

Dr. Akhilesh Das Gupta Institute of Professional Studies



Data Warehousing & Data Mining Lab File

CIE-425P

Submitted

in

Department of Computer Science & Engineering

SUBMITTED TO

Mr Faisal Rais

Assistant Professor

Department of CSE,ADGIPS

SUBMITTED BY

Name

roll no

section

INDEX

S.No	AIM OF EXPERIMENT	Page No	Signature
1	Introduction to weka		
2	Study of ETL process and its tools.		
3	To create an arff file.		
4	Implementation of Classification technique on ARFF files using WEKA.		
5	Implementation of Clustering technique on ARFF files using WEKA.		
6	Implementation of Association Rule technique on ARFF files using WEKA.		
7	Implementation of Visualization technique on ARFF files using WEKA		
8	To use numeric transform filter and floor function to obtain the precision up to same value.		
9	Implementation of Apriori Algorithm		
10	Implementation of K-means algorithm		

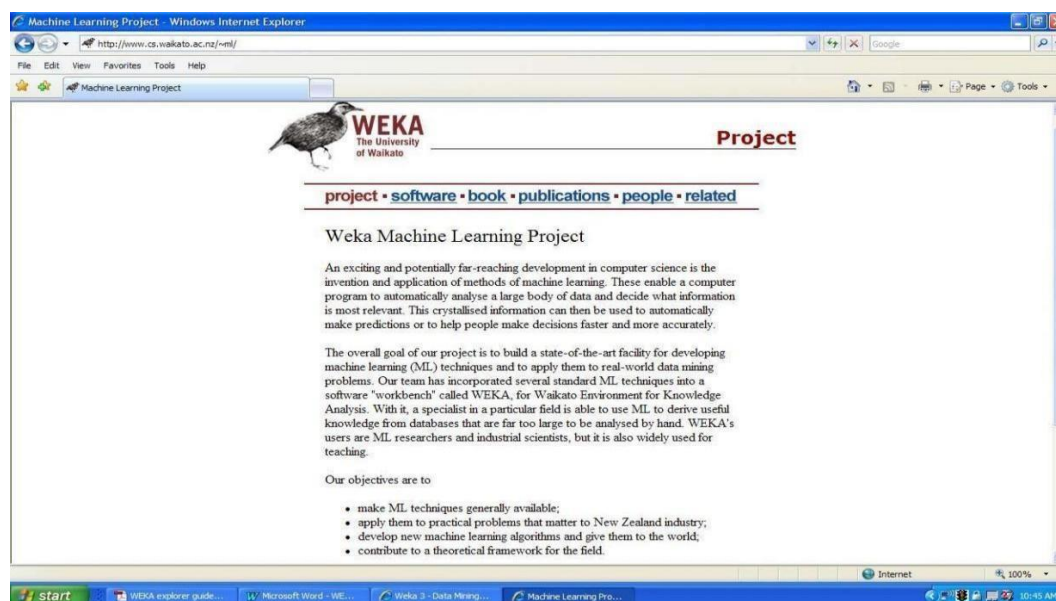
Experiment 1

Aim: Introduction to weka

WEKA, formally called Waikato Environment for Knowledge Learning, is a computer program that was developed at the University of Waikato in New Zealand for the purpose of identifying information from raw data gathered from agricultural domains. WEKA supports many different standard data mining tasks such as data preprocessing, classification, clustering, regression, visualization and feature selection. The basic premise of the application is to utilize a computer application that can be trained to perform machine learning capabilities and derive useful information in the form of trends and patterns. WEKA is an open source application that is freely available under the GNU general public license agreement. Originally written in C the WEKA application has been completely rewritten in Java and is compatible with almost every computing platform. It is user friendly with a graphical interface that allows for quick set up and operation. WEKA operates on the predication that the user data is available as a flat file or relation, this means that each data object is described by a fixed number of attributes that usually are of a specific type, normal alpha-numeric or numeric values. The WEKA application allows novice users a tool to identify hidden information from database and file systems with simple to use options and visual interfaces.

Installation

The program information can be found by conducting a search on the Web for WEKA Data Mining or going directly to the site at www.cs.waikato.ac.nz/~ml/WEKA/. The site has a very large amount of useful information on the program's benefits and background. New users might find some benefit from investigating the user manual for the program. The main WEKA site has links to this information as well as past experiments for new users to refine the potential uses that might be of particular interest to them. When prepared to download the software it is best to select the latest application from the selection offered on the site. The format for downloading the application is offered in a self installation package and is a simple procedure that provides the complete program on the end users machine that is ready to use when extracted.



Opening the program

Once the program has been loaded on the user's machine it is opened by navigating to the programs start option and that will depend on the user's operating system. Figure 1 is an example of the initial opening screen on a computer with Windows XP.



Figure 1 Chooser screen

There are four options available on this initial screen.

- Simple CLI- provides users without a graphic interface option the ability to execute commands from a terminal window.
- Explorer- the graphical interface used to conduct experimentation on raw data
- Experimenter- this option allows users to conduct different experimental variations on data sets and perform statistical manipulation
- Knowledge Flow-basically the same functionality as Explorer with drag and drop functionality. The advantage of this option is that it supports incremental learning from previous results

While the options available can be useful for different applications the remaining focus of the user manual will be on the Experimenter option through the rest of the user guide.

After selecting the Experimenter option the program starts and provides the user with a separate graphical interface.

