

LAB FILE
Data Warehousing and
Data Mining
IT405



Submitted by:
Mrigank Badola

Submitted to:
Ms. Priyanka Meel

Delhi Technological University
Bawana Rd, Shahbad Daulatpur Village, Rohini, Delhi,

Program 1

AIM:

To introduce about launching the WEKA Tool.

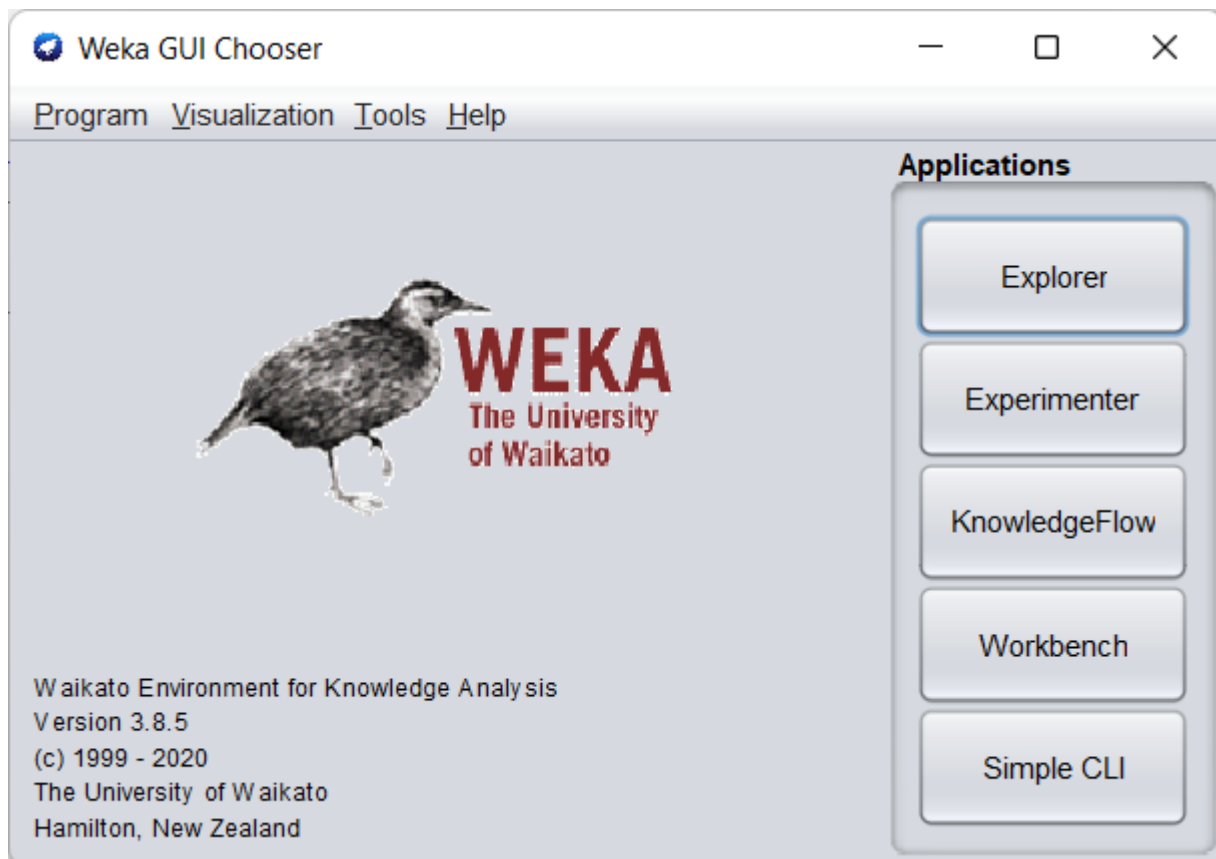
THEORY:

WEKA (Waikato Environment for Knowledge Analysis) is a set of machine learning algorithms that can be applied to a data set directly or called from your own Java code. WEKA contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. A network essentially is collection of routers, some of which are connected with each other by links. It can be interpreted as graph data, with routers and links as vertices and edges.

PROCEDURE:

- Navigate to <https://www.cs.waikato.ac.nz/ml/weka/> and click on 'Download' hyperlink
- If on Windows OS, download a self-extracting executable for 64-bit Windows that includes Azul's 64-bit OpenJDK Java VM 11 (weka-3-8-5-azul-zulu-windows.exe; 124.6 MB)
- The executable file installs WEKA into our system.
- Launching via the Program Menu or shortcuts will automatically use the included JVM to run Weka.

OUTPUT:



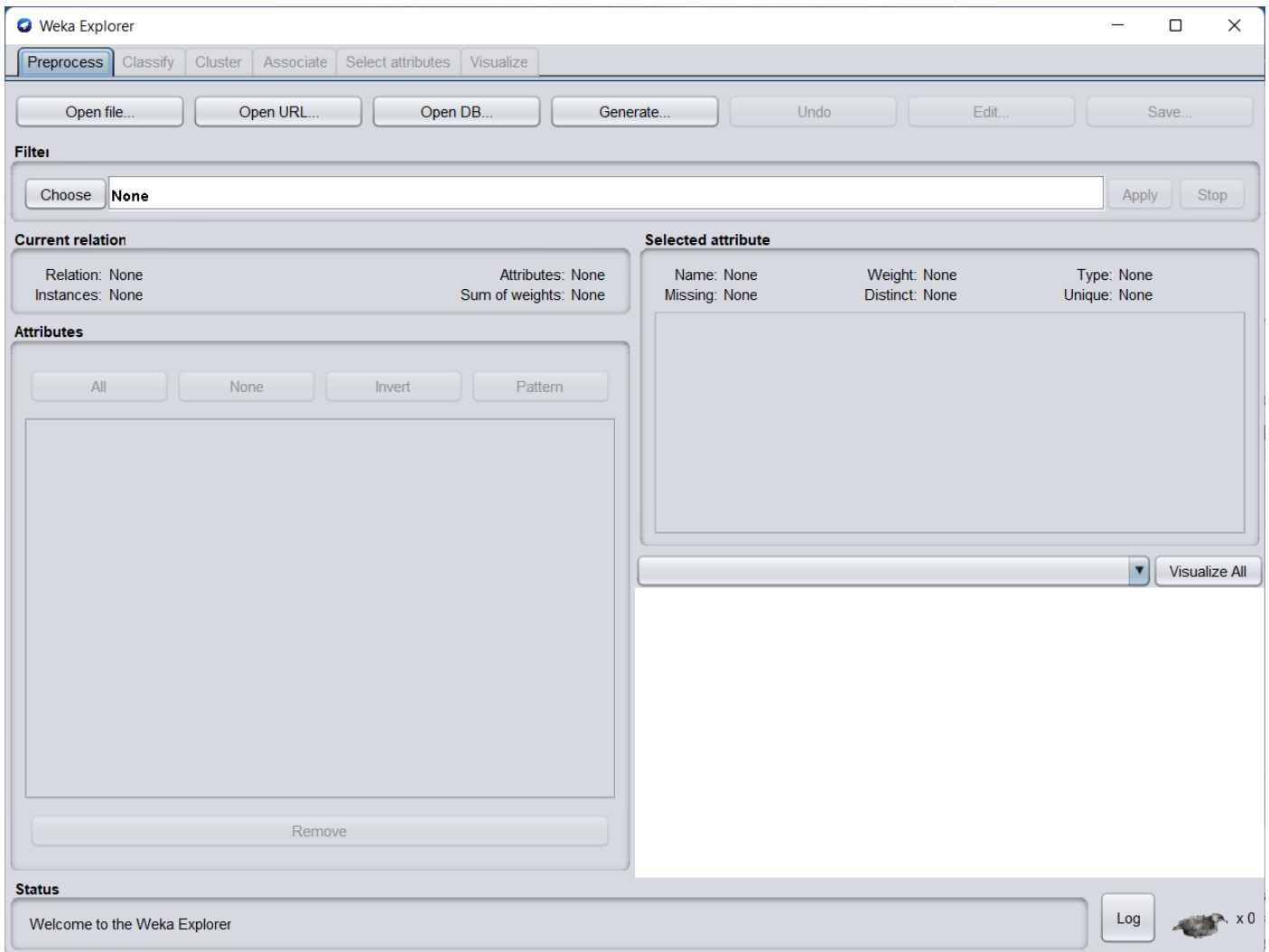
Program 2

AIM:

To introduce to WEKA explorer.

THEORY:

The WEKA Explorer window contains six tabs.



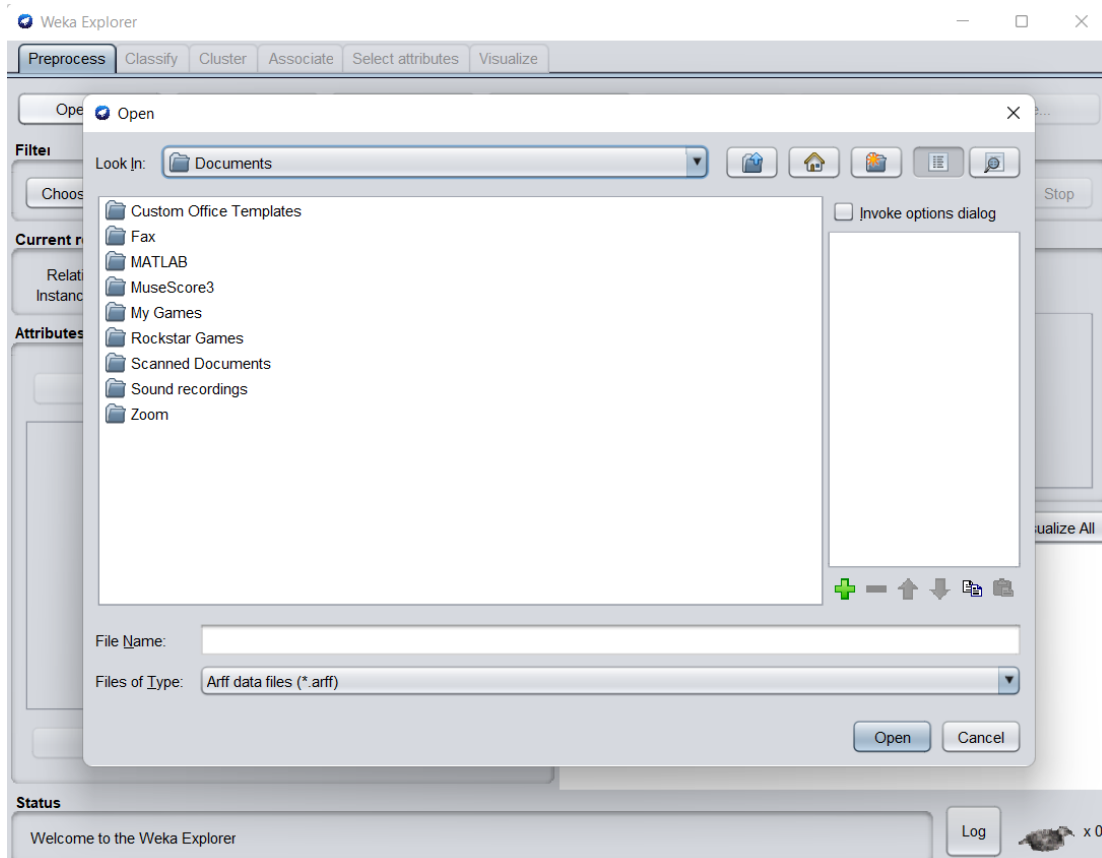
1. Preprocess: This allows us to choose the data file.
2. Classify: This allows us to apply and experiment with different algorithms on preprocessed data files.
3. Cluster: This allows us to apply different clustering tools, which identify clusters within the data file.
4. Association: This allows us to apply association rules, which identify the association within the data.
5. Select attributes: These allow us to see the changes in the inclusion and exclusion of attributes from the experiment.
6. Visualize: This allows us to see the possible visualization produced on the data set in a 2D format, in scatter plot and bar graph output.

PROCEDURE:

- Open the WEKA tool and click on 'Explorer' button.
- For tab 'Preprocess'
 - Data pre-processing is a must. There are three ways to inject the data for pre-processing:
 - Open File - enables the user to select the file from the local machine
 - Open URL - enables the user to select the data file from different locations
 - Open Database - enables users to retrieve a data file from a database source
 - A screen for selecting a file from the local machine to be pre-processed is shown in the OUTPUT section. After loading the data in Explorer, we can refine the data by selecting different options. We can also select or remove the attributes as per our needs and even apply filters on data to refine the result.
- For tab 'Classify'
 - To predict nominal or numeric quantities, we have classifiers in WEKA. Available learning schemes are decision-trees and lists, support vector machines, instance-based classifiers, logistic regression, and Bayes' nets. Once the data has been loaded, all the tabs are enabled. Based on the requirements and by trial and error, we can find out the most suitable algorithm to produce an easily understandable representation of data.
 - Before running any classification algorithm, we need to set test options. Available test options are listed below:
 - Use training set: Evaluation is based on how well it can predict the class of the instances it was trained on.
 - Supplied training set: Evaluation is based on how well it can predict the class of a set of instances loaded from a file.
 - Cross-validation: Evaluation is based on cross-validation by using the number of folds entered in the 'Folds' text field.
 - Split percentage: Evaluation is based on how well it can predict a certain percentage of the data, held out for testing by using the values entered in the '%' field.
 - To classify the data set based on the characteristics of attributes, Weka uses classifiers.
- For tab 'Cluster'
 - The cluster tab enables the user to identify similarities or groups of occurrences within the data set. Clustering can provide data for the user to analyse. The training set, percentage split, supplied test set and classes are used for clustering, for which the user can ignore some attributes from the data set, based on the requirements. Available clustering schemes in Weka are k- Means, EM, Cobweb, X-means, and FarthestFirst.

