.14) [2] conv:(2)
 3. humidity=normal windy=FALSE 4 ==> play=yes 4     <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 4. outlook=sunny play=no 3 ==> humidity=high 3      <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 5. outlook=sunny humidity=high 3 ==> play=no 3      <conf:(1)> lift:(2.8) lev:(0.14) [1] conv:(1.93)
 6. outlook=rainy play=yes 3 ==> windy=FALSE 3      <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 7. outlook=rainy windy=FALSE 3 ==> play=yes 3      <conf:(1)> lift:(1.56) lev:(0.08) [1] conv:(1.07)
 8. temperature=cool play=yes 3 ==> humidity=normal 3      <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 9. outlook=sunny temperature=hot 2 ==> humidity=high 2      <conf:(1)> lift:(2) lev:(0.07) [1] conv:(1)
10. temperature=hot play=no 2 ==> outlook=sunny 2      <conf:(1)> lift:(2.8) lev:(0.09) [1] conv:(1.29)
```

OUTPUT INFORMATION USING WEATHER NOMINAL DATA
15. EVALUATING THE   ACCURACY OF THE CLASSIFIERS

| Preprocess | Classify | Cluster | Associate | Select attributes | Visualize |

Classifier

Choose **J48** -C 0.25 -M 2

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

(Nom) class

Start   Stop

Result list (right-click for options)
11:26:10 - trees.J48

Classifier output

```
|   |   | petalwidth > 1.J. Iris versicolor (3.0/1.0)
|   petalwidth > 1.7: Iris-virginica (46.0/1.0)

Number of Leaves  :     5

Size of the tree :      9

Time taken to build model: 0.01 seconds

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

Correctly Classified Instances        144               96      %
Incorrectly Classified Instances        6                4      %
Kappa statistic                          0.94
Mean absolute error                      0.035
Root mean squared error                  0.1586
Relative absolute error                  7.8705 %
Root relative squared error             33.6353 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===

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

=== Confusion Matrix ===

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

OUTPUT INFORMAATION OF IRIS IN ACCURACY FINDING.
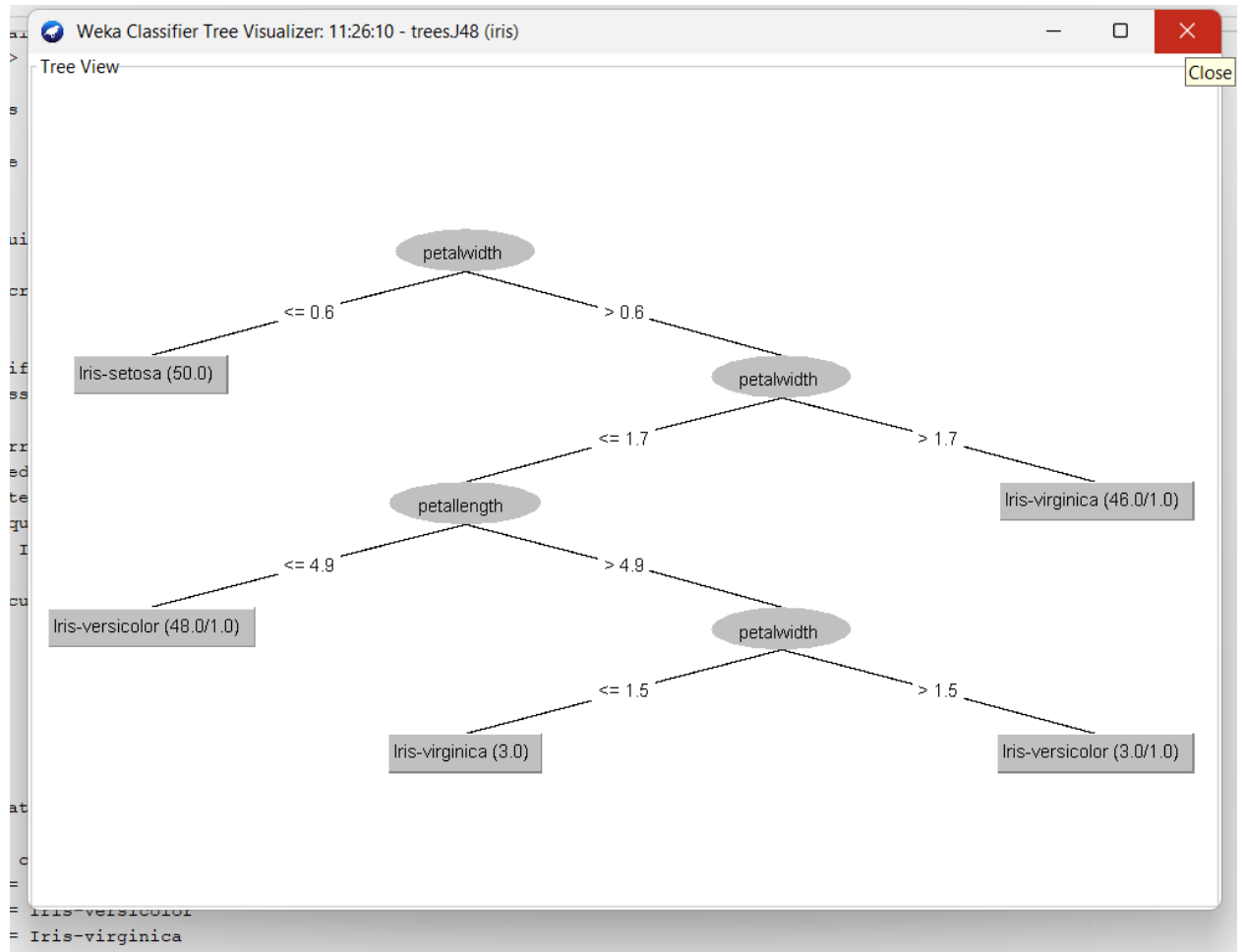
TREE DIAGRAM OF ACCURACY WITH IRIS INFO IN WEKA

ERROR ACCURACY RATE OF IRIS IN WEKA.