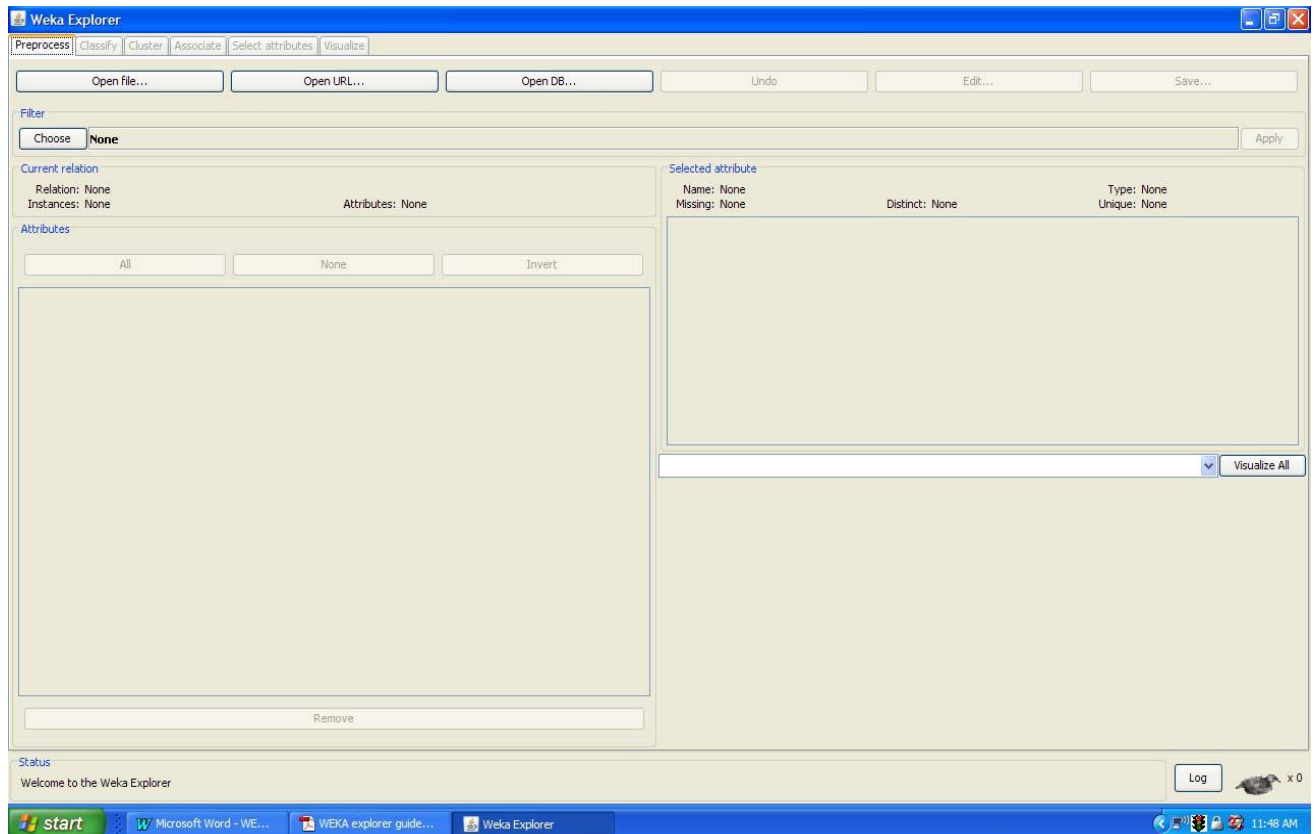


Figure 2

Figure 2 shows the opening screen with the available options. At first there is only the option to select the Preprocess tab in the top left corner. This is due to the necessity to present the data set to the application so it can be manipulated. After the data has been preprocessed the other tabs become active for use.

There are six tabs:

Preprocess- used to choose the data file to be used by the application

Classify- used to test and train different learning schemes on the preprocessed data file under experimentation.

Cluster- used to apply different tools that identify clusters within the data file

Association- used to apply different rules to the data file that identify association within the data.

Select attributes used to apply different rules to reveal changes based on selected attributes inclusion or exclusion from the experiment

Visualize- used to see what the various manipulation produced on the data set in a 2D format, in scatter plot and bar graph output

Once the initial preprocessing of the data set has been completed the user can move between the tab options to perform changes to the experiment and view the results in real time. This provides the benefit of having the ability to move from one option to the next so that when a condition becomes exposed it can be placed in a different environment to be visually changed instantaneously.

Preprocessing

In order to experiment with the application the data set needs to be presented to WEKA in a format that the program understands. There are rules for the type of data that WEKA will accept. There are three options for presenting data into the program.

- Open File- allows for the user to select files residing on the local machine or recorded medium
- Open URL- provides a mechanism to locate a file or data source from a different location specified by the user
- Open Database- allows the user to retrieve files or data from a database source provided by the user

There are restrictions on the type of data that can be accepted into the program. Originally the software was designed to import only ARFF files, newer versions allow different file types such as CSV, C4.5 and serialized instance formats. The extensions for these files include .csv, .arff, .names, .bsi and .data.

Figure 3 shows an example of selection of the file weather.arff.

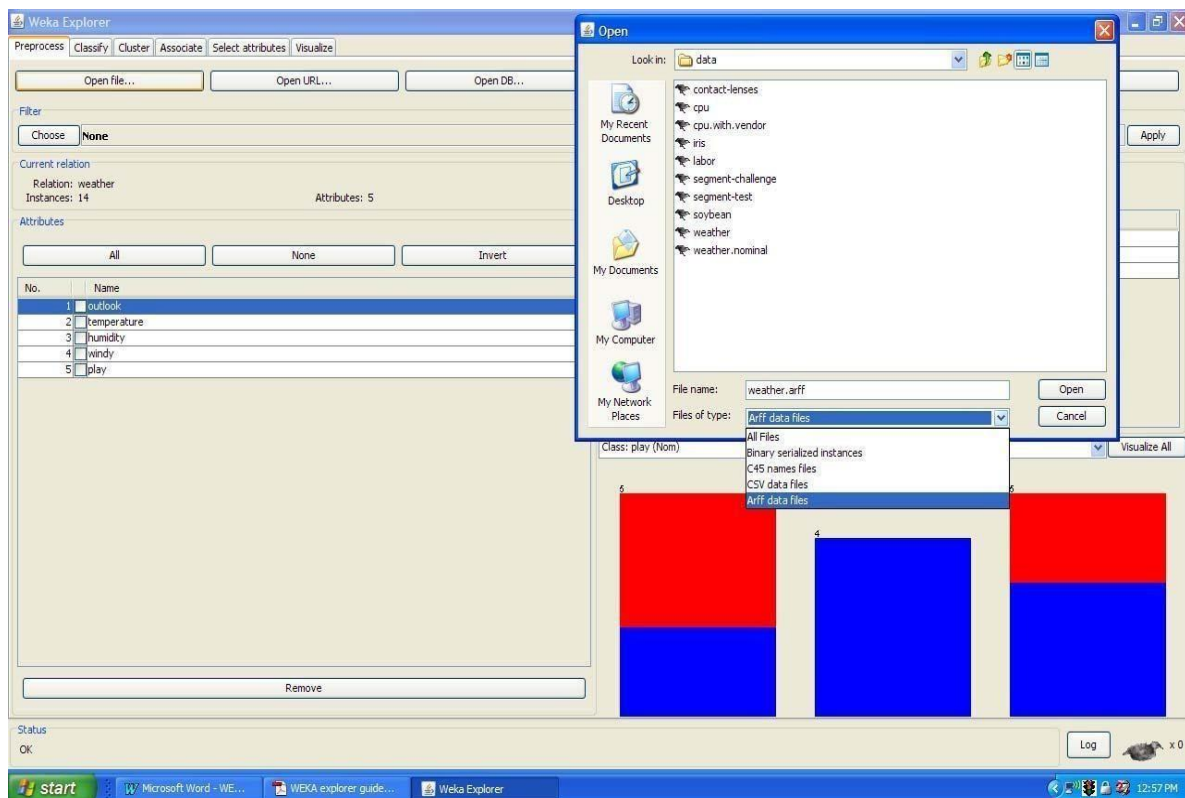


Figure 3

Once the initial data has been selected and loaded the user can select options for refining the experimental data. The options in the preprocess window include selection of optional filters to apply and the user can select or remove different attributes of the data set as necessary to identify specific information. The ability to pick from the available attributes allows users to separate different parts of the data set for clarity in the experimentation. The user can modify the attribute selection and change the relationship among the different attributes by deselecting different choices from the original data set. There are many different filtering options available within the preprocessing window and the user can select the different options based on need and type of data present.

Classify

The user has the option of applying many different algorithms to the data set that would in theory produce a representation of the information used to make observation easier. It is difficult to identify which of the options would provide the best output for the experiment. The best approach is to independently apply a mixture of the available choices and see what yields something close to the desired results. The Classify tab is where the user selects the classifier choices. Figure 4 shows some of the categories.