- For tab 'Associate'
 - The only available scheme for association in Weka is the Apriori algorithm. It identifies statistical dependencies between clusters of attributes, and only works with discrete data. The Apriori algorithm computes all the rules having minimum support and exceeding a given confidence level.
- For tab 'Select attributes'
 - Attribute selection crawls through all possible combinations of attributes in the data to decide which of these will best fit the desired calculation—which subset of attributes works best for prediction. The attribute selection method contains two parts.
 - Search method: Best-first, forward selection, random, exhaustive, genetic algorithm, the ranking algorithm
 - Evaluation method: Correlation-based, wrapper, information gain, chi-squared.
 - All the available attributes are used in the evaluation of the data set by default. But it enables users to exclude some of them if they want to.
- For tab 'Visualize'
 - The user can see the final piece of the puzzle, derived throughout the process. It allows users to visualize a 2D representation of data and is used to determine the difficulty of the learning problem. We can visualize single attributes (1D) and pairs of attributes (2D), and rotate 3D visualizations in Weka. It has the Jitter option to deal with nominal attributes and to detect 'hidden' data points.

OUTPUT:



Program 3

AIM:

To introduce to the classification of Data Mining techniques.

THEORY:

Data mining is the process of discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems. Data mining is an interdisciplinary subfield of computer science and statistics with an overall goal to extract information (with intelligent methods) from a data set and transform the information into a comprehensible structure for further use. Data mining is the analysis step of the "knowledge discovery in databases" process or KDD.

There are several techniques that have been developing and are being used in data mining projects:

- **Classification** is a complex data mining technique that forces you to collect various attributes together into discernible categories, which you can then use to draw further conclusions or serve some function. For example, if you're evaluating data on individual customers' financial backgrounds and purchase histories, you might be able to classify them as "low," "medium," or "high" credit risks.
- **Clustering** is very similar to classification but involves grouping chunks of data together based on their similarities. For example, you might choose to cluster different demographics of your audience into different packets based on how much disposable income they have, or how often they tend to shop at your store.
- **Outlier analysis** determines any anomalies in datasets. Once organizations find aberrations in their data, it becomes easier to understand why these anomalies happen and prepare for any future occurrences to best achieve business objectives. For instance, if there's a spike in the usage of transactional systems for credit cards at a certain time of day, organizations can capitalize on this information by figuring out why it's happening to optimize their sales during the rest of the day.
- **Decision trees** are a specific type of predictive model that lets organizations effectively mine data. Technically, a decision tree is part of machine learning, but it is more popularly known as a white box machine learning technique because of its extremely straightforward nature. A decision tree enables users to clearly understand how the data inputs affect the outputs. When various decision tree models are combined, they create predictive analytics models known as a random forest. Complicated random forest models are considered black-box machine learning techniques because it's not always easy to understand their outputs based on their inputs. In most cases, however, this basic form of ensemble modelling is more accurate than using decision trees on their own.
- **Association** is related to tracking patterns but is more specific to dependently linked variables. In this case, you'll look for specific events or attributes that are highly correlated with another event or attribute; for example, you might notice that when your customers buy a specific item, they also often buy a second, related item. This is usually what's used to populate "people also bought" sections of online stores.
- **Prediction** is one of the most valuable data mining techniques since it's used to project the types of data you'll see in the future. In many cases, just recognizing and understanding historical trends is enough to chart a somewhat accurate prediction of what will happen in the future. For example, you might review consumers' credit histories and past purchases to predict whether they'll be a credit risk in the future.
- One of the most basic techniques in data mining is learning to recognize **sequential patterns** in your data sets. This is usually recognition of some aberration in your data happening at regular intervals or an ebb and flow of a certain variable over time. For example, you might see that your sales of a certain product seem to spike just before the holidays, or notice that warmer weather drives more people to your website.
- **Regression**, used primarily as a form of planning and modelling, is used to identify the likelihood of a certain variable, given the presence of other variables. For example, you could use it to project a certain price, based on other factors like availability, consumer demand, and competition. More specifically, regression's main focus is to help you uncover the exact relationship between two (or more) variables in a given data set.

Program 4

AIM:

To introduce to Attribute Relation File Format (ARFF).

THEORY:

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the Weka machine learning software. As the name suggests the file contains a list of attributes and one class attribute. The attribute relation file format is broadly divided into two portions:

- Header Field
 - The header field describes the name of the attributes, type of relation and their datatypes that are present in the data file. The main difference between the .csv and .arff file is that in .csv files you will find the values of the attributes just below their name, but, in .arff files, the name of the attributes is specified separately followed by the data which is present in a separate data field. The basic syntax for writing the attribute name in the header portion is as follows:
@attribute < attribute – name > < datatype >
- Data Field
 - This field contains the data values of the attributes mentioned above in the attribute field these are the values that will be used by our model to perform prediction and to determine the amount of accuracy that can be provided in the result of our model. The data present is separated by the comas under the heading of @data. The data as mentioned above in the attributes field can be as follows:
 - Numerical: Numeric attributes can be real or integer numbers.
 - Nominal: Nominal values are defined by providing an <nominal-specification> listing the possible values: {< nominal – name1 >, < nominal – name2 >, ...}. For example, the class value of the Iris dataset can be defined as follows:
@ATTRIBUTE class {Iris – setosa, Iris – versicolor, Iris – virginica}
 - String: String attributes allow us to create attributes containing arbitrary textual values. This is very useful in text-mining applications, as we can create datasets with string attributes, then write Weka Filters to manipulate strings (like StringToWordVectorFilter). String attributes are declared as follows:
@ATTRIBUTE LCC string
 - Date-time format: Date attribute declarations take the form: *@attribute < name > date [< date – format >]* where <name> is the name for the attribute and <date-format> is an optional string specifying how date values should be parsed and printed.

OUTPUT:

Sample ARFF file:

```
@RELATION iris

@ATTRIBUTE sepallength NUMERIC
@ATTRIBUTE sepalwidth NUMERIC
@ATTRIBUTE petallength NUMERIC
@ATTRIBUTE petalwidth NUMERIC
@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}

@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
```

Program 5

AIM:

To perform pre-processing, classification and visualization techniques on the customer dataset.

THEORY:

Data Pre-processing: Data pre-processing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data-gathering methods are often loosely controlled, resulting in out-of-range values (e.g., Income: -100), impossible data combinations (e.g., Sex: Female, Pregnant: Yes), missing values, etc.

To predict nominal or numeric quantities, we have classifiers in Weka. Available learning schemes are decision-trees and lists, support vector machines, instance-based classifiers, logistic regression, and Bayes' nets. Once the data has been loaded, all the tabs are enabled. Based on the requirements and by trial and error, we can find out the most suitable algorithm to produce an easily understandable representation of data.

The user can see the final piece of the puzzle, derived throughout the process. It allows users to visualize a 2D representation of data and is used to determine the difficulty of the learning problem. We can visualize single attributes (1D) and pairs of attributes (2D), and rotate 3D visualizations in Weka. It has the Jitter option to deal with nominal attributes and to detect 'hidden' data points.

The customer dataset initially has 6 attributes/features and 15 instances. The last attribute in the dataset is the label which checks on the basis of relation among attributes that given entity is customer or not. The attributes are age, salary, marital status, children and label.

PROCEDURE:

- **Load the dataset:** Open the ARFF file by clicking the 'Open file' button
- **Attribute Selection:** In a dataset, some of the attributes are not required and are also responsible for the skewed nature and problems in classification.
- After obtaining the ranks of the attributes the ones with low ranks are removed.
- **Nominal to Binary transformation of features** - Some of the attributes have values in the form of Yes/No, True/false but it is better to have binary values in the form of 0/1 as the input in the model becomes uniform.
- **Visualization of attributes/features**- All the attributes are represented separately along with the values. This feature is present in the visualization tab in the Weka Explorer.
- **Visualization of relation among attributes** - The relation among attributes helps in finding the connections among various attributes and how one attribute contributes towards the others.
- In the 'Classify' tab of the WEKA explorer we have various options that which algorithm to use for the classification. Here we have used the J-48 algorithm which takes the decision tree approach.

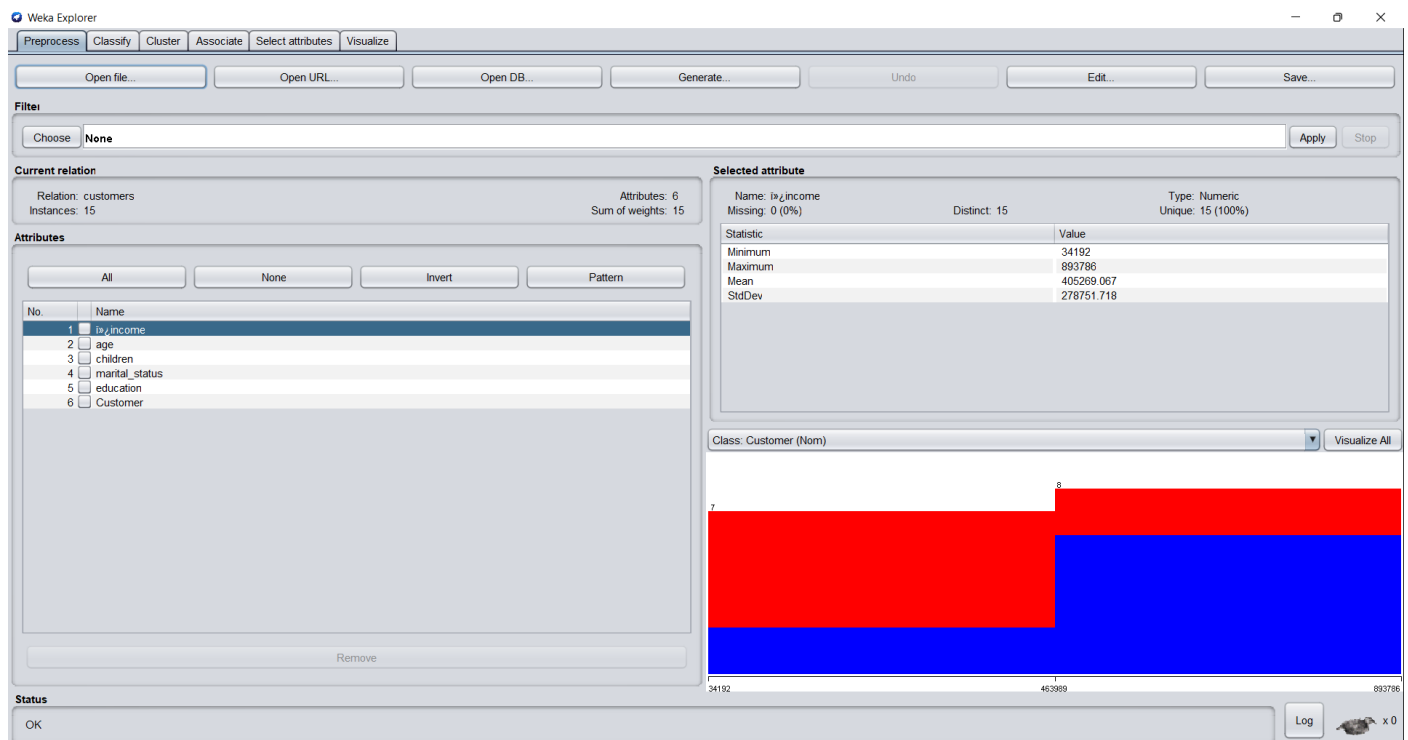
OUTPUT:

The customer dataset initially has 6 attributes/features and 15 instances. The last attribute in the dataset is the label which checks on the basis of relation among attributes that given entity is customer or not. The attributes are age, salary, marital status, children and label.

```
customers.arff - Notepad
File Edit Format View Help
@RELATION customers

@ATTRIBUTE income REAL
@ATTRIBUTE age REAL
@ATTRIBUTE children REAL
@ATTRIBUTE marital_status {Divorced,Married,Unmarried}
@ATTRIBUTE education {Doctorate,Graduate,School Graduate}
@ATTRIBUTE customer {No,Yes}

@DATA
34192,50,3,Married,School Graduate,Yes
171128,42,0,Unmarried,Doctorate,No
479328,39,3,Married,Graduate,Yes
465837,30,4,Married,Graduate,Yes
269113,47,2,Married,Graduate,No
238649,41,0,Unmarried,Graduate,No
561729,54,2,Divorced,Doctorate,No
59965,49,1,Married,School Graduate,No
757268,60,5,Married,Doctorate,Yes
814975,59,5,Married,Doctorate,Yes
566778,30,1,Divorced,Graduate,Yes
474481,44,0,Unmarried,Doctorate,Yes
893786,28,2,Married,Graduate,No
133272,30,2,Married,Graduate,No
158535,29,2,Married,Graduate,Yes
```



Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Attribute Evaluator

Choose InfoGainAttributeEval

Search Method

Choose Ranker -T -1.7978931348623157E308 -N -1

Attribute Selection Mode

☒ Use full training set
☐ Cross-validation Folds 10
Seed 1

(Nom) Customer

Start Stop

Result list (right-click for options)

22.01.25 - Ranker + InfoGainAttributeEval

Attribute selection output

```

Attributes: 6
1>income
age
children
marital_status
education
Customer
Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:
Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 6 Customer):
Information Gain Ranking Filter

Ranked attributes:
0.40926 3 children
0.0325 4 marital_status
0.00647 5 education
0 2 age
0 1 1>income

Selected attributes: 3,4,5,2,1 : 5

```

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

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

Filter

Choose None Apply Stop

Current relation

Relation: customers
Instances: 15

Attributes: 6
Sum of weights: 15

Attributes

All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> 1>income
2	<input checked="" type="checkbox"/> age
3	<input type="checkbox"/> children
4	<input type="checkbox"/> marital_status
5	<input type="checkbox"/> education
6	<input type="checkbox"/> Customer

Remove

Remove selected attributes.