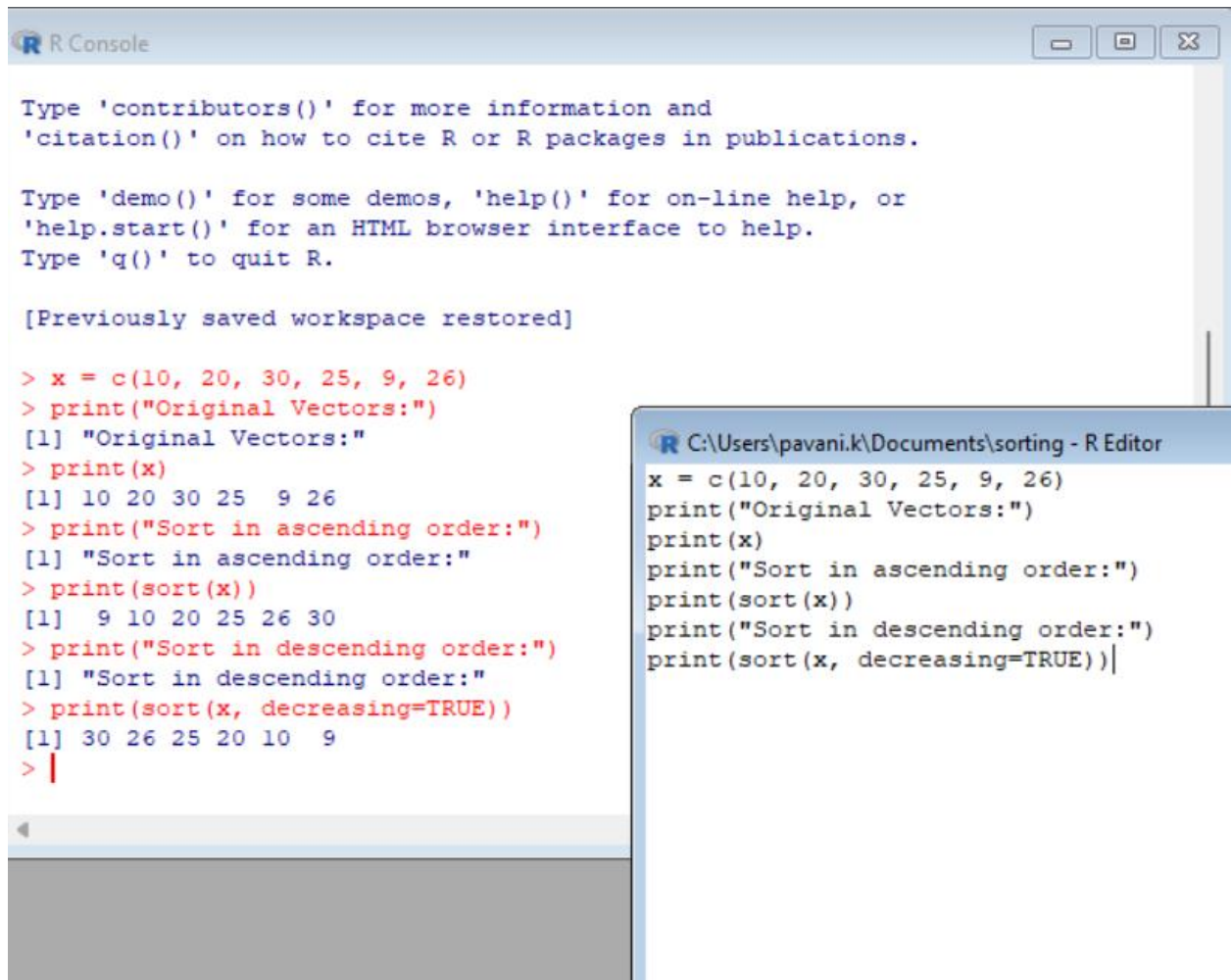1. **CREATE THE ARFF FILE FOR THE DIABETES DATABASE AND PERFORM THE RULE BASED CLASSIFICATION.**

Output info of diabetes by rule based classification.
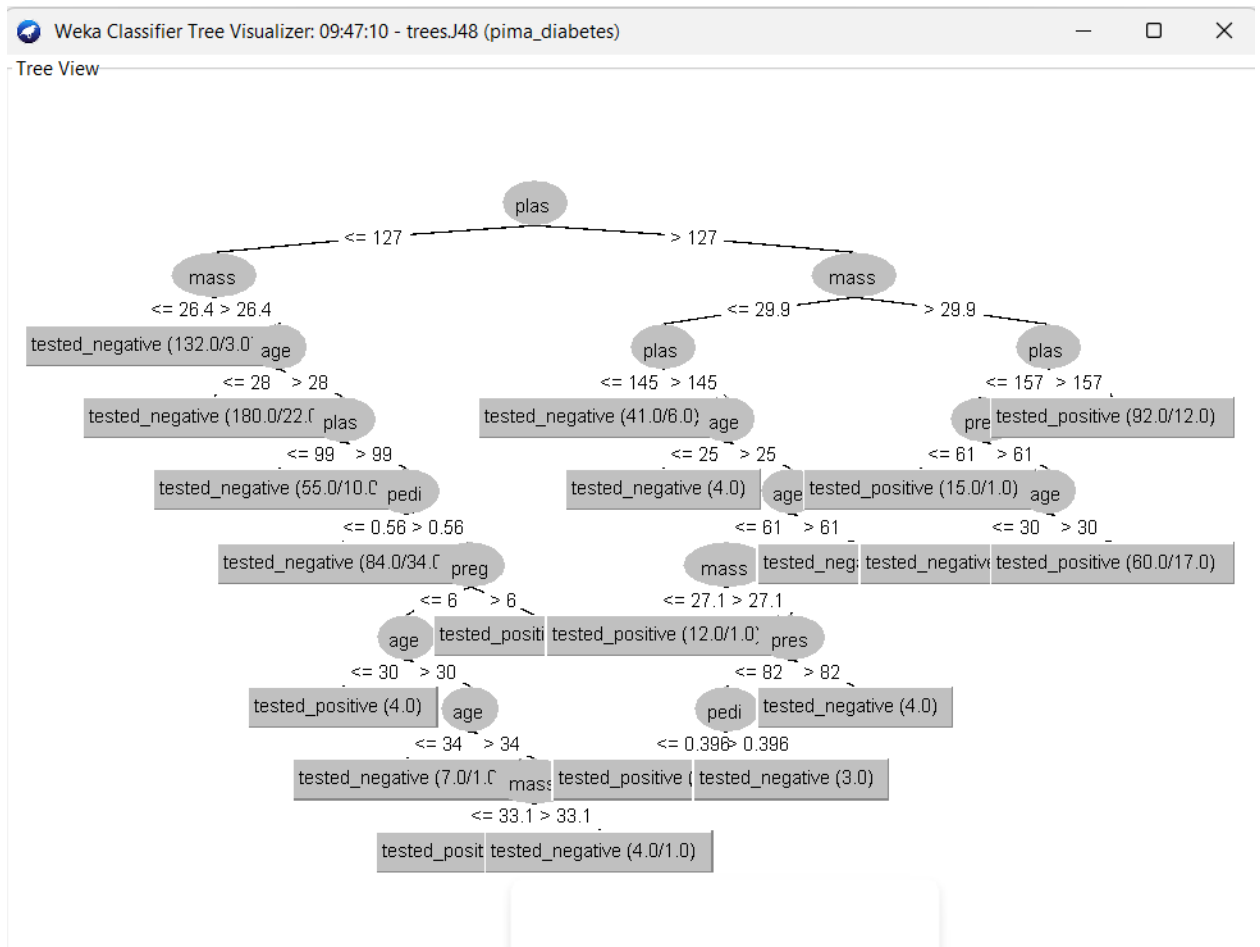
Tree view of diabetes on rule based classification.