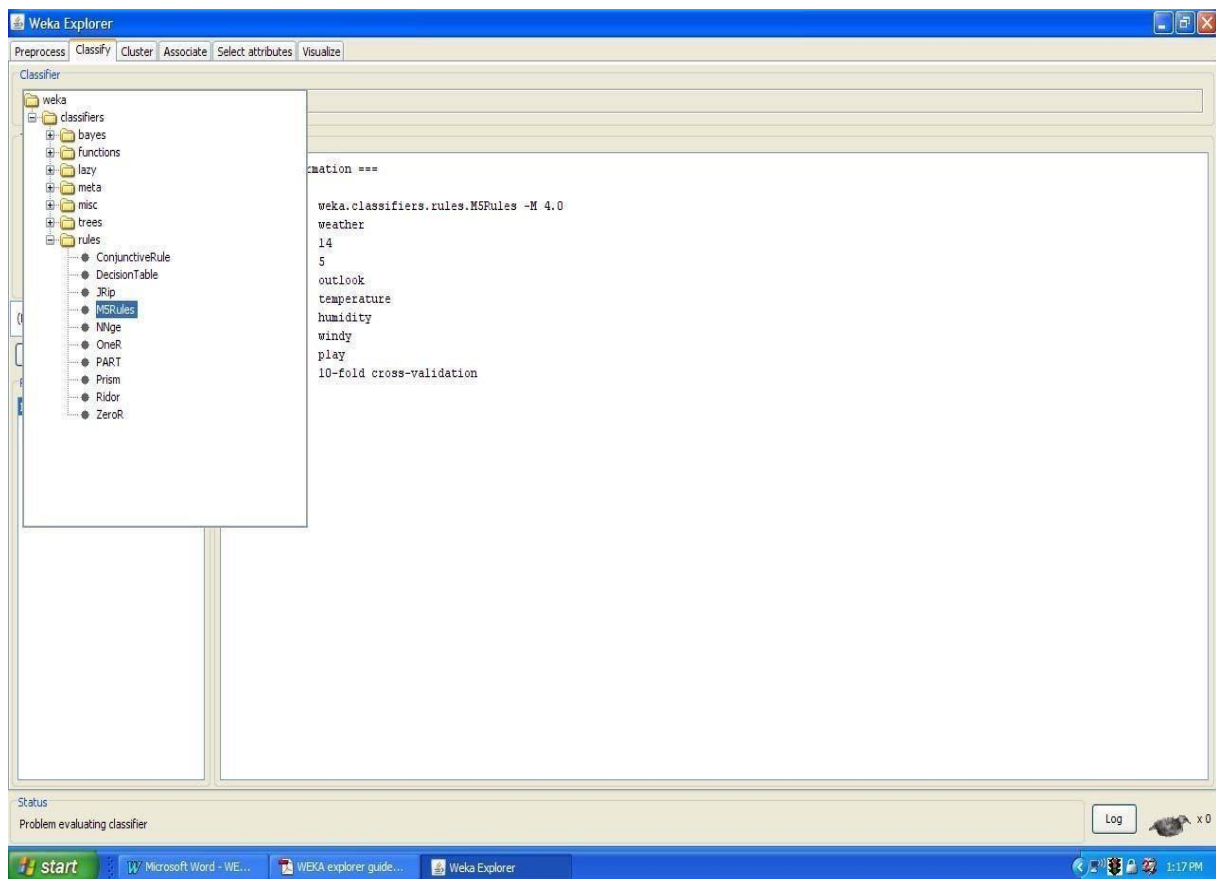


Figure 4

Again there are several options to be selected inside of the classify tab. Test option gives the user the choice of using four different test mode scenarios on the data set:

1. Use training set
2. Supplied training set
3. Cross validation
4. Split percentage

There is the option of applying any or all of the modes to produce results that can be compared by the user. Additionally inside the test options toolbox there is a dropdown menu so the user can select various items to apply that depending on the choice can provide output options such as saving the results to file or specifying the random seed value to be applied for the classification.

The classifiers in WEKA have been developed to train the data set to produce output that has been classified based on the characteristics of the last attribute in the data set. For a specific attribute to be used the option must be selected by the user in the options menu before testing is performed. Finally the results have been calculated and they are shown in the text box on the lower right. They can be saved in a file and later retrieved for comparison

at a later time or viewed within the window after changes and different results have been derived.

Cluster

The Cluster tab opens the process that is used to identify commonalties or clusters of occurrences within the data set and produce information for the user to analyze. There are a few options within the cluster window that are similar to those described in the classifier tab. They are use training set, supplied test set, percentage split. The fourth option is classes to cluster evaluation, which compares how well the data compares with a pre-assigned class within the data. While in cluster mode users have the option of ignoring some of the attributes from the data set. This can be useful if there are specific attributes causing the results to be out of range or for large data sets. Figure 5 shows the Cluster window and some of its options.

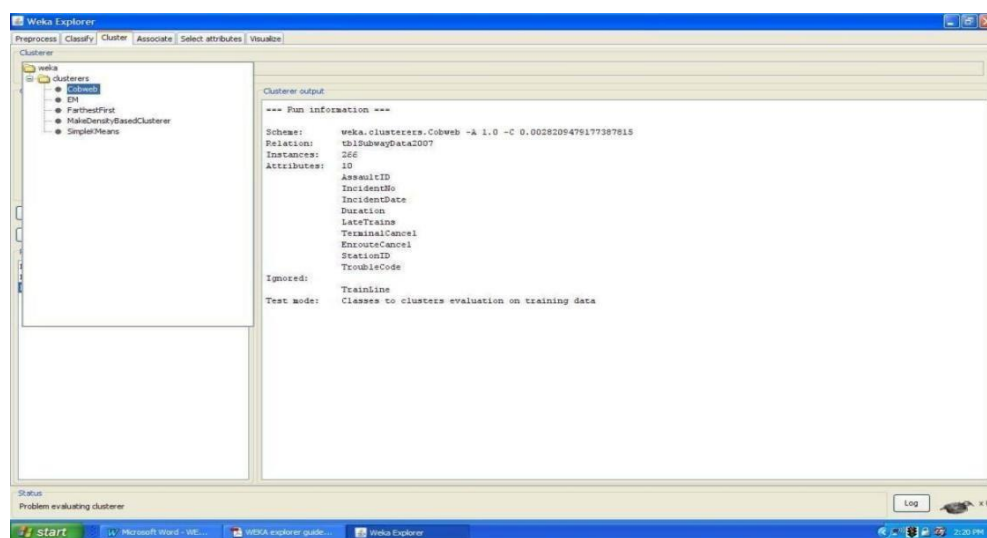


Figure 5

Associate

The associate tab opens a window to select the options for associations within the data set. The user selects one of the choices and presses start to yield the results. There are few options for this window and they are shown in Figure 6 below.