Selected attribute

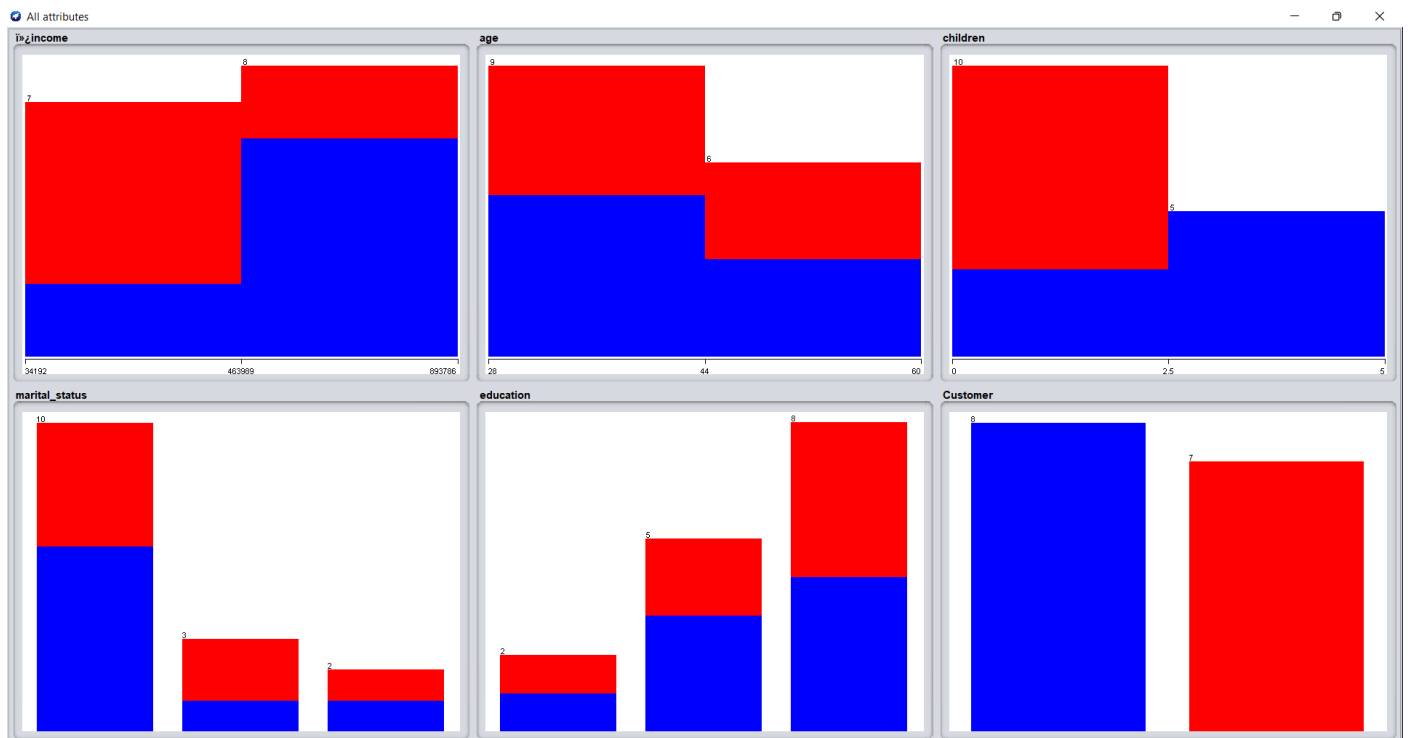
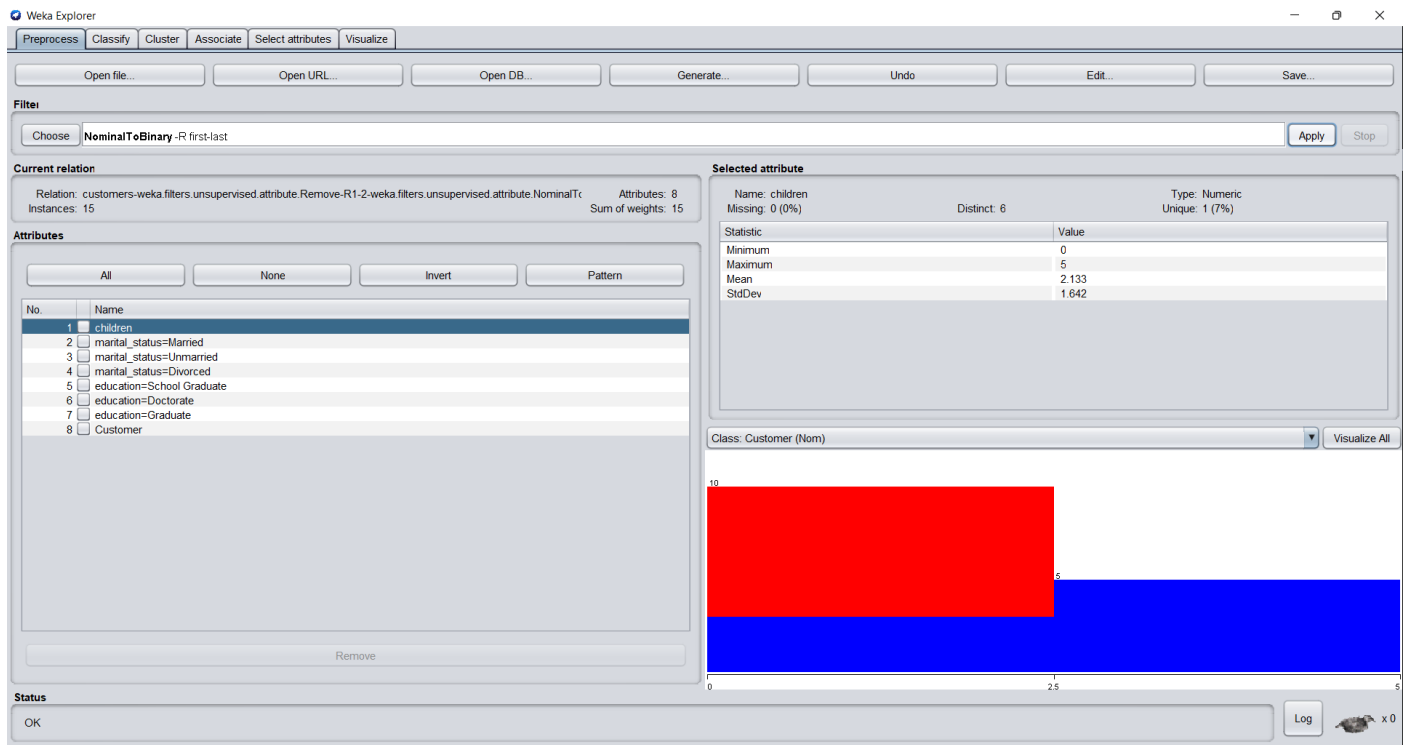
Name: age
Missing: 0 (0%)
Distinct: 13
Type: Numeric
Unique: 12 (80%)

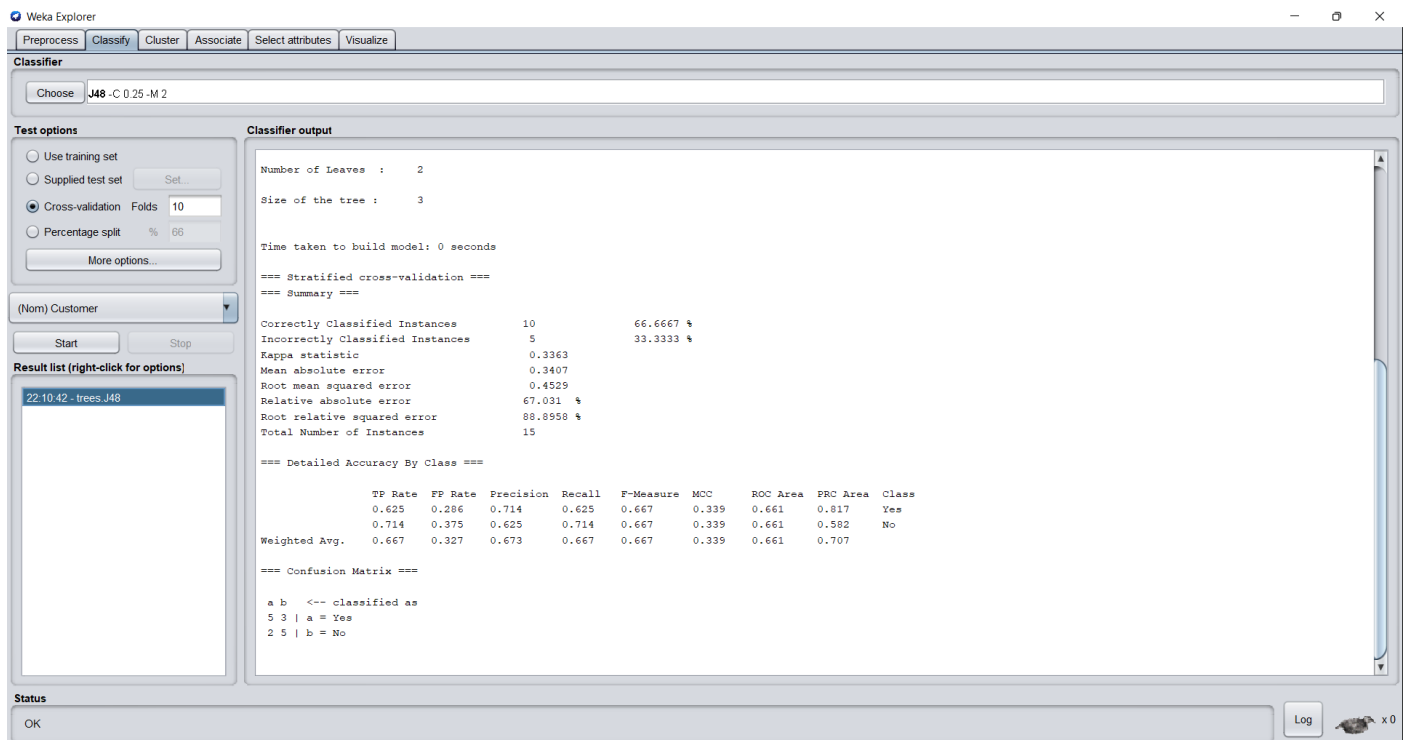
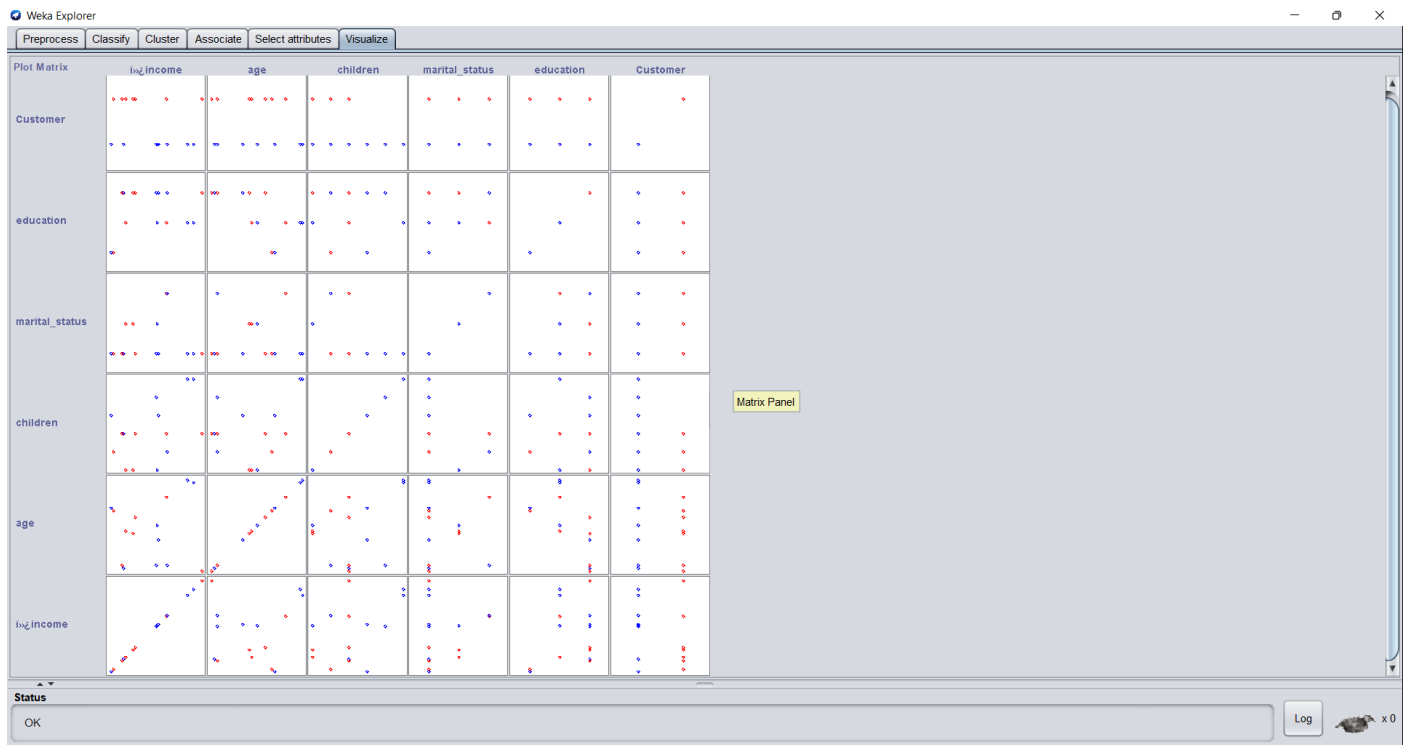
Statistic	Value
Minimum	28
Maximum	60
Mean	42.133
StdDev	11.038

Class: Customer (Nom) Visualize All

Status

OK Log x 0





Program 6

AIM:

To perform pre-processing, classification and visualization techniques on the agriculture dataset.