*. R PROGRAM FOR SORTING VECTOR

```
R 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.

[Previously saved workspace restored]

> x = c(10, 20, 30, 25, 9, 26)
> print("Original Vectors:")
[1] "Original Vectors:"
> print(x)
[1] 10 20 30 25  9 26
> print("Sort in ascending order:")
[1] "Sort in ascending order:"
> print(sort(x))
[1]  9 10 20 25 26 30
> print("Sort in descending order:")
[1] "Sort in descending order:"
> print(sort(x, decreasing=TRUE))
[1] 30 26 25 20 10  9
> |
```
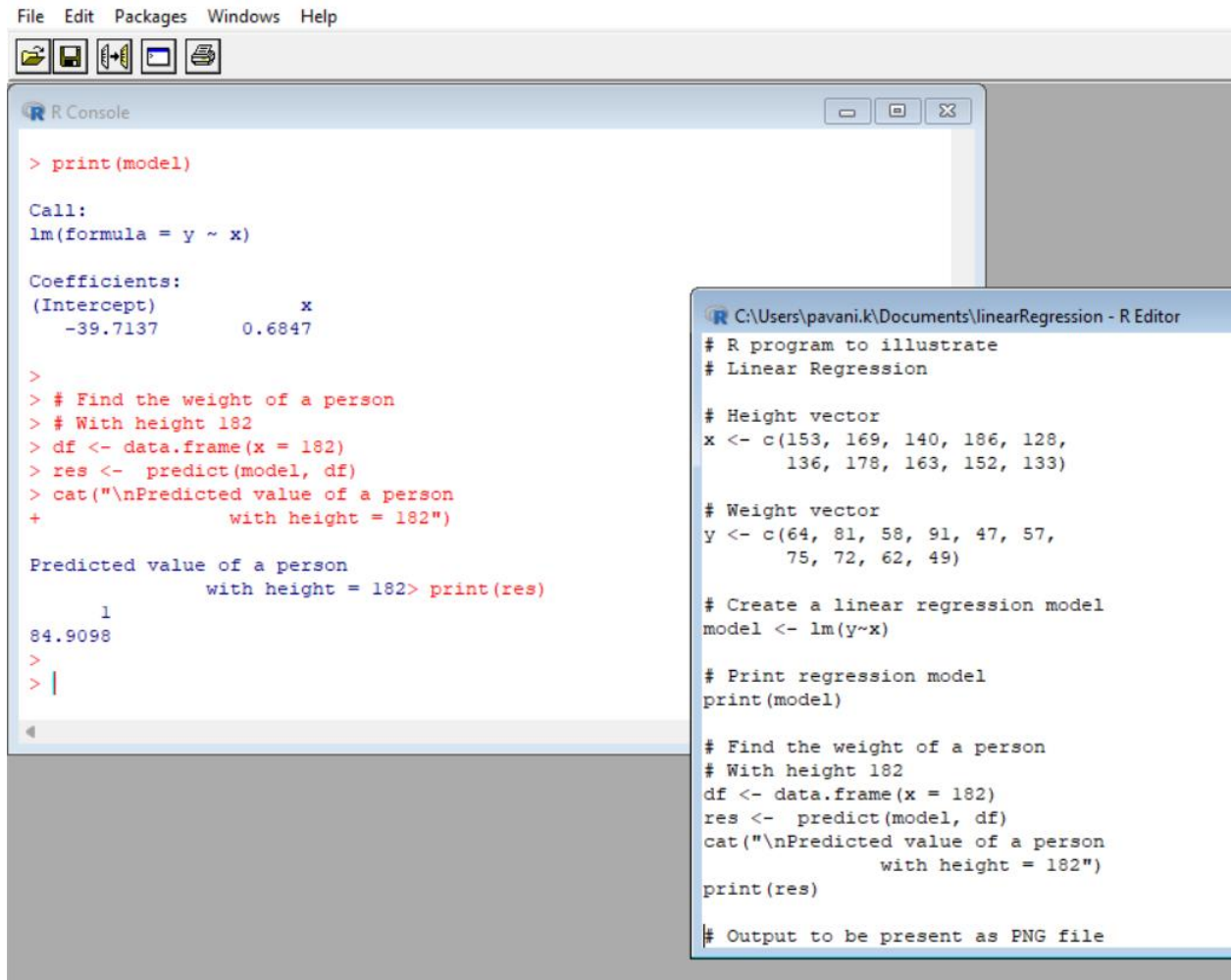
```
R C:\Users\pavani.k\Documents\sorting - R Editor
x = c(10, 20, 30, 25, 9, 26)
print("Original Vectors:")
print(x)
print("Sort in ascending order:")
print(sort(x))
print("Sort in descending order:")
print(sort(x, decreasing=TRUE))|
```

2.R PROGRAM FOR LINEAR REGRESSION.

File   Edit   Packages   Windows   Help

R R Console

```
> print(model)

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)              x
   -39.7137         0.6847

>
> # Find the weight of a person
> # With height 182
> df <- data.frame(x = 182)
> res <-  predict(model, df)
> cat("\nPredicted value of a person
+               with height = 182")

Predicted value of a person
              with height = 182> print(res)
        1
84.9098
>
> |
```

R C:\Users\pavani.k\Documents\linearRegression - R Editor

```
# R program to illustrate
# Linear Regression

# Height vector
x <- c(153, 169, 140, 186, 128,
       136, 178, 163, 152, 133)

# Weight vector
y <- c(64, 81, 58, 91, 47, 57,
       75, 72, 62, 49)

# Create a linear regression model
model <- lm(y~x)

# Print regression model
print(model)

# Find the weight of a person
# With height 182
df <- data.frame(x = 182)
res <-  predict(model, df)
cat("\nPredicted value of a person
              with height = 182")
print(res)

# Output to be present as PNG file
```
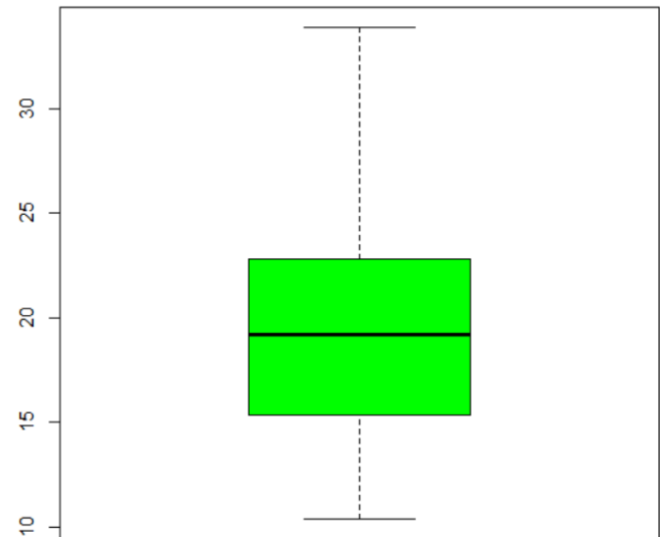
3.PLOTTING GRAPH

```
R C:\Users\pavani.k\Documents\plotting grapgh - R Editor
boxplot(mtcars$mpg, col="green")
```


4.

A) CENTRAL TENDENCY—MEAN:

```
R R Console
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 publicati

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

[Previously saved workspace restored]

> # Defining vector
> x <- c(1, 5, 8, 10)
>
> # Print Harmonic Mean
> print(1 / mean(1 / x))
```

```
R C:\Users\pavani.k\Documents\central tendency - R Editor
###mean::
# Defining vector
x <- c(1, 5, 8, 10)

# Print Harmonic Mean
print(1 / mean(1 / x))

#####median::
# Defining vector
x <- c(3, 7, 5, 13, 20, 23, 39,
       23, 40, 23, 14, 12, 56, 23)

# Print Median
median(x)
```

B) CENTRAL TENDENCY –MEDIAN:

```
Previously saved workspace restored]

# Defining vector
x <- c(1, 5, 8, 10)

# Print Harmonic Mean
print(1 / mean(1 / x))
] 2.807018
# Defining vector
x <- c(3, 7, 5, 13, 20, 23, 39,
        23, 40, 23, 14, 12, 56, 23)

# Print Median
median(x)
] 21.5
```

```
R C:\Users\pavani.k\Documents\central tendency - R Editor
###mean::
# Defining vector
x <- c(1, 5, 8, 10)

# Print Harmonic Mean
print(1 / mean(1 / x))

#####median::
# Defining vector
x <- c(3, 7, 5, 13, 20, 23, 39,
        23, 40, 23, 14, 12, 56, 23)

# Print Median
median(x)

###mode::
# Defining vector
x <- c(3, 7, 5, 13, 20, 23, 39,
        23, 40, 23, 14, 12, 56,
        23, 29, 56, 37, 45, 1, 25, 8)

# Generate frequency table
y <- table(x)

# Print frequency table
print(y)
```