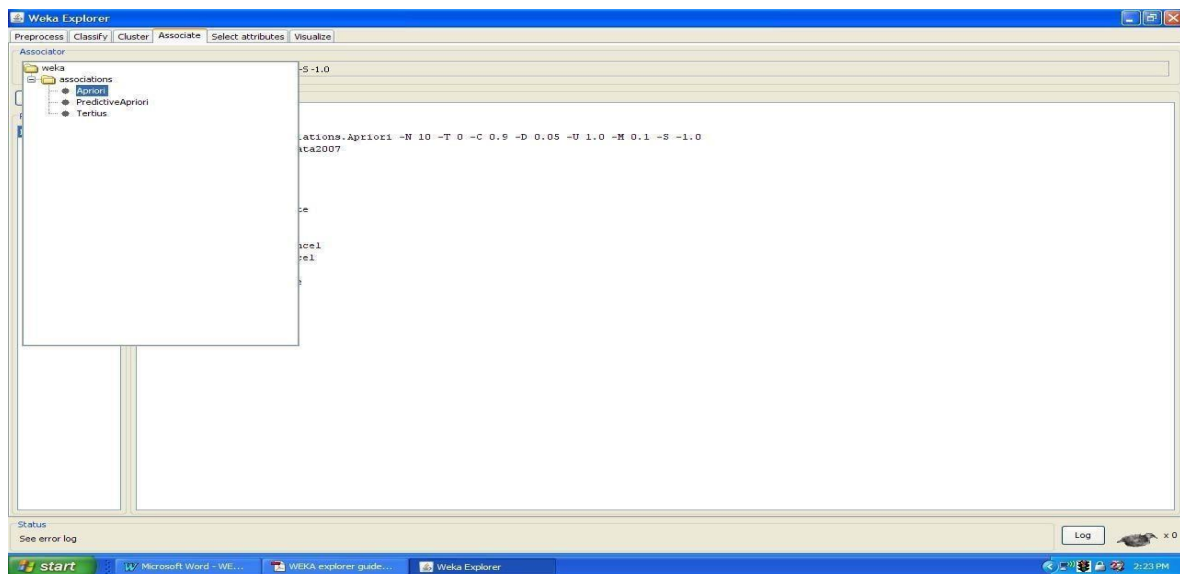


Figure 6

Select Attributes

The next tab is used to select the specific attributes used for the calculation process. By default all of the available attributes are used in the evaluation of the data set. If the user wanted to exclude certain categories of the data they would deselect those specific choices from the list in the cluster window. This is useful if some of the attributes are of a different form such as alphanumeric data that could alter the results. The software searches through the selected attributes to decide which of them will best fit the desired calculation. To perform this, the user has to select two options, an attribute evaluator and a search method. Once this is done the program evaluates the data based on the sub set of the attributes then performs the necessary search for commonality with the data. Figure 7 shows the opinions of attribute evaluation. Figure 8 shows the options for the search method.

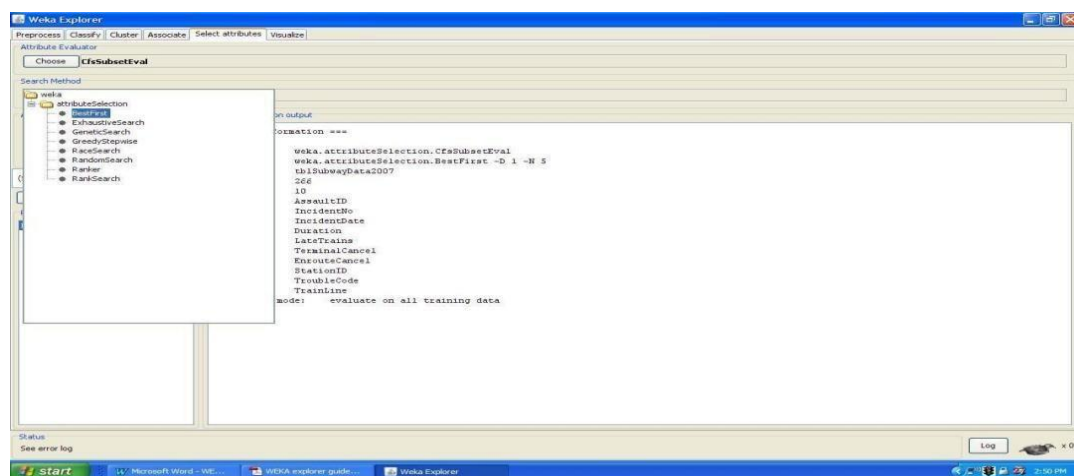


Figure-7

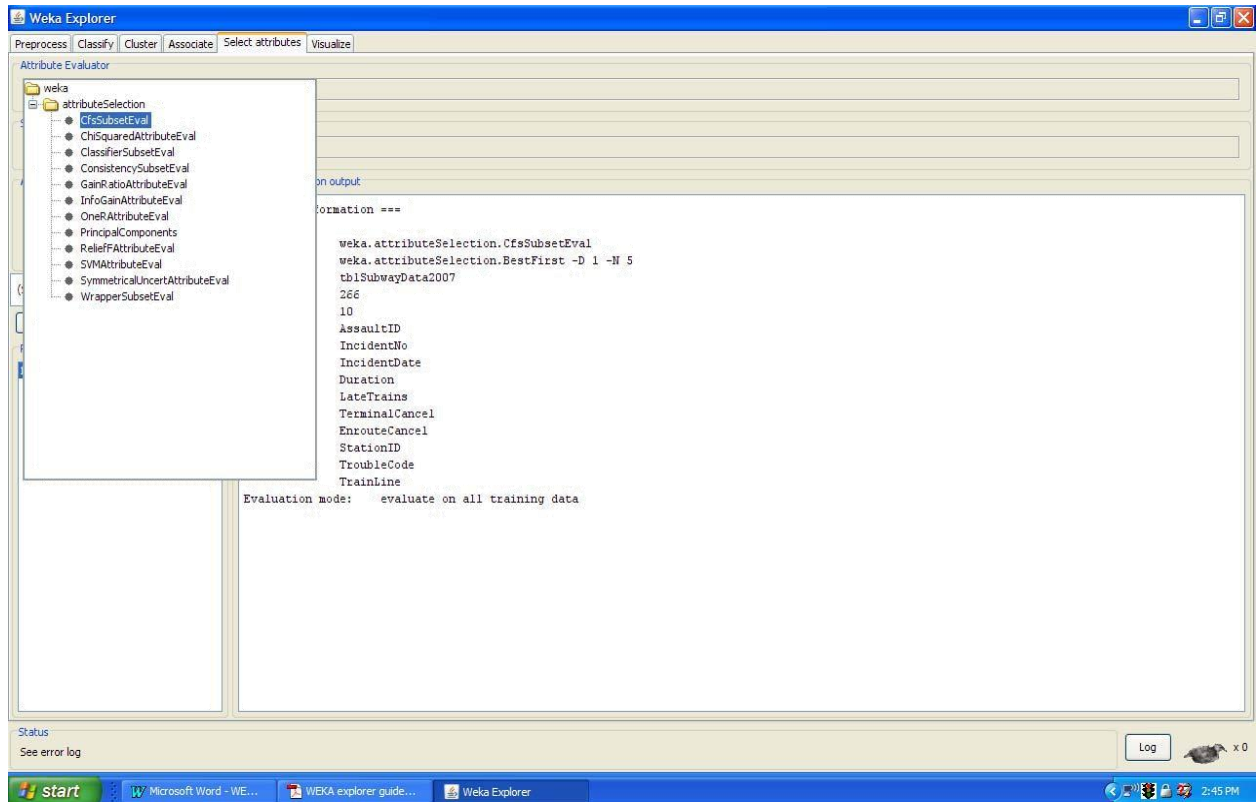


Figure-8

Visualization

The last tab in the window is the visualization tab. Within the program calculations and comparisons have occurred on the data set. Selections of attributes and methods of manipulation have been chosen. The final piece of the puzzle is looking at the information that has been derived throughout the process. The user can now actually see the fruit of their efforts in a two dimensional representation of the information. The first screen that the user sees when they select the visualization option is a matrix of plots representing the different attributes within the data set plotted against the other attributes. If necessary there is a scroll bar to view all of the produced plots. The user can select a specific plot from the matrix to view its contents for analyzation. A grid pattern of the plots allows the user to select the attribute positioning to their liking and for better understanding. Once a specific plot has been selected the user can change the attributes from one view to another providing flexibility. Figure 9 shows the plot matrix view.