PROCEDURE:

- Loading the dataset- The dataset has to be imported in ARFF format, if it is csv format, convert it to arff format. In Weka it is done using opening the file location
- Removing the missing values - There were certain null values in the dataset which were removed as they do not contribute in any ways towards the dataset.
- Attribute Selection - In a dataset, some of the attributes are not required and are also responsible for the skewed nature and problems in classification. After obtaining the ranks of the attributes the ones with low ranks are removed i.e., Frosts, altitude and rep as they don't contribute in classification.
- Removing the feature with low ranks- After the ranks are obtained the features/attributes with low ranks are removed.
- Nominal to Binary transformation of features - Some of the attributes have values in the form of Yes/No, True/false but it is better to have binary values in the form of 0/1 as the input in the model becomes uniform.
- Visualization of attributes/features- All the attributes are represented separately along with the values. This feature is present in the visualization tab in the Weka Explorer.
- Visualization of relation among attributes - The relation among attributes helps in finding the connections among various attributes and how one attribute contributes towards the others.
- Perform classification and obtain the results- In the classification tab of the Weka explorer we have various options that which algorithm to use for the classification. Here we have used the J-48 algorithm which takes the decision tree approach.

OUTPUT:

```
eucalyptusarff - Notepad
File Edit Format View Help
@relation eucalyptus

@attribute 'Abbrev' {Cra,Cly,Nga,Wai,K81,Wak,K82,WSp,K83,Lon,Puk,Paw,K81a,Mor,Wen,WSh}
@attribute 'Rep' numeric
@attribute 'Locality' {Central_Hawkes_Bay,Northern_Hawkes_Bay,Southern_Hawkes_Bay,Central_Hawkes_Bay_(coastal),Central_Wairarapa,South_Wairarapa,Southern_Hawkes_Bay_(coastal),Central_Povert
@attribute 'Map_Ref' {N135_382/137,N116_848/985,N145_874/586,N142_377/957,N158_344/626,N162_081/300,N158_343/625,N151_912/221,N162_097/424,N166_063/197,N146_273/737,N141_295/063,N98_539/567
@attribute 'Latitude' {39_38,39_00,40_11,39_50,40_57,41_12,40_36,41_08,41_16,40_00,39_43,82_32}
@attribute 'Altitude' numeric
@attribute 'Rainfall' numeric
@attribute 'Frosts' numeric
@attribute 'Year' numeric
@attribute 'Sp' {co,fr,ma,nd,ni,ob,ov,pu,rd,si,mn,ag,bxs,br,el,fa,jo,ka,re,sm,ro,nc,am,cr,pa,ra,te}
@attribute 'PMcno' numeric
@attribute 'DBH' numeric
@attribute 'Ht' numeric
@attribute 'Surv' numeric
@attribute 'Vig' numeric
@attribute 'Ins_res' numeric
@attribute 'Stem_Fm' numeric
@attribute 'Crown_Fm' numeric
@attribute 'Brnch_Fm' numeric
@attribute 'Utility' {none,low,average,good,best}

@data

Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,co,1520,18.45,9.96,40,4,3,3.5,4,3.5,good
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,fr,1487,13.15,9.65,90,4.5,4,3.5,3.5,3,best
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ma,1362,10.32,6.5,50,2.3,2.5,3,3.5,3,low
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,nd,1596,14.8,9.48,70,3.7,3,3.3,4,3.5,good
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ni,2088,14.5,10.78,90,4,2,7,3,3,3,3,good
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ob,1522,17.01,12.28,70,5,4,5,4,4,5,best
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ov,1521,13.93,9.77,80,4,2,3,3,3,3,good
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,pu,1523,19.05,11.26,100,5,4,5,3.5,4,2,best
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,rd,1524,7.62,6.59,80,2.5,2.5,3,3.3,3.5,3.5,average
Cra,1,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,si,1525,14.75,9.13,50,4,4,3,3,3,3,good
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,co,1520,8.15,6.72,70,1,5,3,3,3,3,low
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,fr,1487,15.07,7.17,10,4,4,3,3,3,2,good
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ma,1362,8.99,6.05,60,2.5,3,4,4,4,low
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,nd,1596,13.13,6.08,80,3.5,4,4,3.5,3.5,best
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ni,2088,10.78,8.93,100,3,2,3,3,3,good
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ob,1522,13.08,9.79,90,5,4,5,5,4,4,best
Cra,2,Central_Hawkes_Bay,N135_382/137,39_38,100,850,-2,1980,ov,1521,10.27,8.32,100,3,1,8,3,3,3,average
```

Weka Explorer

PreprocessClassifyClusterAssociateSelect attributesVisualize

Open file...Open URL...Open DB...Generate...UndoEdit...Save...

FilterChooseNominalToBinary - R first-lastApplyStop

Current relationRelation: eucalyptusInstances: 736Attributes: 20Sum of weights: 736

AttributesAllNoneInvertPattern

No.	Name
1	<input checked="" type="checkbox"/> Abbrev
2	<input type="checkbox"/> Rep
3	<input type="checkbox"/> Locality
4	<input type="checkbox"/> Map_Ref
5	<input type="checkbox"/> Latitude
6	<input type="checkbox"/> Altitude
7	<input type="checkbox"/> Rainfall
8	<input type="checkbox"/> Frosts
9	<input type="checkbox"/> Year
10	<input type="checkbox"/> Sp
11	<input type="checkbox"/> PMcno
12	<input type="checkbox"/> DBH
13	<input type="checkbox"/> Ht
14	<input type="checkbox"/> Surv
15	<input type="checkbox"/> Vig
16	<input type="checkbox"/> Ins_res
17	<input type="checkbox"/> Stem_Fm
18	<input type="checkbox"/> Crown_Fm
19	<input type="checkbox"/> Brnch_Fm
20	<input type="checkbox"/> Utility

Remove

StatusOK

Selected attributeName: AbbrevMissing: 0 (0%)Distinct: 16Type: NominalUnique: 0 (0%)

No.	Label	Count	Weight
1	Cra	30	30.0
2	Cly	24	24.0
3	Nga	22	22.0
4	Wai	70	70.0
5	K81	65	65.0
6	Wak	73	73.0
7	K82	45	45.0
8	WSp	59	59.0
9	K83	49	49.0
10	Lon	53	53.0
11	Puk	84	84.0

Class: Utility (Nom)Visualize All

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Attribute Evaluator

Choose InfoGainAttributeEval

Search Method

Choose Ranker -T -1.7978931348623157E308 -N -1

Attribute Selection Mode

☒ Use full training set
☐ Cross-validation Folds 10 Seed 1

(Nom) Utility

Start Stop

Result list (right-click for options)

22.32 19 - Ranker + InfoGainAttributeEval

Attribute selection output

Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 20 Utility):
Information Gain Ranking Filter

Ranked attributes:

0.6404	15	Vig
0.4275	10	Sp
0.4244	1	Abbrev
0.4077	4	Map_Ref
0.4001	5	Latitude
0.2824	9	Year
0.2782	7	Rainfall
0.274	13	Ht
0.2465	3	Locality
0.2416	11	PMCno
0.2003	12	DBH
0.1607	16	Ins_res
0.147	17	Stem_Fm
0.1079	6	Altitude
0.0968	18	Crown_Fm
0.0759	14	Surv
0.0352	8	Frosts
0	2	Rep
0	19	Brnch_Fm

Selected attributes: 15,10,1,4,5,9,7,13,3,11,12,16,17,6,18,14,0,2,19 : 19

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

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

Filter

Choose NominalToBinary -R first-last Apply Stop

Current relation

Relation: eucalyptus Instances: 736 Attributes: 20 Sum of weights: 736

Selected attribute

Name: Brnch_Fm Missing: 69 (9%) Distinct: 28 Type: Numeric Unique: 6 (1%)

Statistic	Value
Minimum	0
Maximum	5
Mean	2.941
StdDev	0.788

Class: Utility (Nom) Visualize All

Status

OK Log x 0

Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

NominalToBinary -R first-last

Apply

Stop

Current relation

Relation: eucalyptus-weka.filters.unsupervised.attribute.NominalToBinary-Rfirst-last

Instances: 736

Attributes: 92

Sum of weights: 736

Attributes

All

None

Invert

Pattern

No.	Name
1	<input checked="" type="checkbox"/> Abbrev=Cra
2	<input type="checkbox"/> Abbrev=Cly
3	<input type="checkbox"/> Abbrev=Nga
4	<input type="checkbox"/> Abbrev=Wai
5	<input type="checkbox"/> Abbrev=K81
6	<input type="checkbox"/> Abbrev=Wak
7	<input type="checkbox"/> Abbrev=K82
8	<input type="checkbox"/> Abbrev=WSp
9	<input type="checkbox"/> Abbrev=K83
10	<input type="checkbox"/> Abbrev=Lon
11	<input type="checkbox"/> Abbrev=Puk
12	<input type="checkbox"/> Abbrev=Paw
13	<input type="checkbox"/> Abbrev=K81a
14	<input type="checkbox"/> Abbrev=Mor
15	<input type="checkbox"/> Abbrev=Wen
16	<input type="checkbox"/> Abbrev=WSh
17	<input type="checkbox"/> Rep
18	<input type="checkbox"/> Locality=Central_Hawkes_Bay
19	<input type="checkbox"/> Locality=Northern_Hawkes_Bay
20	<input type="checkbox"/> Locality=Southern_Hawkes_Bay
21	<input type="checkbox"/> Locality=Central_Hawkes_Bay_(coastal)

Remove

Selected attribute

Name: Abbrev=Cra

Missing: 0 (0%)

Distinct: 2

Type: Numeric

Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.041
StdDev	0.198

Class: Utility (Nom)