C) CENTRAL   TENDENCY—MODE:

```
> # Generate frequency table
> y <- table(x)
>
> # Print frequency table
> print(y)
x
 1  3  5  7  8 12 13 14 20 23 25 29 37 39 40 45 56
 1  1  1  1  1  1  1  1  1  4  1  1  1  1  1  1  2
> |
```

```
x <- c(3, 7, 5, 13, 20, 23, 39,
        23, 40, 23, 14, 12, 56,

# Print Median
median(x)

###mode::
# Defining vector
x <- c(3, 7, 5, 13, 20, 23, 39,
        23, 40, 23, 14, 12, 56,
        23, 29, 56, 37, 45, 1, 2

# Generate frequency table
y <- table(x)

# Print frequency table
print(y)
```

5.NORMALISATION AND ANALYSIS:

```
#apply Min-Max normalization to first four columns in iris datas
iris_norm <- as.data.frame(lapply(iris[1:4], min_max_norm))

#view first six rows of normalized iris dataset
head(iris_norm)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
   0.22222222   0.6250000   0.06779661   0.04166667
   0.16666667   0.4166667   0.06779661   0.04166667
   0.11111111   0.5000000   0.05084746   0.04166667
   0.08333333   0.4583333   0.08474576   0.04166667
   0.19444444   0.6666667   0.06779661   0.04166667
   0.30555556   0.7916667   0.11864407   0.12500000
```

R C:\Users\pavani.k\Documents\normalisation - R Editor

```
#define Min-Max normalization function
min_max_norm <- function(x) {
    (x - min(x)) / (max(x) - min(x))
  }

#apply Min-Max normalization to first four columns in iris datase
iris_norm <- as.data.frame(lapply(iris[1:4], min_max_norm))

#view first six rows of normalized iris dataset
head(iris_norm)
```

6. REGRESSION:

```
> # Generate vector with pass and fail values of 40 students
> result <- c(0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
+ 1, 0, 0, 0, 1, 1, 0, 0, 1, 0,
+ 0, 0, 1, 0, 0, 1, 1, 0, 1, 1,
+ 1, 1, 1, 0, 1, 1, 1, 1, 0, 1)
>
> # Data Frame
> df <- as.data.frame(cbind(IQ, result))
>
> # Print data frame
> print(df)
        IQ result
1  26.88303      0
2  26.99592      0
3  27.27634      0
4  27.39254      1
5  27.40339      0
6  27.49747      0
7  27.66223      0
8  27.93250      0
9  28.18333      0
10 28.19356      1
11 28.23130      1
```

R C:\Users\pavani.k\Documents\regression - R Editor

```
# Generate random IQ values with mean = 30 and sd =2
IQ <- rnorm(40, 30, 2)

# Sorting IQ level in ascending order
IQ <- sort(IQ)

# Generate vector with pass and fail values of 40 students
result <- c(0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
1, 0, 0, 0, 1, 1, 0, 0, 1, 0,
0, 0, 1, 0, 0, 1, 1, 0, 1, 1,
1, 1, 1, 0, 1, 1, 1, 1, 0, 1)

# Data Frame
df <- as.data.frame(cbind(IQ, result))

# Print data frame
print(df)

# output to be present as PNG file
png(file="LogisticRegressionGFG.png")

# Plotting IQ on x-axis and result on y-axis
plot(IQ, result, xlab = "IQ Level",
ylab = "Probability of Passing")

# Create a logistic model
```

12.FP GROWTH

atrix   outlook   temperature   humidity   windy   play

lity

erature

ok

ze: [15...