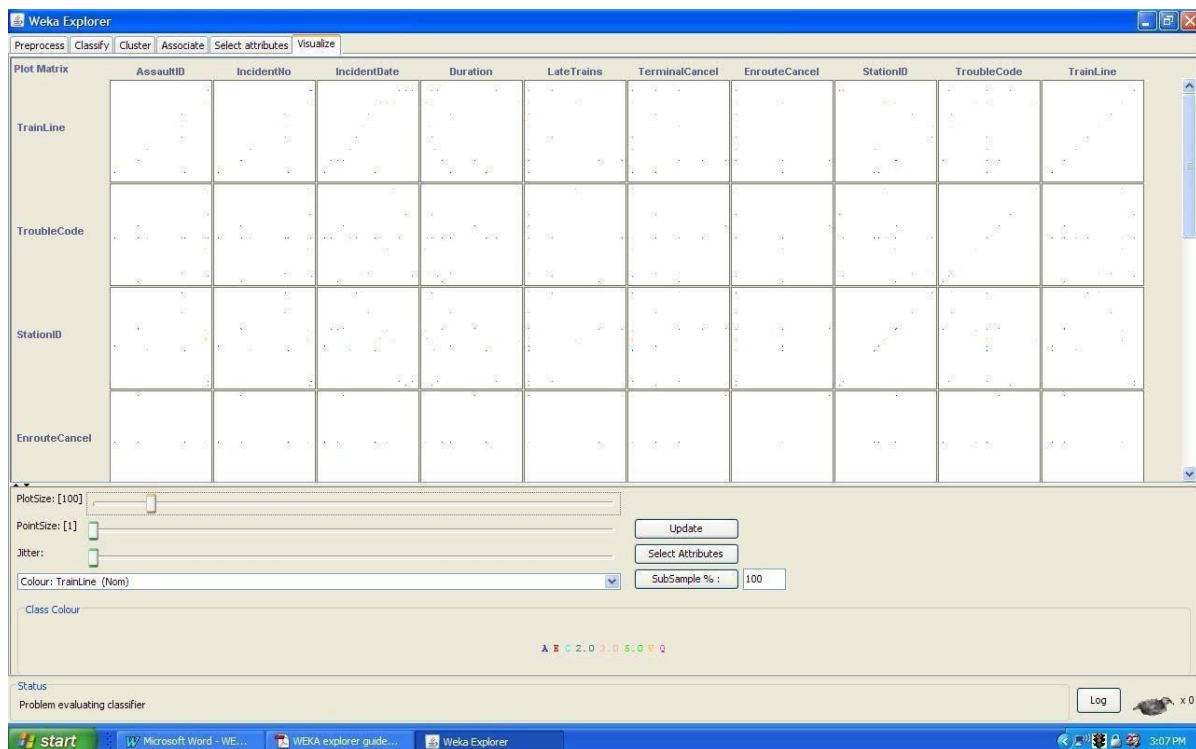


Figure 9

The scatter plot matrix gives the user a visual representation of the manipulated data sets for selection and analysis. The choices are the attributes across the top and the same from top to bottom giving the user easy access to pick the area of interest. Clicking on a plot brings up a separate window of the selected scatter plot. The user can then look at a visualization of the data of the attributes selected and select areas of the scatter plot with a selection window or by clicking on the points within the plot to identify the point's specific information. Figure 10 shows the scatter plot for two attributes and the points derived from the data set. There are a few options to view the plot that could be helpful to the user. It is formatted similar to an X/Y graph yet it can show any of the attribute classes that appear on the main scatter plot matrix. This is handy when the scale of the attribute is unable to be ascertained in one axis over the other. Within the plot the points can be adjusted by utilizing a feature called jitter. This option moves the individual points so that in the event of close data points users can reveal hidden multiple occurrences within the initial plot. Figure 11 shows an example of this point selection and the results the user sees.