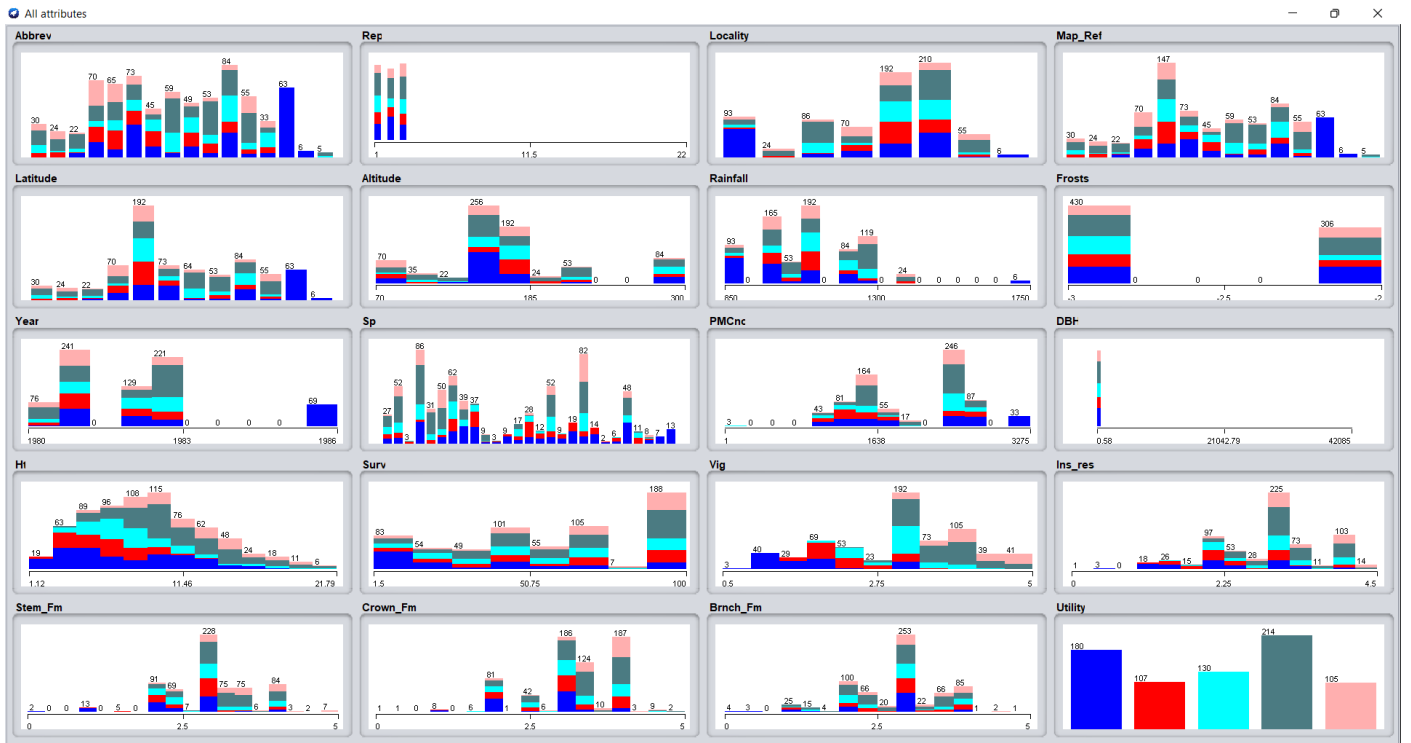
Visualize All

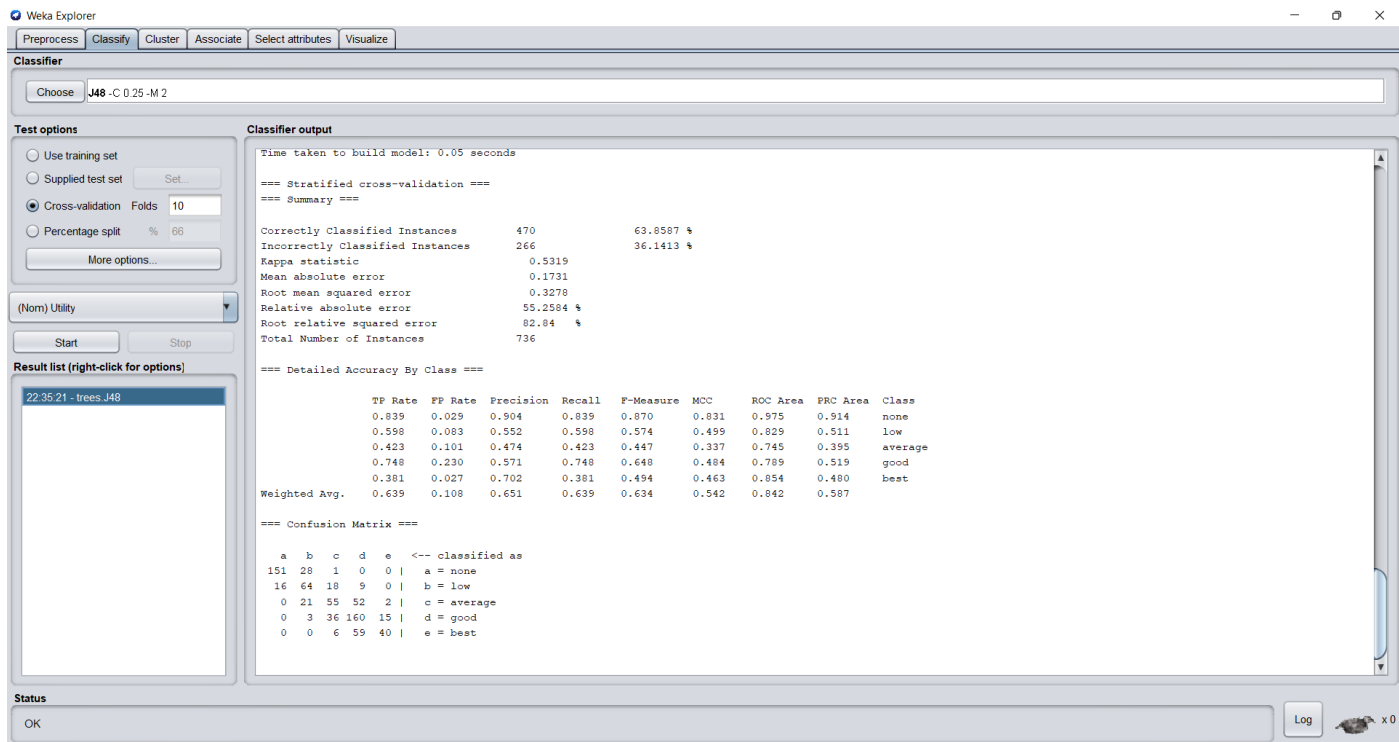
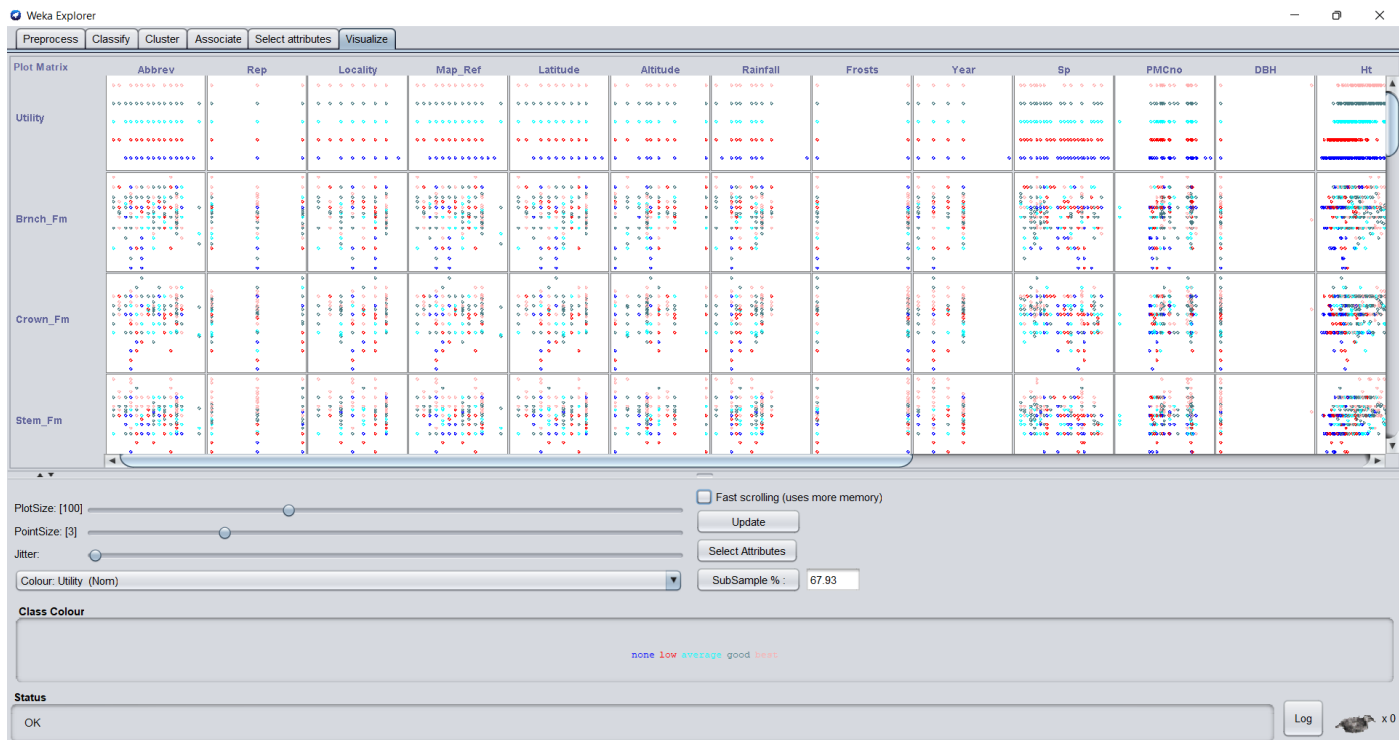
Status

OK

Log

x 0





Program 7

AIM:

To perform preprocessing, classification and visualization techniques on weather dataset.

PROCEDURE:

- Loading the dataset- The dataset has to be imported in ARFF format, if it is csv format, convert it to arff format. In Weka it is done using opening the file location
- Removing the missing values - There were certain null values in the dataset which were removed as they do not contribute in any ways towards the dataset.
- Attribute Selection - In a dataset, some of the attributes are not required and are also responsible for the skewed nature and problems in classification. After obtaining the ranks of the attributes the ones with low ranks are removed i.e., Frosts, altitude and rep as they don't contribute in classification.
- Removing the feature with low ranks- After the ranks are obtained the features/attributes with low ranks are removed.
- Nominal to Binary transformation of features - Some of the attributes have values in the form of Yes/No, True/false but it is better to have binary values in the form of 0/1 as the input in the model becomes uniform.
- Visualization of attributes/features- All the attributes are represented separately along with the values. This feature is present in the visualization tab in the Weka Explorer.
- Visualization of relation among attributes - The relation among attributes helps in finding the connections among various attributes and how one attribute contributes towards the others.
- Perform classification and obtain the results- In the classification tab of the Weka explorer we have various options that which algorithm to use for the classification. Here we have used the J-48 algorithm which takes the decision tree approach.

OUTPUT:

weather.arff - Notepad

File Edit Format View Help

@relation weather

@attribute outlook {sunny, overcast, rainy}

@attribute temperature real

@attribute humidity real

@attribute windy {TRUE, FALSE}

@attribute pleasant {yes, no}

@data

sunny,85,85,FALSE,no

sunny,80,90,TRUE,no

overcast,83,86,FALSE,yes

rainy,70,96,FALSE,yes

rainy,68,80,FALSE,yes

rainy,65,70,TRUE,no

overcast,64,65,TRUE,yes

sunny,72,95,FALSE,no

sunny,69,70,FALSE,yes

rainy,75,80,FALSE,yes

sunny,75,70,TRUE,yes

overcast,72,90,TRUE,yes

overcast,81,75,FALSE,yes

rainy,71,91,TRUE,no

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

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

Filter: Choose None Apply Stop

Current relation: Relation: weather Instances: 14 Attributes: 5 Sum of weights: 14

Attributes: All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> outlook
2	<input type="checkbox"/> temperature
3	<input type="checkbox"/> humidity
4	<input type="checkbox"/> windy
5	<input type="checkbox"/> pleasant

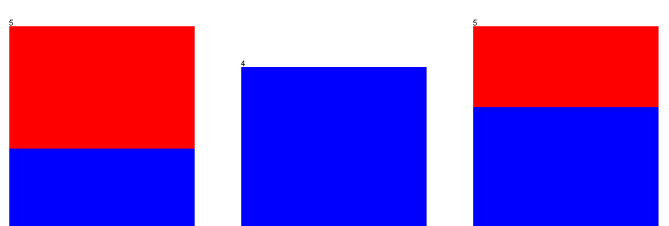
Remove

Status: OK Log x 0

Selected attribute: Name: outlook Missing: 0 (0%) Distinct: 3 Type: Nominal Unique: 0 (0%)

No.	Label	Count	Weight
1	sunny	5	5.0
2	overcast	4	4.0
3	rainy	5	5.0

Class: pleasant (Nom) Visualize All



Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Attribute Evaluator

Choose InfoGainAttributeEval

Search Method

Choose Ranker -T -1.7978931348623157E308 -N -1

Attribute Selection Mode

☒ Use full training set
☐ Cross-validation Folds 10 Seed 1

(Nom) pleasant

Start Stop

Result list (right-click for options)

22.4853 - Ranker + InfoGainAttributeEval

Attribute selection output

```
Relation: weather
Instances: 14
Attributes: 5
  outlook
  temperature
  humidity
  windy
  pleasant
Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:
  Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 5 pleasant):
  Information Gain Ranking Filter

Ranked attributes:
0.2467 1 outlook
0.0481 4 windy
0      3 humidity
0      2 temperature

Selected attributes: 1,4,3,2 : 4
```

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

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

Filter

Choose None Apply Stop

Current relation

Relation: weather Attributes: 5
Instances: 14 Sum of weights: 14

Attributes

All None Invert Pattern

No.	Name
1	<input type="checkbox"/> outlook
2	<input checked="" type="checkbox"/> temperature
3	<input checked="" type="checkbox"/> humidity
4	<input type="checkbox"/> windy
5	<input type="checkbox"/> pleasant

Remove

Remove selected attributes.