ize: [1]
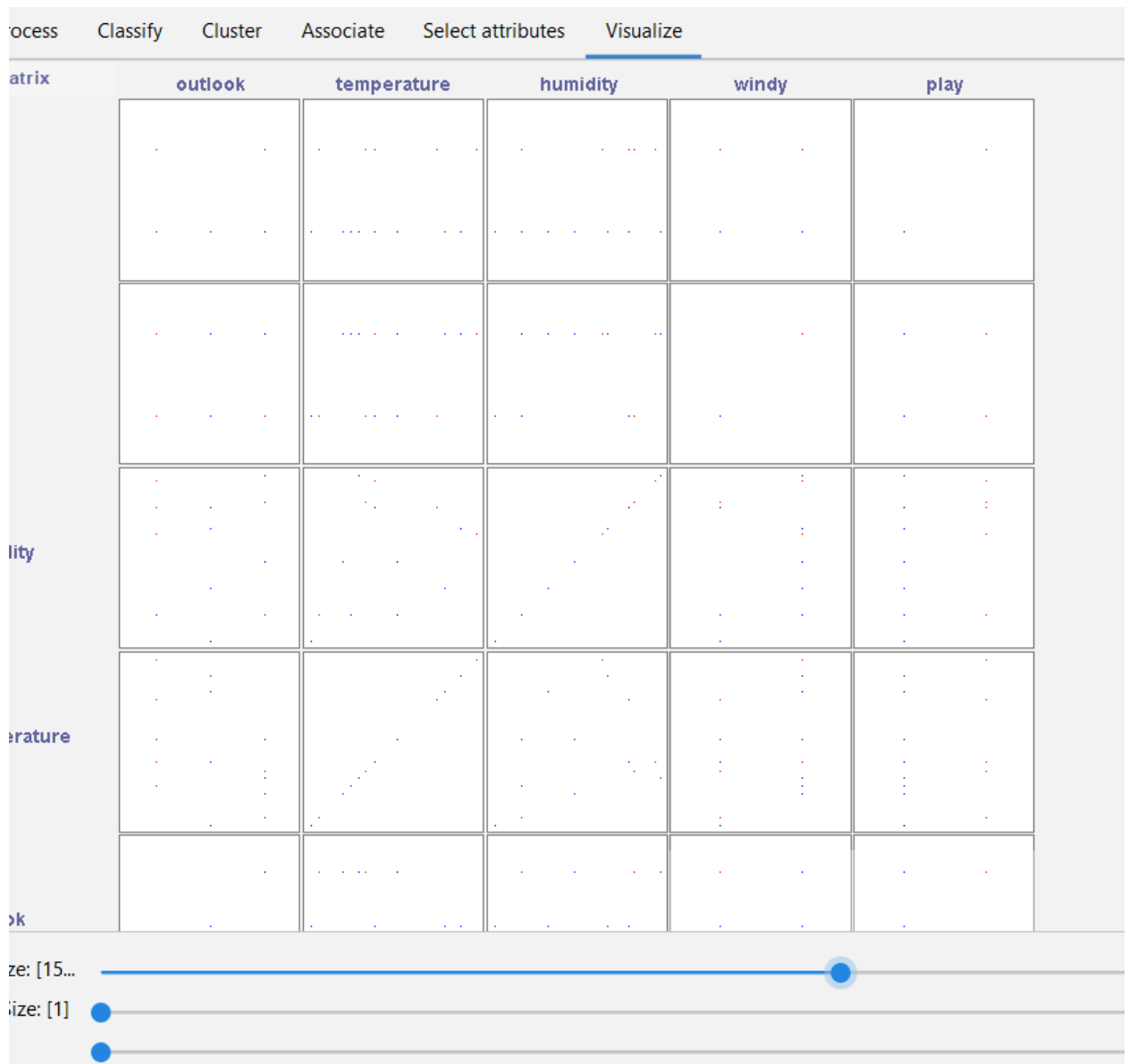
FP GROWTH VISUALISATION

Associator output

```
                temperature
                humidity
                windy
                play
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.15 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 12

Size of set of large itemsets L(2): 47

Size of set of large itemsets L(3): 39

Size of set of large itemsets L(4): 6

Best rules found:

 1. outlook=overcast 4 ==> play=yes 4      <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 2. temperature=cool 4 ==> humidity=normal 4     <conf:(1)> lift:(2) lev:(0.14) [2] conv:(2)
 3. humidity=normal windy=FALSE 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 4. outlook=sunny play=no 3 ==> humidity=high 3      <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 5. outlook=sunny humidity=high 3 ==> play=no 3      <conf:(1)> lift:(2.8) lev:(0.14) [1] conv:(1.93)
 6. outlook=rainy play=yes 3 ==> windy=FALSE 3      <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 7. outlook=rainy windy=FALSE 3 ==> play=yes 3      <conf:(1)> lift:(1.56) lev:(0.08) [1] conv:(1.07)
 8. temperature=cool play=yes 3 ==> humidity=normal 3     <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 9. outlook=sunny temperature=hot 2 ==> humidity=high 2     <conf:(1)> lift:(2) lev:(0.07) [1] conv:(1)
10. temperature=hot play=no 2 ==> outlook=sunny 2      <conf:(1)> lift:(2.8) lev:(0.09) [1] conv:(1.29)
```

13.DECISION TREE

TREE VISUALISATION

OUTPUT INFO.

16. CREATE THE ARFF FILE FOR THE DIABETES DATABASE AND PERFORM SVM BASED CLASSIFICATION

ssifier

Choose   **Logistic** -R 1.0E-8 -M -1 -num-decimal-places 4

t options

) Use training set

) Supplied test set   Set...

) Cross-validation   Folds   10

) Percentage split   %   66

More options...

om) class

Start   Stop

ult list (right-click for options)

Classifier output

```
66 2:tested_positive 2:tested_positive          0.862
67 2:tested_positive 2:tested_positive          0.863
68 2:tested_positive 2:tested_positive          0.786
69 2:tested_positive 2:tested_positive          0.666
70 2:tested_positive 2:tested_positive          0.514
71 2:tested_positive 1:tested_negative     +    0.785
72 2:tested_positive 2:tested_positive          0.815
73 2:tested_positive 1:tested_negative     +    0.652
74 2:tested_positive 1:tested_negative     +    0.52
75 2:tested_positive 1:tested_negative     +    0.634
76 2:tested_positive 1:tested_negative     +    0.851


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

Correctly Classified Instances          597              77.7344 %
Incorrectly Classified Instances        171              22.2656 %
Kappa statistic                           0.4855
Mean absolute error                       0.319
Root mean squared error                   0.3996
Relative absolute error                  70.1878 %
Root relative squared error              83.8438 %
Total Number of Instances               768

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall  F-Measure  MCC    ROC Area  PRC Area  Class
                 0.884    0.422    0.796      0.884   0.838      0.492  0.825     0.886     tested_negative
                 0.578    0.116    0.728      0.578   0.644      0.492  0.825     0.697     tested_positive
Weighted Avg.    0.777    0.315    0.772      0.777   0.770      0.492  0.825     0.820

=== Confusion Matrix ===

   a    b    <-- classified as
 442   58 |   a = tested_negative
 113  155 |   b = tested_positive
```

Output information of data diabetes with logistics.

14. PREDICTION OF CATEGORICAL DATA USING SMO ALGORITHM USING WEKA.

The output information of diabetes data using smo..

17. PREDICTION OF CATEGORICAL DATA USING BAYESIAN ALGORITHM USING WEKA

OUTPUT INFORMATION WEATHER NUMERIC BASED ON NAVIES BAYES RULE..

18. **DATA ANALYSIS BY DENSITY BASED CLUSTERING ALGORITHM USING WEKA.**

Weka Explorer — □

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Clusterer

Choose | EM -I 100 -N -1 -X 10 -max -1 -ll-cv 1.0E-6 -ll-iter 1.0E-6 -M 1.0E-6 -K 10 -num-slots 1 -S 100

Cluster mode

○ Use training set
○ Supplied test set — Set...
○ Percentage split — % 66
● Classes to clusters evaluation
   (Nom)   Train on a percentage of the data and cluster the remainder
☑ Store clusters for visualization

Ignore attributes

Start | Stop

Result list (right-click for options)

08:53:20 - EM

Clusterer output

```
 high        8
 normal      8
 [total]    16
windy
 TRUE        7
 FALSE       9
 [total]    16


Time taken to build model (full training data) : 0.04 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      14 (100%)


Log likelihood: -3.54934


Class attribute: play
Classes to Clusters:

 0  <-- assigned to cluster
 9 | yes
 5 | no


Cluster 0 <-- yes

Incorrectly clustered instances :        5.0        35.7143 %
```
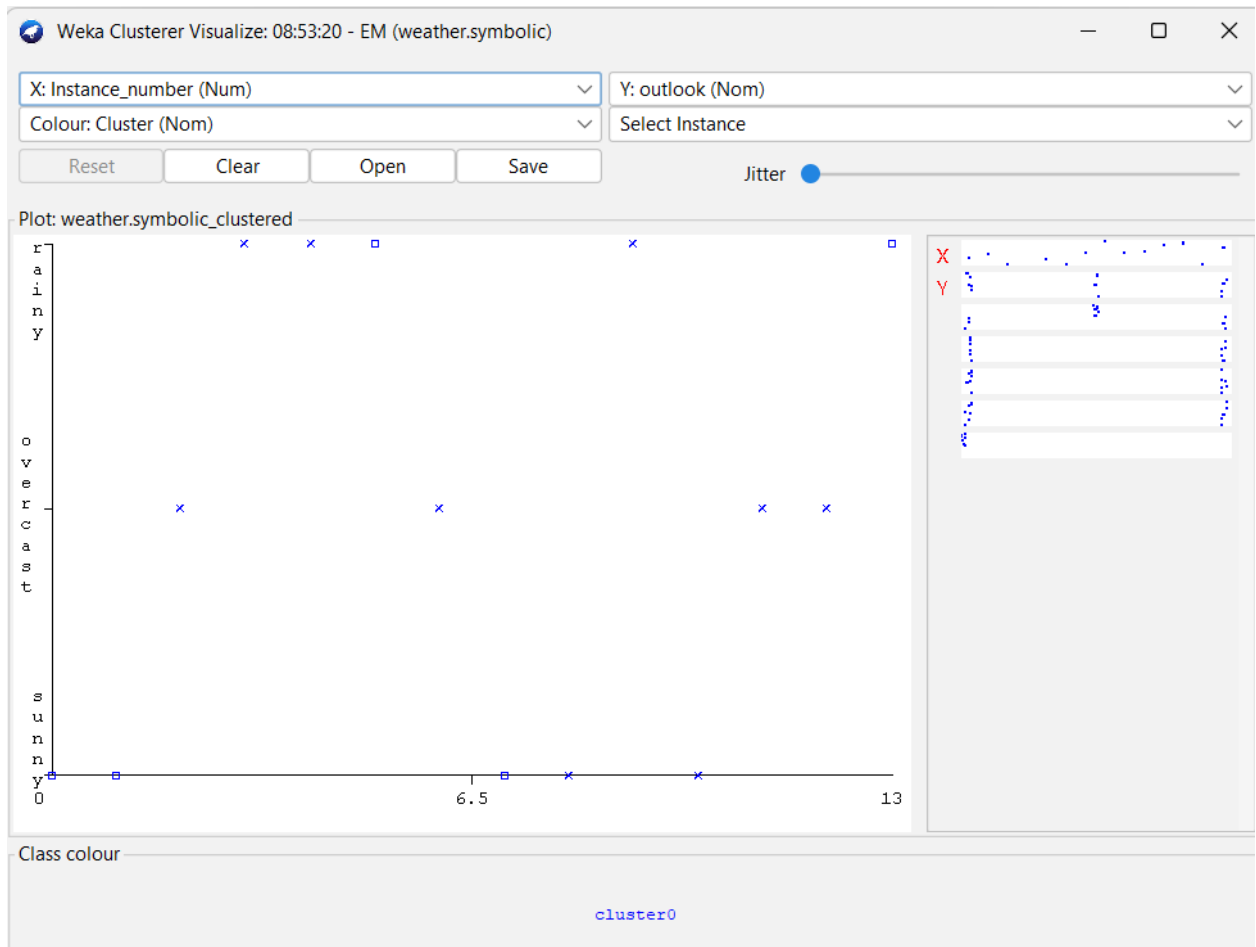
Status

OK — Log

Output information...