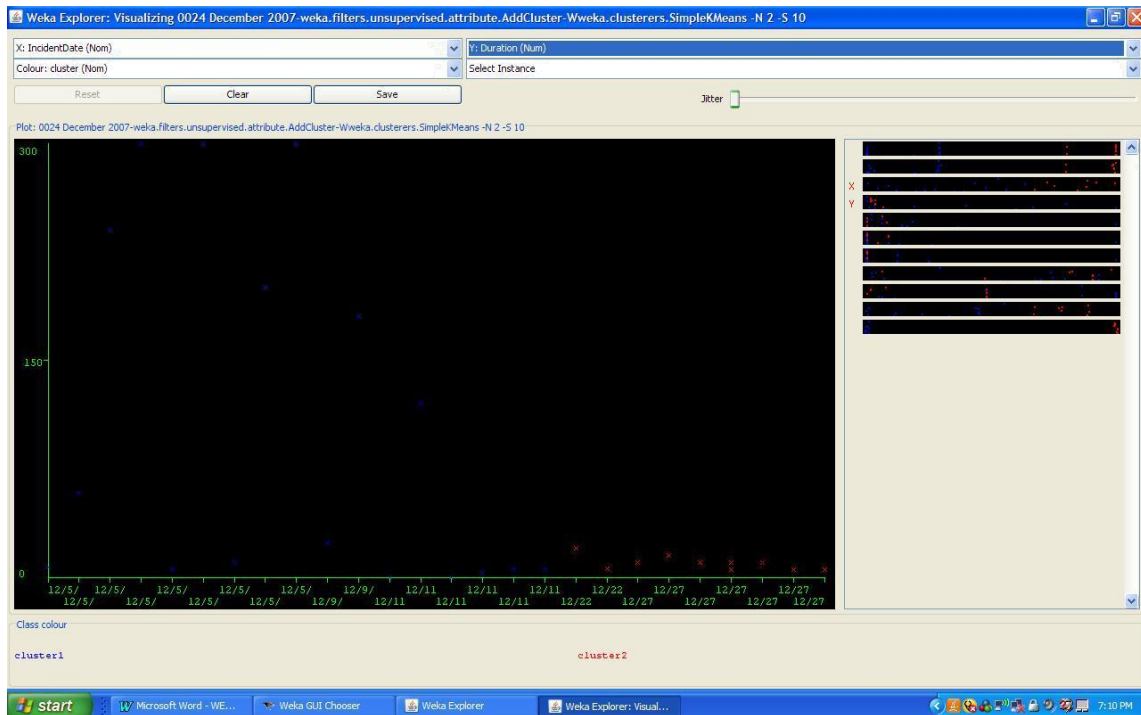


Figure 10

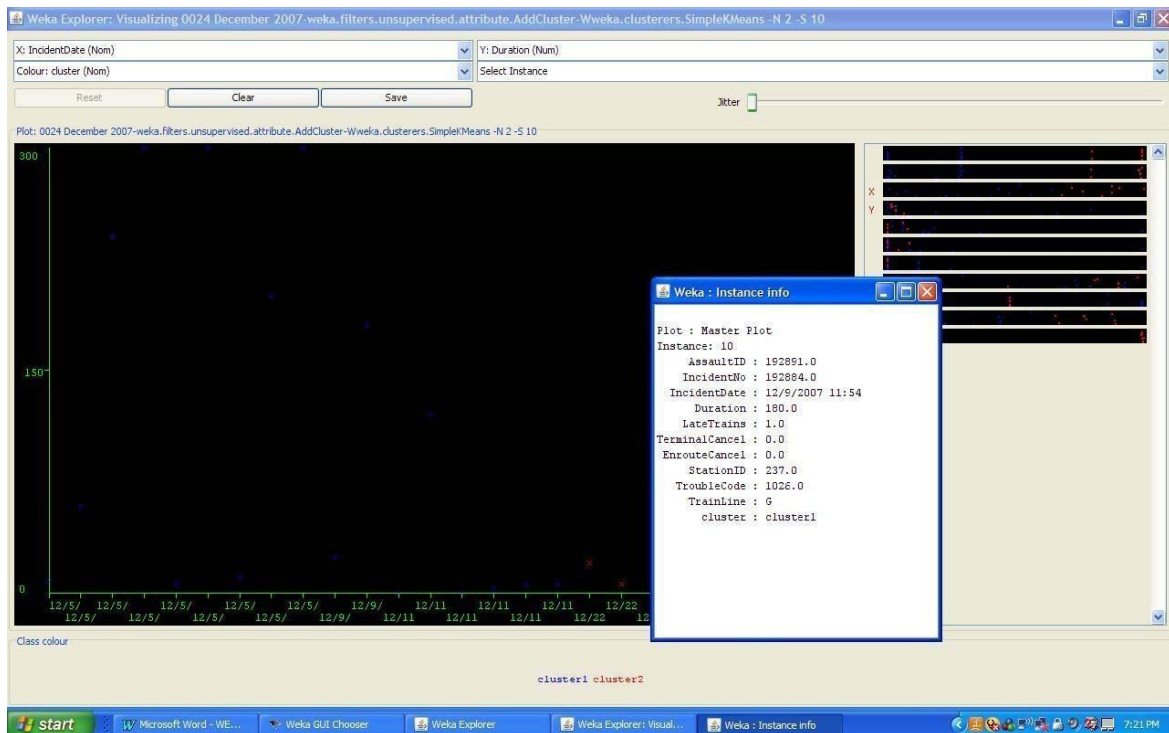


Figure 11

There are a few options to manipulate the view for the identification of subsets or to separate the data points on the plot.

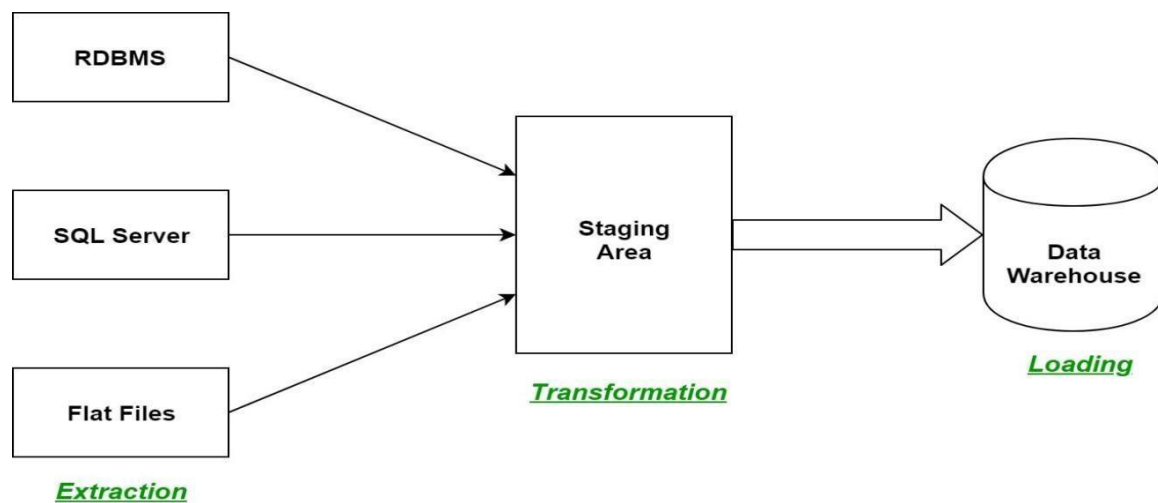
- Polyline---can be used to segment different values for additional visualization clarity on the plot. This is useful when there are many data points represented on the graph.
- Rectangle--this tool is helpful to select instances within the graph for copying or clarification.

Polygon—Users can connect points to segregate information and isolate points for reference.

EXPERIMENT-2

Aim : Study of ETL process and its tools.

ETL is a process in Data Warehousing and it stands for Extract, Transform and Load. It is a process in which an ETL tool extracts the data from various data source systems, transforms it in the staging area, and then finally, loads it into the Data Warehouse system.



1. Data Extraction:

The first step of the ETL process is extraction. In this step, data from various source systems is extracted which can be in various formats like relational databases, No SQL, XML, and flat files into the staging area. It is important to extract the data from various source systems and store it into the staging area first and not directly into the data warehouse because the extracted data is in various formats and can be corrupted also. Hence loading it directly into the data warehouse may damage it and rollback will be much more difficult. Therefore, this is one of the most important steps of ETL process.

2. Data Transformation:

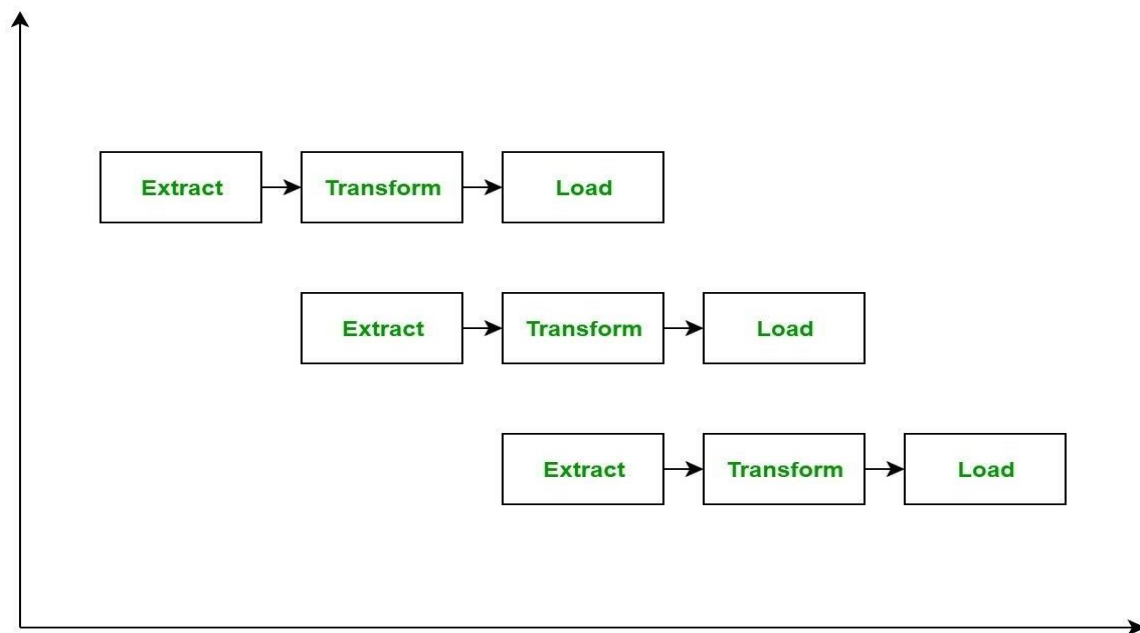
The second step of the ETL process is transformation. In this step, a set of rules or functions are applied on the extracted data to convert it into a single standard format. It may involve following processes/tasks:

- ◆ **Filtering** – loading only certain attributes into the data warehouse.
- ◆ **Cleaning** – filling up the NULL values with some default values, mapping U.S.A, United States, and America into USA, etc.
- ◆ **Joining** – joining multiple attributes into one.
- ◆ **Splitting** – splitting a single attribute into multiple attributes.
- ◆ **Sorting** – sorting tuples on the basis of some attribute (generally key-attribute).