Selected attribute

Name: temperature
Missing: 0 (0%) Distinct: 12 Type: Numeric
Unique: 10 (71%)

Statistic	Value
Minimum	64
Maximum	85
Mean	73.571
StdDev	6.572

Class: pleasant (Nom) Visualize All

Status

OK Log x 0

Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

NominalToBinary -R first-last

Apply

Stop

Current relation

Relation: weather-weka.filters.unsupervised.attribute.NominalToBinary-Rfirst-last

Instances: 14

Attributes: 7

Sum of weights: 14

Attributes

All

None

Invert

Pattern

No.	Name
1	<input checked="" type="checkbox"/> outlook=sunny
2	<input type="checkbox"/> outlook=overcast
3	<input type="checkbox"/> outlook=rainy
4	<input type="checkbox"/> temperature
5	<input type="checkbox"/> humidity
6	<input type="checkbox"/> windy=FALSE
7	<input type="checkbox"/> pleasant

Remove

Selected attribute

Name: outlook=sunny

Missing: 0 (0%)

Distinct: 2

Type: Numeric

Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.357
StdDev	0.497

Class: pleasant (Nom)

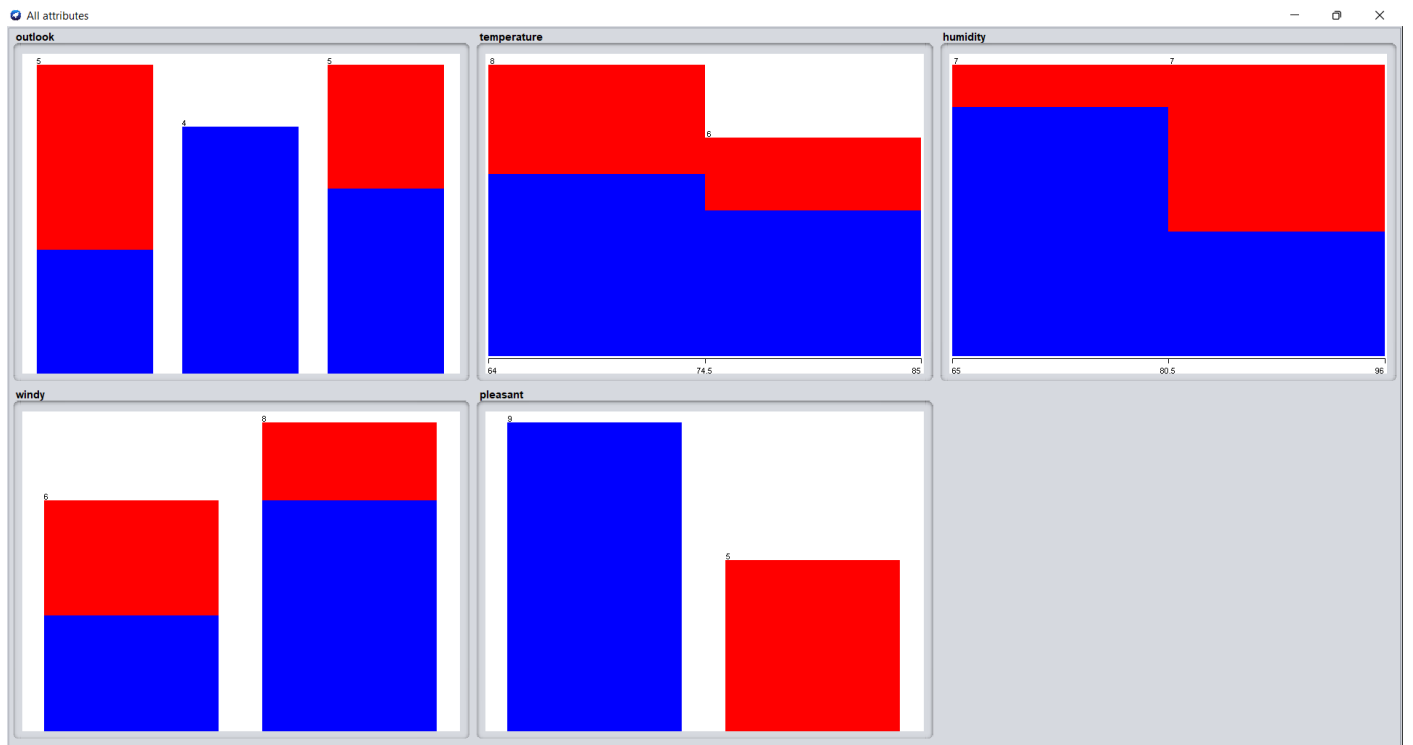
Visualize All

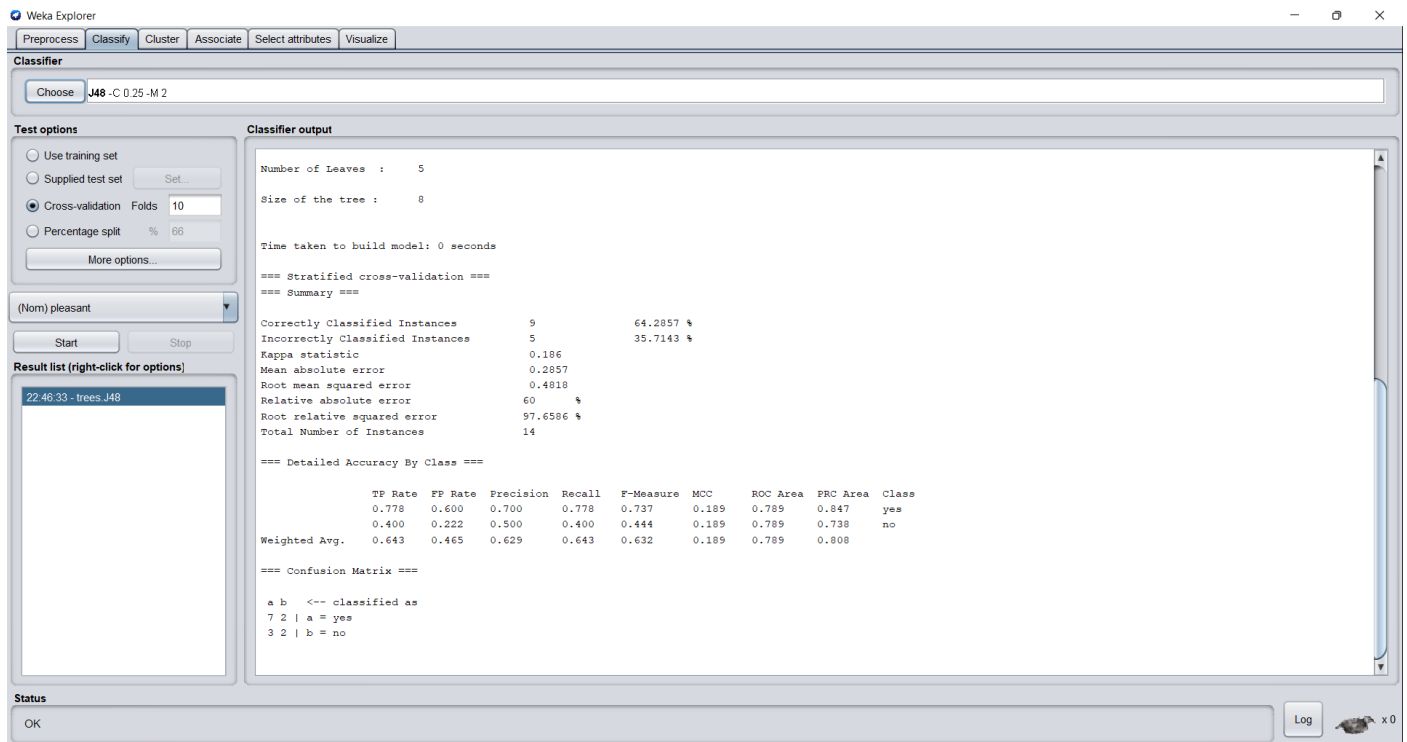
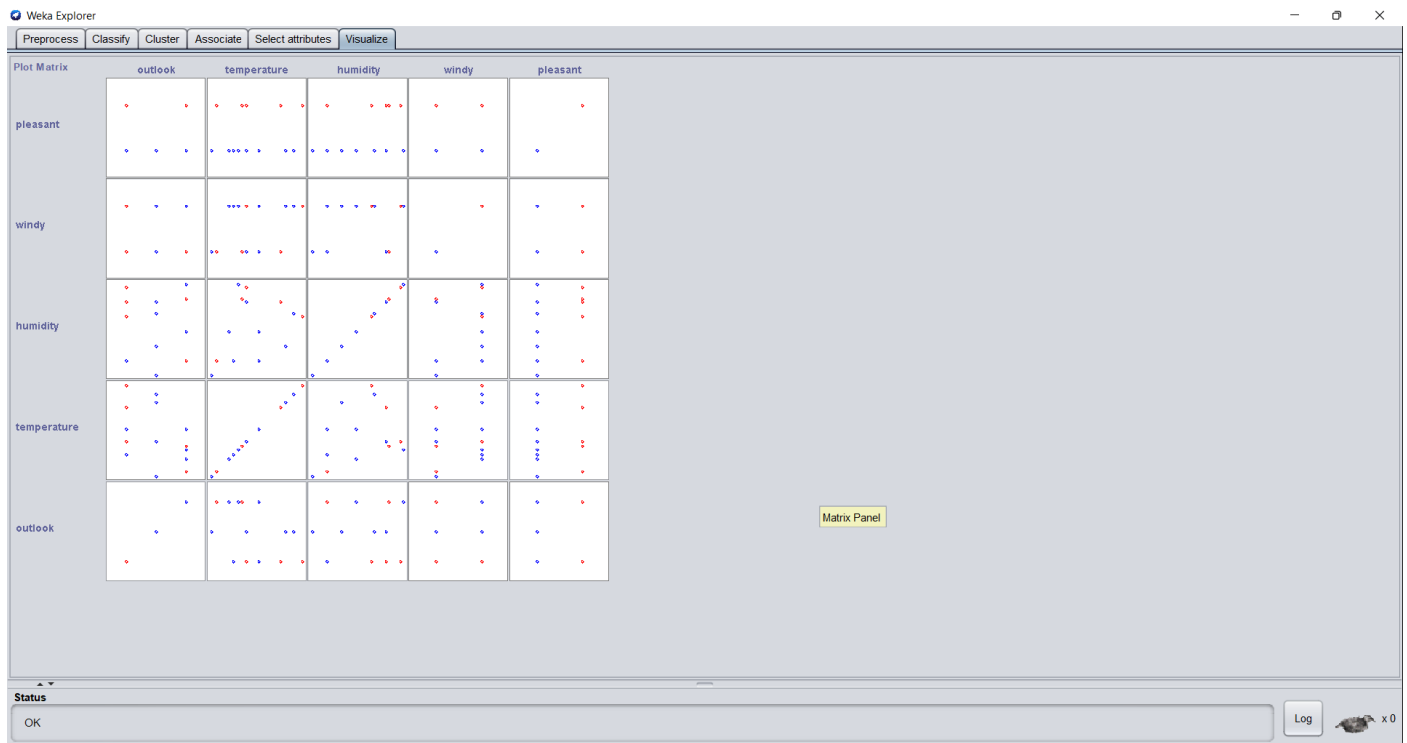
Status

OK

Log

x 0





Program 8

AIM:

To perform clustering techniques on weather dataset using k-means and coweb.

THEORY:

Clustering analysis or simply Clustering is basically an Unsupervised learning method that divides the data points into a number of specific batches or groups, such that the data points in the same groups have similar properties and data points in different groups have different properties in some sense. Some clustering techniques produce a hierarchical clustering - a tree of clusters. Clusters at one level break up into child sub clusters, and so on. Leaves contain indivisible clusters consisting of one or more instances. Cobweb is a hierarchical clustering method based on a measure of clustering quality called "category utility". The category utility is denoted CU (clusters). Higher values of CU indicate better clustering's. This experiment illustrates how clustering algorithms like k-means and Coweb are performed on weather dataset using Weka. The dataset contains 14 instances.

K-means:

```
k-means (k, attributes, instances, epsilon) {  
    choose k cluster centers c1..ck randomly in attribute space  
    changing <- true  
    while (changing) do {  
        for each instance I in instances:  
            set cluster(I) <- cluster(argmin(dist(I,c(k))))  
        for each j up to k:  
            oldcj <- cj  
            cj <- centroid(cluster(cj))  
        changing = ( dist((c1..ck),(oldc1...oldck)) > epsilon )  
    }  
    return cluster(c1)...cluster(ck)  
}
```

Coweb:

```
incrementallyCluster(instances, attributes):  
    initialize clusters <- single non-leaf root node  
    for each training instance I:  
        addInstance(I, root)  
  
addInstance(I, root):
```

if root is not a leaf:

baseCU = CU(clusters with I added as leaf at depth one)

bestCU <- max_C (CU(clusters with I added to C), baseCU)

bestC <-argmax_C(CU(clusters with I added to C), baseCU)

nextC <-argrank2_C(CU(clusters with I added to C),baseCU)

if (CU(clusters with bestC and nextC merged) > bestCU):

clusters = same with bestC, nextC merged into C*

addInstance(I, C*) // recursion

else if (CU(clusters with bestC split) > (1+cutoff%)*bestCU):

clusters = clusters with bestC split (unrooted)

addInstance(I, clusters)

else:

addInstance(I, bestC)

PROCEDURE:

- Run the WEKA explorer and load the data file weather.arff in preprocessing interface.
- In order to perform clustering select the 'cluster' tab in the explorer and click on the choose button. This step results in a dropdown list of available clustering algorithms.
- In this case we select 'simple k-means'.
- Next click in text button to the right of the choose button to get popup window shown in the screenshots. In this window we enter six on the number of clusters and we leave the value of the seed on as it is. The seed value is used in generating a random number which is used for making the internal assignments of instances of clusters.
- Once of the option have been specified. We run the clustering algorithm there we must make sure that they are in the 'cluster mode' panel. The use of training set option is selected and then we click 'start' button. This process and resulting window are shown in the following screenshots.
- The result window shows the centroid of each cluster as well as statistics on the number and the percent of instances assigned to different clusters. Here clusters centroid is means vectors for each cluster. These clusters can be used to characterize the cluster.
- Another way of understanding characteristics of each cluster through visualization, we can do this, try right clicking the result set on the result. List panel and selecting the visualize cluster assignments.