19. **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**


```
############code for creating boxplots#####
# Give the chart file a name.
png(file = "boxplot.png")
# Plot the chart.
boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders",
    ylab = "Miles Per Gallon", main = "Mileage Data")
# Save the file.
dev.off()
```

Code for creating boxplots in r programing.

```
   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.

[Previously saved workspace restored]

> # Give the chart file a name.
> png(file = "boxplot.png")
> # Plot the chart.
> boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders",
+    ylab = "Miles Per Gallon", main = "Mileage Data")
> # Save the file.
> dev.off()
null device
```
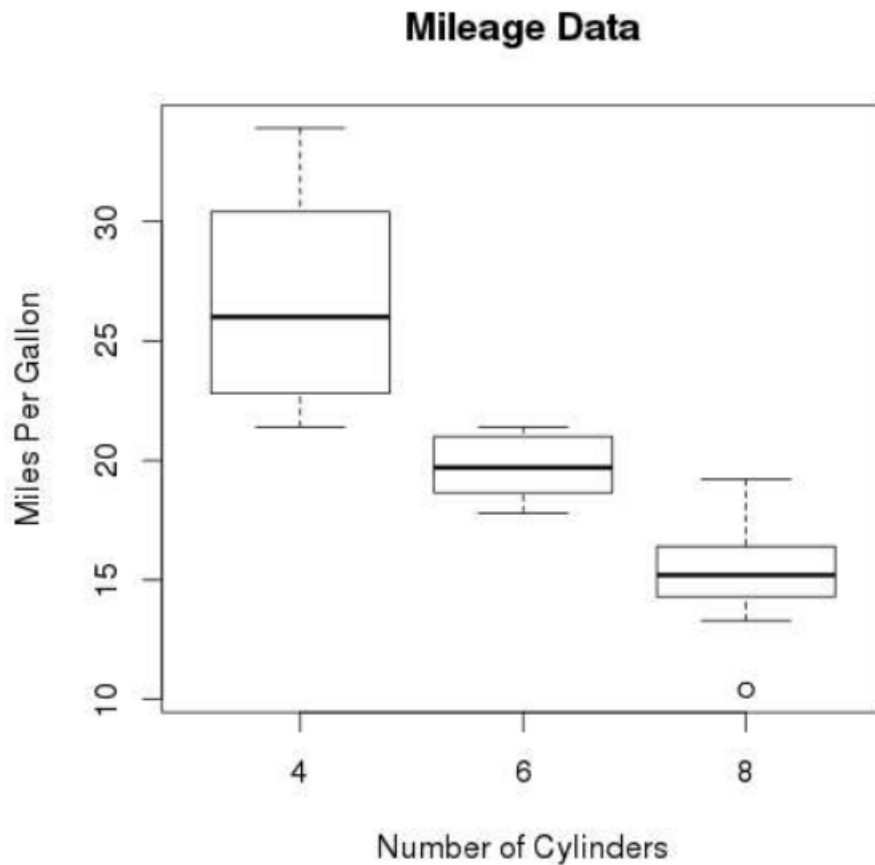
Output result of created boxplot..

Mileage Data

Output graph of boxplot..

22. USING R PROGRAM 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

```
R Untitled - R Editor
#####code for creating an histogram#####
hist(AirPassengers,
     main="Histogram for Air Passengers",
     xlab="Passengers",
     border="blue",
     col="green",
     xlim=c(100,700),
     las=1,
     breaks=5)
```

Code for creating an histogram..
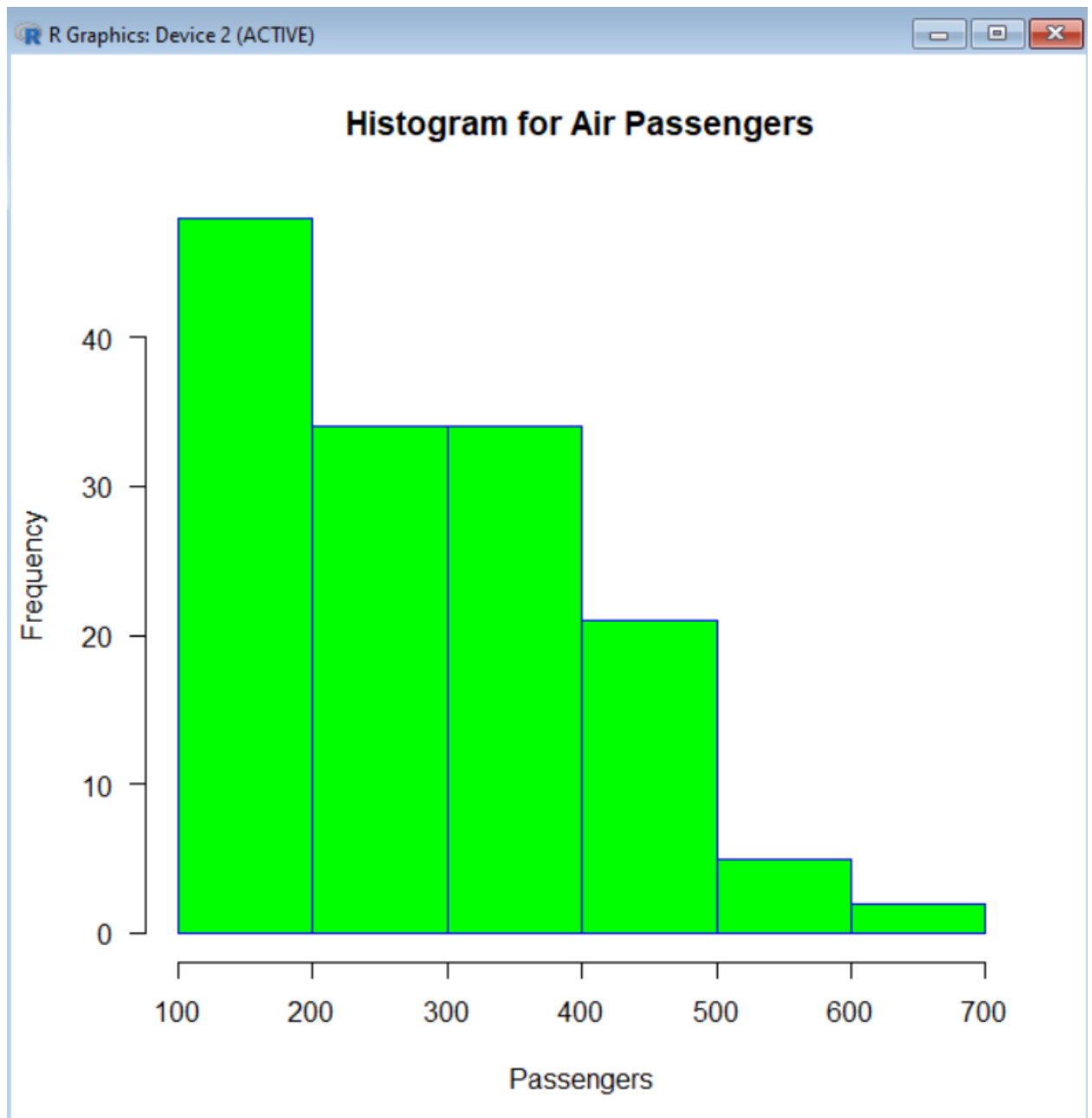
Output of histogram graphical representation..