3. Data Loading:

The third and final step of the ETL process is loading. In this step, the transformed data is finally loaded into the data warehouse. Sometimes the data is updated by loading into the data warehouse very frequently and sometimes it is done after longer but regular intervals. The rate and period of loading solely depends on the requirements and varies from system to system.

ETL process can also use the pipelining concept i.e. as soon as some data is extracted, it can be transformed and during that period some new data can be extracted. And while the transformed data is being loaded into the data warehouse, the already extracted data can be transformed. The block diagram of the pipelining of ETL process is shown below:



ETL Tools:

Extraction, transformation, and load help the organization to make the data accessible, meaningful, and usable across different data systems. An ETL tool is a software used to extract, transform, and loading the data. An ETL tool is a set of libraries written in any programming language which will simplify our work to make data integration and transformation operation for any need.

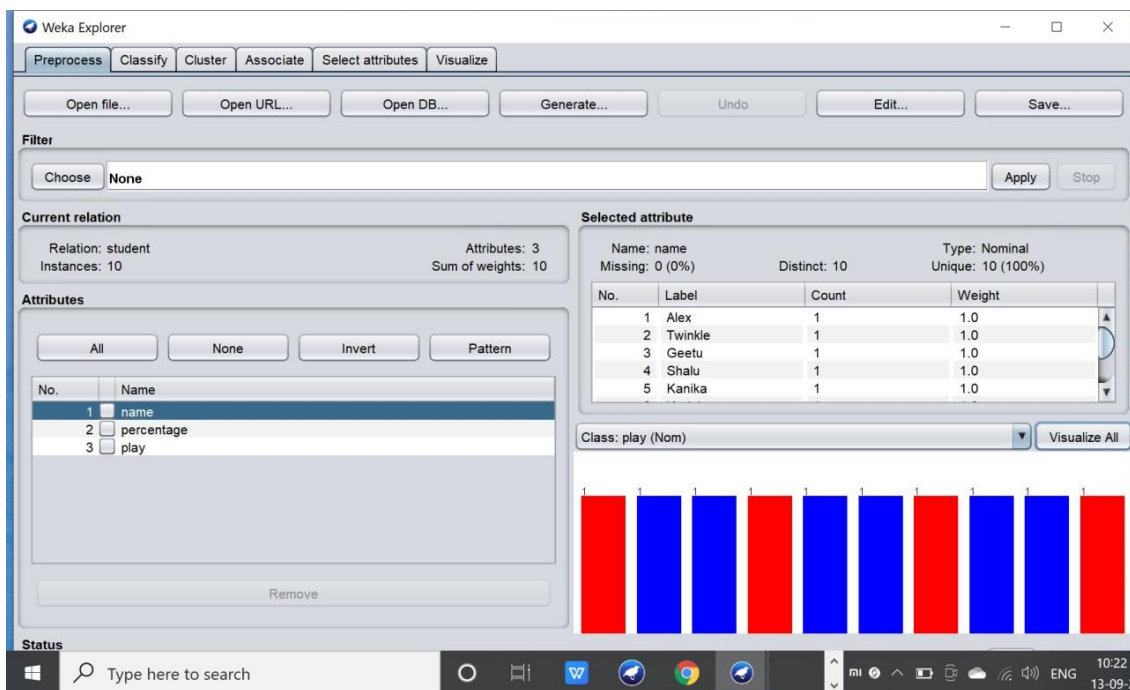
Most commonly used ETL tools are Sybase, Oracle Warehouse builder, CloverETL, and MarkLogic.

EXPERIMENT-3

AIM:- To create an arff file.

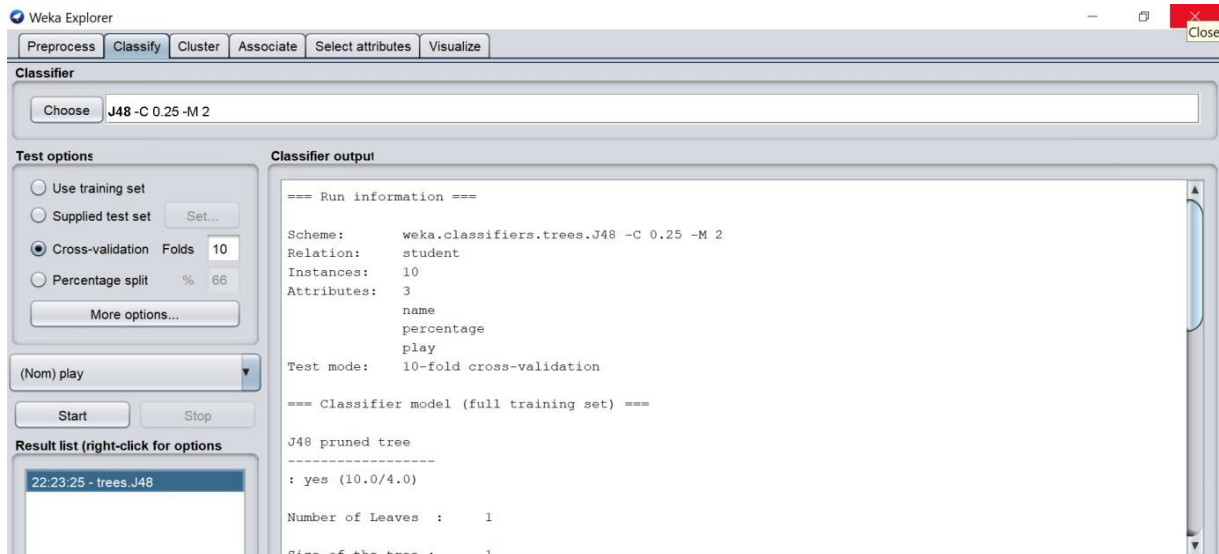
To create an arff file the steps involved are:-

1. Open a notepad file. Make a relation student, provide the attributes n data to the relation.
2. Save the file with .arff extension. Open WEKA. Open file from the open file option.



Go to the **Classify** menu.