- We can assure that resulting dataset which included each instance along with its assign cluster. To do so we click the save button in the visualization window and save the result student k-means. The top portion of this file is shown in the following figure.
- Next, we do the same clustering Using Coweb and visualize the clustering using tree.

OUTPUT:

The screenshot shows the Weka Explorer interface with the 'Cluster' tab selected. The 'Choose' button is set to 'Cobweb -A 1.0 -C 0.0026209479177387815 -S 42'. The 'Cluster mode' section on the left has 'Classes to clusters evaluation' selected, with '(Nom) pleasant' chosen from the dropdown. The 'Store clusters for visualization' checkbox is checked. The 'Cluster output' pane on the right displays the following text:

```
kMeans
=====
Number of iterations: 3
Within cluster sum of squared errors: 11.237456311387234

Initial starting points (random):
Cluster 0: rainy,75,80,FALSE
Cluster 1: overcast,64,65,TRUE

Missing values globally replaced with mean/mode

Final cluster centroids:
Attribute      Full Data      Cluster#
              (14.0)      (9.0)      (5.0)
=====
outlook        sunny        sunny        overcast
temperature    73.5714      75.8889      69.4
humidity       81.6429      84.1111      77.2
windy           FALSE        FALSE        TRUE

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

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

Clustered Instances
0  9 ( 64%)
1  5 ( 36%)
```

The 'Result list (right-click for options)' pane on the left shows '23.04.50 - SimpleKMeans'.

The screenshot shows the Weka Explorer interface with the 'Cluster' tab selected. The 'Choose' button is set to 'SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10'. The 'Cluster mode' section on the left has 'Classes to clusters evaluation' selected, with '(Nom) pleasant' chosen from the dropdown. The 'Store clusters for visualization' checkbox is checked. The 'Cluster output' pane on the right displays the following text:

```

              (14.0)      (9.0)      (5.0)
=====
outlook        sunny        sunny        overcast
temperature    73.5714      75.8889      69.4
humidity       81.6429      84.1111      77.2
windy           FALSE        FALSE        TRUE

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

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

Clustered Instances
0  9 ( 64%)
1  5 ( 36%)

Class attribute: pleasant
Classes to Clusters:
0 1 <-- assigned to cluster
6 3 | yes
3 2 | no

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

Incorrectly clustered instances :      6.0      42.8571 %
```

The 'Result list (right-click for options)' pane on the left shows '23.04.50 - SimpleKMeans'.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Clusterer

Choose Cobweb -A 1.0 -C 0.0028209479177387815 -S 42

Cluster mode

☐ Use training set
☐ Supplied test set Set...
☐ Percentage split % 66
☒ Classes to clusters evaluation
(Nom) pleasant
☒ Store clusters for visualization

Ignore attributes

Start Stop

Result list (right-click for options)

23.04.50 - SimpleKMeans
23.07.19 - Cobweb

Clusterer output

```
=== Clustering model (full training set) ===  
  
Number of merges: 1  
Number of splits: 0  
Number of clusters: 22  
  
node 0 [14]  
| node 1 [8]  
| | node 2 [2]  
| | | leaf 3 [1]  
| | node 2 [2]  
| | | leaf 4 [1]  
| node 1 [8]  
| | leaf 5 [1]  
| node 1 [8]  
| | leaf 6 [1]  
| node 1 [8]  
| | node 7 [3]  
| | | leaf 8 [1]  
| | node 7 [3]  
| | | leaf 9 [1]  
| | node 7 [3]  
| | | leaf 10 [1]  
| node 1 [8]  
| | leaf 11 [1]  
node 0 [14]  
| node 12 [6]  
| | node 13 [2]  
| | | leaf 14 [1]  
| | node 13 [2]  
| | | leaf 15 [1]  
| node 12 [6]  
| | node 16 [2]  
| | | leaf 17 [1]
```

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Clusterer

Choose Cobweb -A 1.0 -C 0.0028209479177387815 -S 42

Cluster mode

☐ Use training set
☐ Supplied test set Set...
☐ Percentage split % 66
☒ Classes to clusters evaluation
(Nom) pleasant
☒ Store clusters for visualization

Ignore attributes

Start Stop

Result list (right-click for options)

23.04.50 - SimpleKMeans
23.07.19 - Cobweb

Clusterer output

```
=== Clustering model (full training set) ===  
  
Number of merges: 1  
Number of splits: 0  
Number of clusters: 22  
  
node 0 [14]  
| node 1 [8]  
| | node 2 [2]  
| | | leaf 3 [1]  
| | node 2 [2]  
| | | leaf 4 [1]  
| node 1 [8]  
| | leaf 5 [1]  
| node 1 [8]  
| | leaf 6 [1]  
| node 1 [8]  
| | node 7 [3]  
| | | leaf 8 [1]  
| | node 7 [3]  
| | | leaf 9 [1]  
| | node 7 [3]  
| | | leaf 10 [1]  
| node 1 [8]  
| | leaf 11 [1]  
node 0 [14]  
| node 12 [6]  
| | node 13 [2]  
| | | leaf 14 [1]  
| | node 13 [2]  
| | | leaf 15 [1]  
| node 12 [6]  
| | node 16 [2]  
| | | leaf 17 [1]  
  
Time taken to build model (full training data) : 0 seconds  
  
=== Model and evaluation on training set ===  
  
Clustered Instances  
  
3 1 ( 7%)  
4 1 ( 7%)  
5 1 ( 7%)  
6 1 ( 7%)  
8 1 ( 7%)  
9 1 ( 7%)  
10 1 ( 7%)  
11 1 ( 7%)  
14 1 ( 7%)  
15 1 ( 7%)  
17 1 ( 7%)  
18 1 ( 7%)  
20 1 ( 7%)  
21 1 ( 7%)
```

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Clusterer

Choose Cobweb -A 1.0 -C 0.0028209479177387815 -S 42

Cluster mode

☐ Use training set
☐ Supplied test set Set...
☐ Percentage split % 66
☒ Classes to clusters evaluation
 (Nom) pleasant
☒ Store clusters for visualization

Ignore attributes

Start Stop

Result list (right-click for options)

23:04:50 - SimpleKMeans
 23:07:19 - Cobweb

Clusterer output

```

11 1 ( 7%)
14 1 ( 7%)
15 1 ( 7%)
17 1 ( 7%)
18 1 ( 7%)
20 1 ( 7%)
21 1 ( 7%)

Class attribute: pleasant
Classes to Clusters:

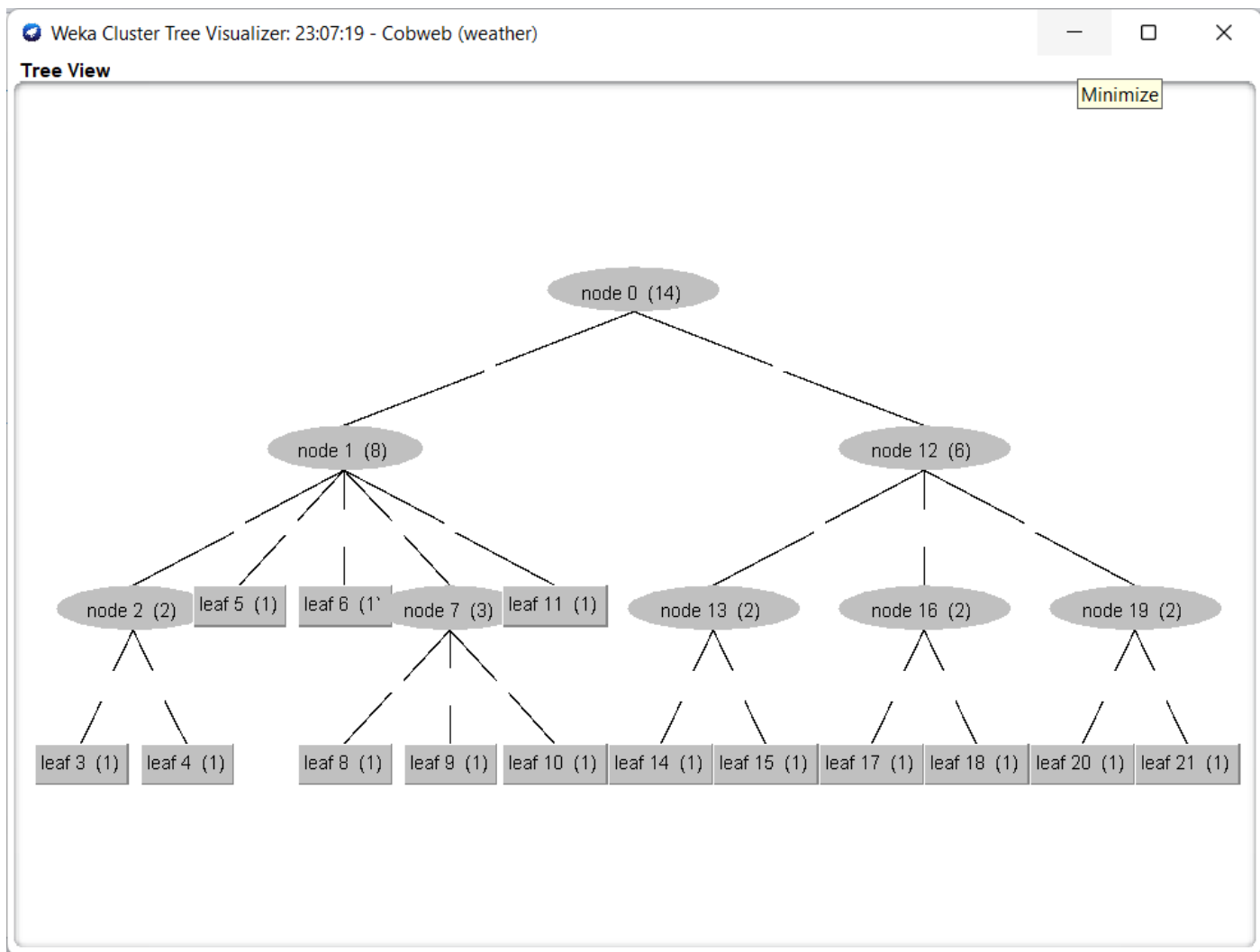
 3  4  5  6  8  9 10 11 14 15 17 18 20 21 <-- assigned to cluster
1  1  1  1  0  0  1  1  1  0  0  0  1  1 | yes
0  0  0  0  1  1  0  0  0  1  1  1  0  0 | no

Cluster 3 <-- No class
Cluster 4 <-- No class
Cluster 5 <-- No class
Cluster 6 <-- No class
Cluster 8 <-- No class
Cluster 9 <-- No class
Cluster 10 <-- No class
Cluster 11 <-- No class
Cluster 14 <-- No class
Cluster 15 <-- No class
Cluster 17 <-- No class
Cluster 18 <-- no
Cluster 20 <-- No class
Cluster 21 <-- yes

Incorrectly clustered instances :      12.0      85.7143 %
  
```

Status

OK Log x 0



Program 9

AIM:

To perform Association technique on Weather dataset.

THEORY:

Association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. It identifies frequent if-then associations called association rules which consists of an antecedent (if) and a consequent (then). There are three common metrics to measure association. Support is an indication of how frequently the items appear in the data. Mathematically, support is the fraction of the total number of transactions in which the item set occurs.

$$\text{Support}(\{X\} \rightarrow \{Y\}) = \frac{\text{Transactions containing both } X \text{ and } Y}{\text{Total number of transactions}}$$

Confidence indicates the number of times the if-then statements are found true. Confidence is the conditional probability of occurrence of consequent given the antecedent.

$$\text{Confidence}(\{X\} \rightarrow \{Y\}) = \frac{\text{Transactions containing both } X \text{ and } Y}{\text{Transactions containing } X}$$

Lift can be used to compare confidence with expected confidence. This says how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is. Mathematically,

$$\text{Lift}(\{X\} \rightarrow \{Y\}) = \frac{(\text{Transactions containing both } X \text{ and } Y) / (\text{Transactions containing } X)}{\text{Fraction of transactions containing } Y}$$

Algorithm:

Apriori(T, ϵ)

$L_1 \leftarrow \{\text{large 1 - itemsets}\}$

$k \leftarrow 2$

while $L_{k-1} \neq \emptyset$

$C_k \leftarrow \{c = a \cup \{b\} \mid a \in L_{k-1} \wedge b \notin a, \{s \subseteq c \mid |s| = k-1\} \subseteq L_{k-1}\}$

for transactions $t \in T$

$D_t \leftarrow \{c \in C_k \mid c \subseteq t\}$

for candidates $c \in D_t$

$\text{count}[c] \leftarrow \text{count}[c] + 1$

$L_k \leftarrow \{c \in C_k \mid \text{count}[c] \geq \epsilon\}$

$k \leftarrow k + 1$

return $\bigcup_k L_k$

PROCEDURE:

- Open the data file in WEKA Explorer. It is presumed that the required data fields have been discretized.
- Clicking on the associate tab will bring up the interface for association rule algorithm.
- We will use Apriori algorithm. This is the default algorithm.
- In order to change the parameters for the run (example support, confidence etc) we click on the text box immediately to the right of the choose button.