23. USING R PROGRAM CREATE A 3D PIE CHART FOR THE DATASET "POLITICAL KNOWLEDGE" WITH SUITABLE LABELS AND COLOURS.

```
R Untitled - R Editor                                                    ⊟

###code for piechart creaton ########
values <-  c(906, 264, 289, 339, 938)
countries <-  c("India","Sri Lanka","Nepal","Bhutan", "China")

pie(values, labels = countries)


"Ch

289
ries
"Ch
```

Code for creating the pie charts
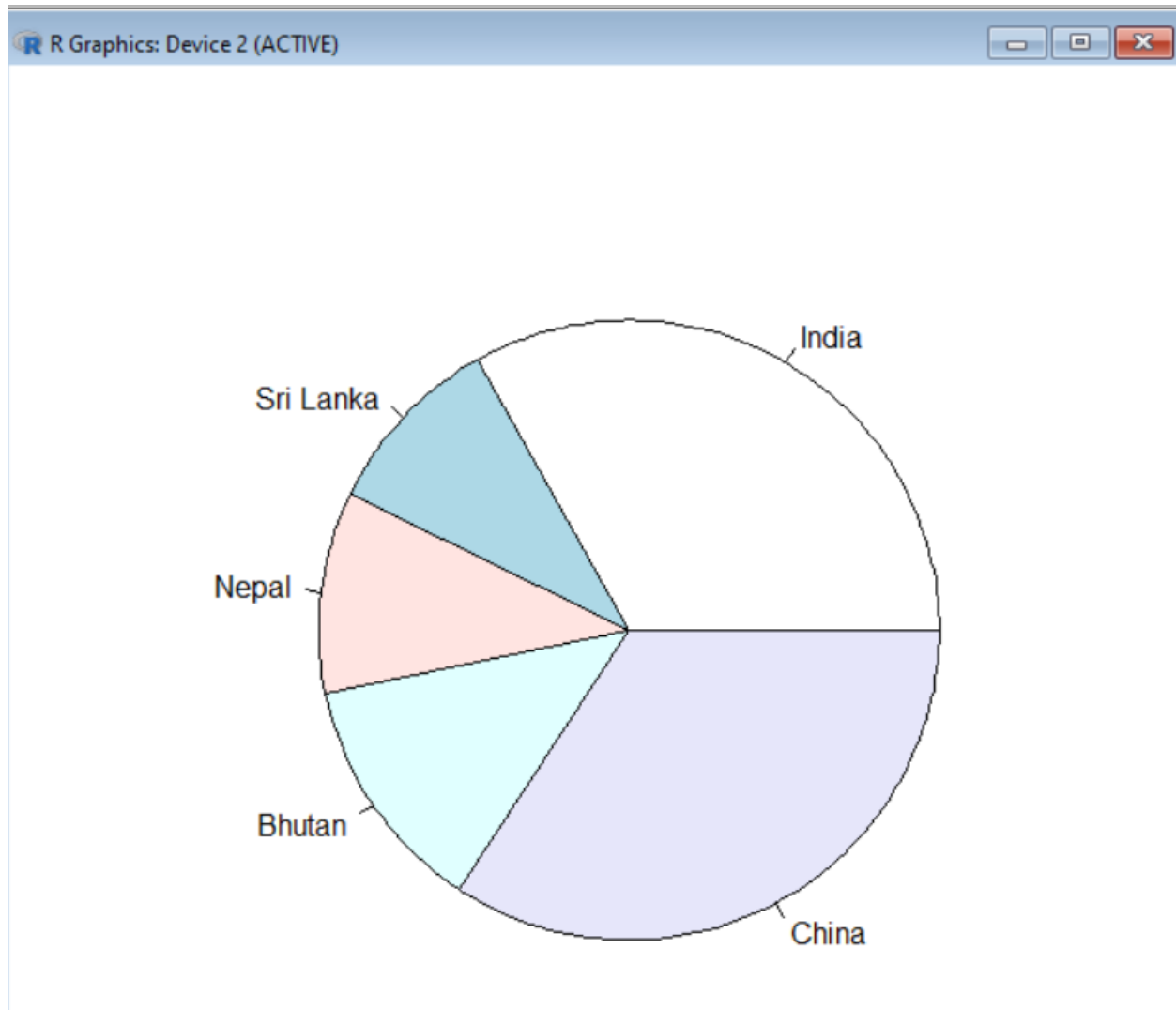
Output of created pie chart.

24. OBTAIN MULTIPLE LINES IN LINE CHART USING A SINGLE PLOT FUNCTION IN R.USE ATTRIBUTES "MPG"AND "QSEC" OF   THE DATASET "MTCARS"

```
R Untitled - R Editor                                              □   ▣   ✕

######multiple lines in a chart##
#Create a fake dataset with 3 columns (ncol=3) composed of randomly generated
#numbers from a uniform distribution with minimum = 1 and maximum = 10
data <- matrix(runif(30,1,10), ncol=3)
data
          [,1]      [,2]      [,3]
#[1,]  5.371653 3.490919 3.953603
#[2,]  9.551883 2.681054 9.506765
#[3,]  3.525686 1.027758 8.059011
#[4,]  9.923080 1.337935 1.112361
#[5,]  7.273972 7.627546 1.174340
#[6,]  8.859109 3.778144 9.384526
#[7,]  9.614542 3.866029 7.301729
#[8,]  9.288085 5.804041 8.347907
#[9,]  1.696849 4.650687 7.220209
#[10,] 5.820941 4.799682 5.243663

#plot the three columns of the dataset as three lines and add a legend in
#the top right corner of the chart
matplot(data, type = "b",pch=1,col = 1:3)
legend("topright", legend = 1:3, col=1:3, pch=1)
```