OUTPUT:

*contact-lenses.arff - Notepad
File Edit Format View Help

@relation contact-lenses

@attribute age {young, pre-presbyopic, presbyopic}
@attribute spectacle-prescrip {myope, hypermetrope}
@attribute astigmatism {no, yes}
@attribute tear-prod-rate {reduced, normal}
@attribute contact-lenses {soft, hard, none}

@data

%

% 24 instances

%

young,myope,no,reduced,none
young,myope,no,normal,soft
young,myope,yes,reduced,none
young,myope,yes,normal,hard
young,hypermetrope,no,reduced,none
young,hypermetrope,no,normal,soft
young,hypermetrope,yes,reduced,none
young,hypermetrope,yes,normal,hard
pre-presbyopic,myope,no,reduced,none
pre-presbyopic,myope,no,normal,soft
pre-presbyopic,myope,yes,reduced,none
pre-presbyopic,myope,yes,normal,hard
pre-presbyopic,hypermetrope,no,reduced,none
pre-presbyopic,hypermetrope,no,normal,soft
pre-presbyopic,hypermetrope,yes,reduced,none
pre-presbyopic,hypermetrope,yes,normal,none
presbyopic,myope,no,reduced,none
presbyopic,myope,no,normal,none
presbyopic,myope,yes,reduced,none
presbyopic,myope,yes,normal,hard
presbyopic,hypermetrope,no,reduced,none
presbyopic,hypermetrope,no,normal,soft
presbyopic,hypermetrope,yes,reduced,none
presbyopic,hypermetrope,yes,normal,none

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

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

Filter: Choose NumericToNominal -R first-last Apply Stop

Current relation: Relation: contact-lenses-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last Instances: 24 Attributes: 5 Sum of weights: 24

Attributes: All None Invert Pattern

No.	Name
1	age
2	spectacle-prescrip
3	astigmatism
4	tear-prod-rate
5	contact-lenses

Remove

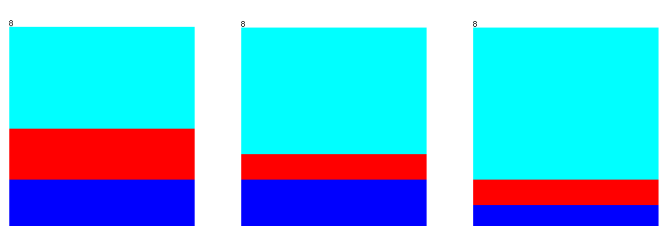
Status: OK

Selected attribute

Name: age Missing: 0 (0%) Distinct: 3 Type: Nominal Unique: 0 (0%)

No.	Label	Count	Weight
1	young	8	8.0
2	pre-presbyopic	8	8.0
3	presbyopic	8	8.0

Class: contact-lenses (Nom) Visualize All



Confidence

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Associator

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

Start Stop

Result list (right-click...)

23:34:43 - Apriori

Associator output

```
tear-prod-rate
contact-lenses
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.2 (5 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 16

Generated sets of large itemsets:

Size of set of large itemsets L(1): 11
Size of set of large itemsets L(2): 21
Size of set of large itemsets L(3): 6

Best rules found:

1. tear-prod-rate=reduced 12 ==> contact-lenses=none 12    <conf:(1)> lift:(1.6) lev:(0.19) [4] conv:(4.5)
2. spectacle-prescrip=myope tear-prod-rate=reduced 6 ==> contact-lenses=none 6    <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
3. spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 ==> contact-lenses=none 6    <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
4. astigmatism=no tear-prod-rate=reduced 6 ==> contact-lenses=none 6    <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
5. astigmatism=yes tear-prod-rate=reduced 6 ==> contact-lenses=none 6    <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)
6. contact-lenses=soft 5 ==> astigmatism=no 5    <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
7. contact-lenses=soft 5 ==> tear-prod-rate=normal 5    <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
8. tear-prod-rate=normal contact-lenses=soft 5 ==> astigmatism=no 5    <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
9. astigmatism=no contact-lenses=soft 5 ==> tear-prod-rate=normal 5    <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)
10. contact-lenses=soft 5 ==> astigmatism=no tear-prod-rate=normal 5    <conf:(1)> lift:(4) lev:(0.16) [3] conv:(3.75)
```

Status

OK Log x 0

Lift

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Associator

Choose Apriori -N 10 -T 1 -C 1.1 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

Start Stop

Result list (right-click...)

23:34:43 - Apriori
23:35:01 - Apriori

Associator output

```
tear-prod-rate
contact-lenses
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.25 (6 instances)
Minimum metric <lift>: 1.1
Number of cycles performed: 15

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10
Size of set of large itemsets L(2): 18
Size of set of large itemsets L(3): 4

Best rules found:

1. tear-prod-rate=reduced 12 ==> spectacle-prescrip=myope contact-lenses=none 6    conf:(0.5) < lift:(1.71)> lev:(0.1) [2] conv:(1.21)
2. spectacle-prescrip=myope contact-lenses=none 7 ==> tear-prod-rate=reduced 6    conf:(0.86) < lift:(1.71)> lev:(0.1) [2] conv:(1.75)
3. tear-prod-rate=reduced 12 ==> astigmatism=no contact-lenses=none 6    conf:(0.5) < lift:(1.71)> lev:(0.1) [2] conv:(1.21)
4. astigmatism=no contact-lenses=none 7 ==> tear-prod-rate=reduced 6    conf:(0.86) < lift:(1.71)> lev:(0.1) [2] conv:(1.75)
5. tear-prod-rate=reduced 12 ==> contact-lenses=none 12    conf:(1) < lift:(1.6)> lev:(0.19) [4] conv:(4.5)
6. contact-lenses=none 15 ==> tear-prod-rate=reduced 12    conf:(0.8) < lift:(1.6)> lev:(0.19) [4] conv:(1.88)
7. spectacle-prescrip=myope tear-prod-rate=reduced 6 ==> contact-lenses=none 6    conf:(1) < lift:(1.6)> lev:(0.09) [2] conv:(2.25)
8. contact-lenses=none 15 ==> spectacle-prescrip=myope tear-prod-rate=reduced 6    conf:(0.4) < lift:(1.6)> lev:(0.09) [2] conv:(1.13)
9. spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 ==> contact-lenses=none 6    conf:(1) < lift:(1.6)> lev:(0.09) [2] conv:(2.25)
10. contact-lenses=none 15 ==> spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6    conf:(0.4) < lift:(1.6)> lev:(0.09) [2] conv:(1.13)
```

Status

OK Log x 0

Program 10

AIM:

To perform prediction in Weka using Nearest Neighbour.

THEORY:

Nearest Neighbour (also known as Collaborative Filtering or Instance-based Learning) is a useful data mining technique that allows you to use your past data instances, with known output values, to predict an unknown output value of a new data instance. KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry.

Algorithm:

1. Load the data
2. Initialise the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points
 1. Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
 2. Sort the calculated distances in ascending order based on distance values
 3. Get top k rows from the sorted array
 4. Get the most frequent class of these rows
 5. Return the predicted class

Code:

```
public void classify(){
    super.confusionMatrix = new int[classPossibilities][classPossibilities];
    IBk nereastNeighbor = new IBk(neighbors);
    for (int i = 0; i < iterations; i++) {
        Instances dataTrain = dataSource.trainCV(partitions,i);
        Instances dataTest = dataSource.testCV(partitions, i);
        int realClass = 0;
        int resultClass = 0;
        try {
            nereastNeighbor.buildClassifier(dataTrain);
            for (int j = 0; j < dataTest.numInstances(); j++) {
                Instance example = dataTest.instance(j);
                realClass = (int)example.value(classIndex);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```

        example.setClassMissing();

        resultClass = (int) nereastNeighbor.classifyInstance(example);

        this.confusionMatrix[realClass][resultClass]++;

    }

    } catch(Exception e) {

        e.printStackTrace();

    }

}
}

```

PROCEDURE:

- Load the ARFF file into WEKA.
- We should next select the Classify tab. On this tab, we should select lazy, then select IBk (the IB stands for Instance-Based, and the k allows us to specify the number of neighbors to examine).
- We are ready to create our model in WEKA. Ensure that Use training set is selected so we use the data set we just loaded to create our model. Click Start and let WEKA run.

OUTPUT:

Training set

```

bmw-training.arff - Notepad
File Edit Format View Help
@relation bmwreponses

@attribute IncomeBracket {0,1,2,3,4,5,6,7}
@attribute FirstPurchase numeric
@attribute LastPurchase numeric
@attribute responded {1,0}

@data

4,200210,200601,0
5,200301,200601,1
6,200411,200601,0
5,199609,200603,0
6,200310,200512,1
4,200502,200601,1
5,200309,200602,1
6,200110,200603,0
6,199710,200603,1
5,200310,200601,1
5,199801,200512,0
1,200402,200511,0
4,200011,200601,1

```


Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

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

Filter: Choose **None** Apply Stop

Current relation
Relation: bmwreponses
Instances: 3000
Attributes: 4
Sum of weights: 3000

Attributes
All None Invert Pattern

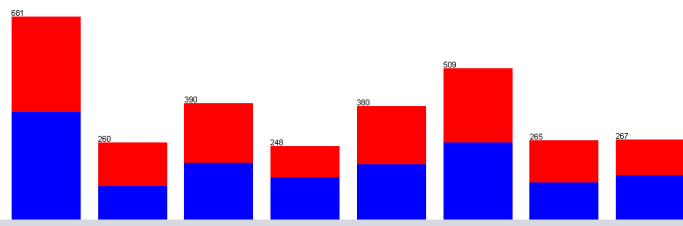
No.	Name
1	<input checked="" type="checkbox"/> IncomeBracket
2	<input type="checkbox"/> FirstPurchase
3	<input type="checkbox"/> LastPurchase
4	<input type="checkbox"/> responded

Remove

Selected attribute
Name: IncomeBracket
Missing: 0 (0%)
Distinct: 8
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	0	681	681.0
2	1	260	260.0
3	2	390	390.0
4	3	248	248.0
5	4	380	380.0
6	5	509	509.0
7	6	265	265.0
8	7	267	267.0

Class: responded (Nom) Visualize All



Status: OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier: Choose **IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch" -A "weka.core.EuclideanDistance" -R first-last"**

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

(Nom) responded
Start Stop

Result list (right-click for options)
23.47.37 - lazy IBk

Classifier output
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.42 seconds

=== Summary ===

Correctly Classified Instances	2663	88.7667 %
Incorrectly Classified Instances	337	11.2333 %
Kappa statistic	0.7748	
Mean absolute error	0.1326	
Root mean squared error	0.2573	
Relative absolute error	26.522 %	
Root relative squared error	51.462 %	
Total Number of Instances	3000	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Weighted Avg.	0.950	0.177	0.847	0.950	0.896	0.781	0.972	0.968	1
	0.823	0.050	0.941	0.823	0.878	0.781	0.972	0.967	0

=== Confusion Matrix ===

a	b	<-- classified as	
1449	76	a = 1	
261	1214	b = 0	

Status: OK Log x 0

Test set

bmw-test.arff - Notepad

File Edit Format View Help

@relation bmwreponses

@attribute IncomeBracket {0,1,2,3,4,5,6,7}

@attribute FirstPurchase numeric

@attribute LastPurchase numeric

@attribute responded {1,0}

@data

3,200102,200601,1

2,200410,200602,1

0,200309,200506,0

4,200406,200511,0

0,200201,200510,1

6,200506,200506,1

4,200506,200602,1

5,200103,200510,0

7,199802,200512,0

2,199704,200603,0

7,199803,200505,1

0,200203,200702,0

1,199803,200512,0

0,200502,200601,0

2,200310,200602,1

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch" -A "weka.core.EuclideanDistance" -R first-last"

Test options

☐ Use training set

☒ Supplied test set

☐ Cross-validation

☐ Percentage split

Folds 10

% 66

More options...

(Nom) responded

Start

Stop

Result list (right-click for options)

23:47:37 - lazyIBk

23:50:36 - lazyIBk

Classifier output

using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.14 seconds

=== Summary ===

Correctly Classified Instances	850	56.6667 %
Incorrectly Classified Instances	650	43.3333 %
Kappa statistic	0.1355	
Mean absolute error	0.4353	
Root mean squared error	0.6211	
Relative absolute error	87.8305 %	
Root relative squared error	124.1602 %	
Total Number of Instances	1500	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.633	0.497	0.550	0.633	0.589	0.137	0.583	0.547	1
	0.503	0.367	0.588	0.503	0.542	0.137	0.583	0.577	0
Weighted Avg.	0.567	0.431	0.569	0.567	0.565	0.137	0.583	0.562	

=== Confusion Matrix ===

a	b	<-- classified as	
465	270	a = 1	
380	385	b = 0	

Status

OK

Log

x 0