**code for getting multiple lines in chart.**
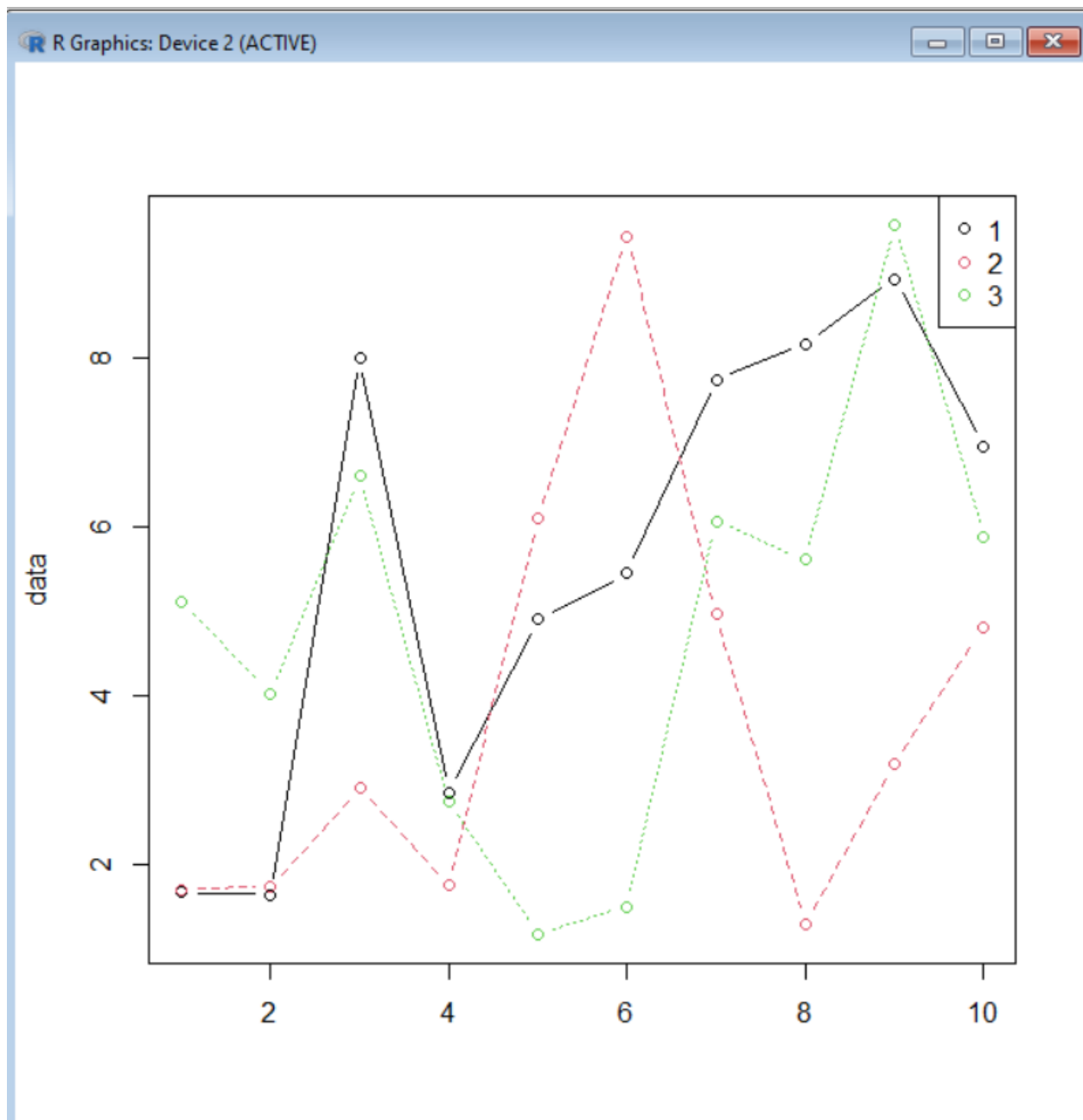
Output of multiple lines in chart.

25. USING R PROGRAM MAKE A HISTOGRAM FOR THE "TOOTHGROWTH" DATASET, START AT 100 ON THE X-AXIS, AND FROM VALUES 200 TO 700, MAKE THE BINS 150 WIDE

```
R Untitled - R Editor                                                    □  □

##code for tooth growth 25 #####
hist (tooth growth, xlim=c (150,600), ylim=c (0,35))
In the above example x limit varies from 150 to 600 and Y - 0 to 35.
// Adding breaks
hist (AirPassengers,
main="Histogram with more Arg",
xlab="Name List",
border="Green",
col="Orange",
xlim=c (100,600),
ylim=c(0,40),
breaks=5)
```

Code for creating a histogram for tooth growth.
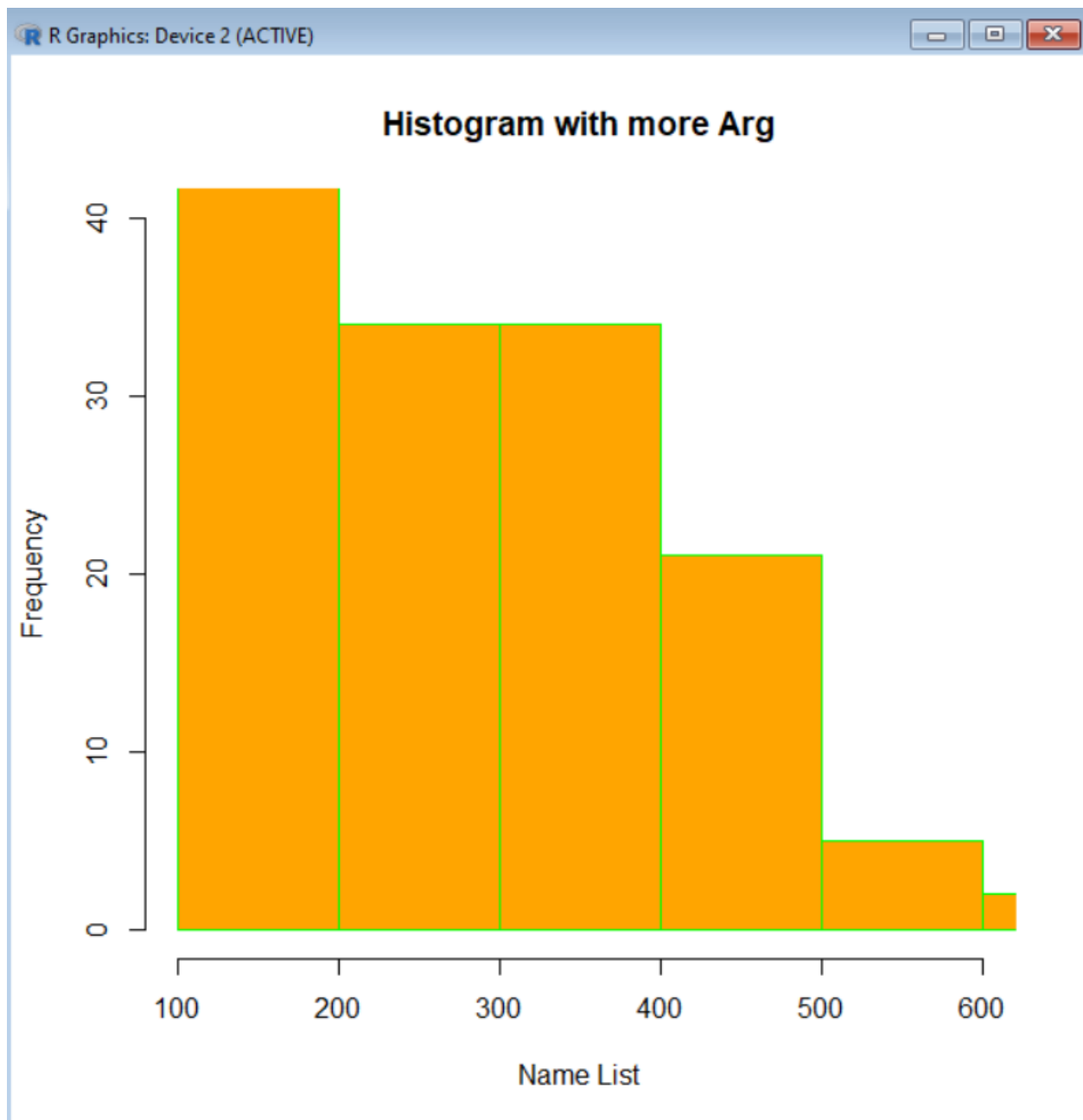
Output of histogram, for tooth growth.