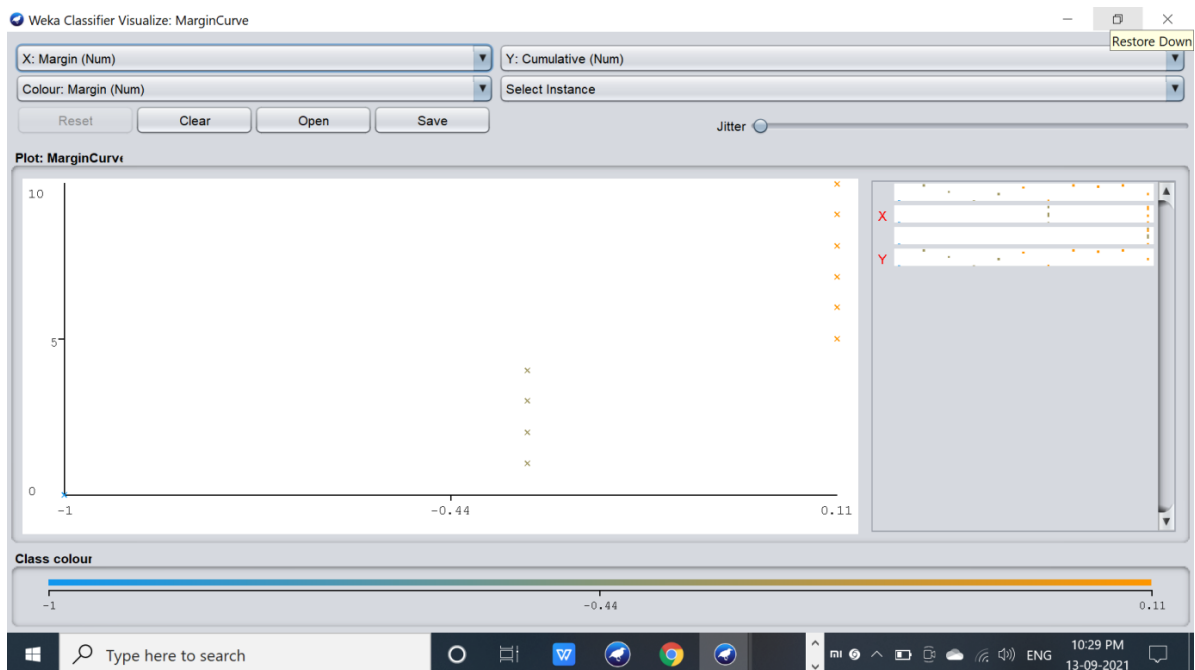
3. Select a **classifier**. At the top of the classify section is the classifier box. This box has a text field that gives the name of the currently selected **classifier/algorithm**, and its option.
4. Select a classifier by clicking on the choose option. The **choose** button allows you to choose one of the classifiers that are available in WEKA.
5. Choose any classifier as per your need and click on the close button for instance, choose the **J48** classifier for your file and press the **Start** button.



Click **Start**

The result is obtained in the **Classifier output window**.

6. Now, right click in the **Result list window** on the text selected and select the type of graph you want to evaluate.



We have successfully created an arff file and obtained the output.

=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: student

Instances: 10

Attributes: 3

name

percentage

play

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree

: yes (10.0/4.0)

Number of Leaves : 1

Size of the tree : 1

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	6	60	%
Incorrectly Classified Instances	4	40	%
Kappa statistic	0		
Mean absolute error	0.5333		
Root mean squared error	0.5443		
Relative absolute error	101.1494	%	
Root relative squared error	101.793	%	
Total Number of Instances	10		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	1.000	0.600	1.000	0.750	?	0.000	0.600	yes
	0.000	0.000	?	0.000	?	?	0.000	0.400	no
Weighted Avg.	0.600	0.600	?	0.600	?	?	0.000	0.520	

=== Confusion Matrix ===

a b <-- classified as

6 0 | a = yes

4 0 | b = no

The contents of the notepad file are:-

@relation student

@attribute

name{Alex,Twinkle,Geetu,Shalu,Kanika,Karishma,Ayush,Akshay,Sarvesh,Mahesh}

@attribute percentage real

@attribute play{yes,no}

@data

Alex,88,no

Twinkle,90,yes

Geetu,85,yes

Shalu,84,no

Kanika,75,yes

Karishma,80,yes

Ayush,83,no

Akshay,74,yes

Sarvesh,88,yes

Mahesh,78,no

Experiment 4

Aim : Implementation of Classification technique on ARFF files using WEKA.

This experiment illustrates the use of j48 classifier in weka. The sample data set used in this experiment is “student” data available at .arff format. This document assumes that appropriate data pre-processing has been performed.

Steps involved in this experiment:

Step-1: We begin the experiment by loading the data (exp-4.arff) into weka.

Step2: Next we select the “classify” tab and click “choose” button to select the

“j48” classifier

Step3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the choose button. In this example, we accept the default values. The default version does perform some pruning but does not perform error pruning.

Step4: Under the “text” options in the main panel. We select the 10-fold cross validation as our evaluation approach. Since we don’t have separate evaluation data set, this is necessary to get a reasonable idea of accuracy of generated model.

Step-5: We now click “start” to generate the model. The Ascii version of the tree as well as evaluation statistic will appear in the right panel when the model construction is complete.

Step-6: Note that the classification accuracy of model is about 69%.this indicates that we may find more work. (Either in preprocessing or in selecting current parameters for the classification)

Step-7: Now weka also lets us a view a graphical version of the classification tree. This can be done by right clicking the last result set and selecting “visualize tree” from the pop-up menu.

Step-8: We will use our model to classify the new instances.

Step-9: In the main panel under “text” options click the “supplied test set” radio button and then click the “set” button. This will pop-up a window which will allow you to open the file containing test instances.

=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: student

Instances: 14

Attributes: 5

age

income

student

credit-rating

buyspc

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree

age = <30: no (5.0/1.0)

age = 30-40: yes (4.0)

age = >40

| credit-rating = fair: yes (3.0)

| credit-rating = excellent: no (2.0)

Number of Leaves : 4

Size of the tree : 6

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	11	78.5714 %
Incorrectly Classified Instances	3	21.4286 %
Kappa statistic	0.5532	
Mean absolute error	0.25	
Root mean squared error	0.4058	
Relative absolute error	49.5283 %	
Root relative squared error	79.6745 %	

Total Number of Instances 14

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

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC
Area Class								
	0.875	0.333	0.778	0.875	0.824	0.559	0.854	yes
	0.667	0.125	0.800	0.667	0.727	0.559	0.854	no
Weighted Avg.	0.786	0.244	0.787	0.786	0.782	0.559	0.854	0.837

==== Confusion Matrix ====

a b <-- classified as

7 1 | a = yes

2 4 | b = no

Dataset exp-4.arff

@relation student

@attribute age {<30,30-40,>40}

@attribute income {low, medium, high}

@attribute student {yes, no}

@attribute credit-rating {fair, excellent}

@attribute buyspc {yes, no}

@data

%

<30, high, no, fair, no

<30, high, no, excellent, no

30-40, high, no, fair, yes

>40, medium, no, fair, yes

>40, low, yes, fair, yes

>40, low, yes, excellent, no

30-40, low, yes, excellent, yes

<30, medium, no, fair, no

<30, low, yes, fair, no

>40, medium, yes, fair, yes

<30, medium, yes, excellent, yes

30-40, medium, no, excellent, yes

30-40, high, yes, fair, yes

>40, medium, no, excellent, no

%

The following screenshot shows the classification rules that were generated when j48 algorithm is applied on the given dataset.

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: student Instances: 14 Attributes: 5 Sum of weights: 14

Attributes: All None Invert Pattern

No.	Name
1	age
2	income
3	student
4	credit-rating
5	buyspc

Remove

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

No.	Label	Count	Weight
1	<30	5	5.0
2	30-40	4	4.0
3	>40	5	5.0

Class: buyspc (Nom) Visualize All

Status: OK Log x 0

Weka Explorer

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

Classifier: Choose J48 -C 0.25 -M 2

Test options:

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

 More options...

(Nom) buyspc Start Stop

Result list (right-click for options): 13:56:22 - trees.J48

Classifier output:

```

age = 30-40: yes (4.0)
age = >40
| credit-rating = fair: yes (3.0)
| credit-rating = excellent: no (2.0)

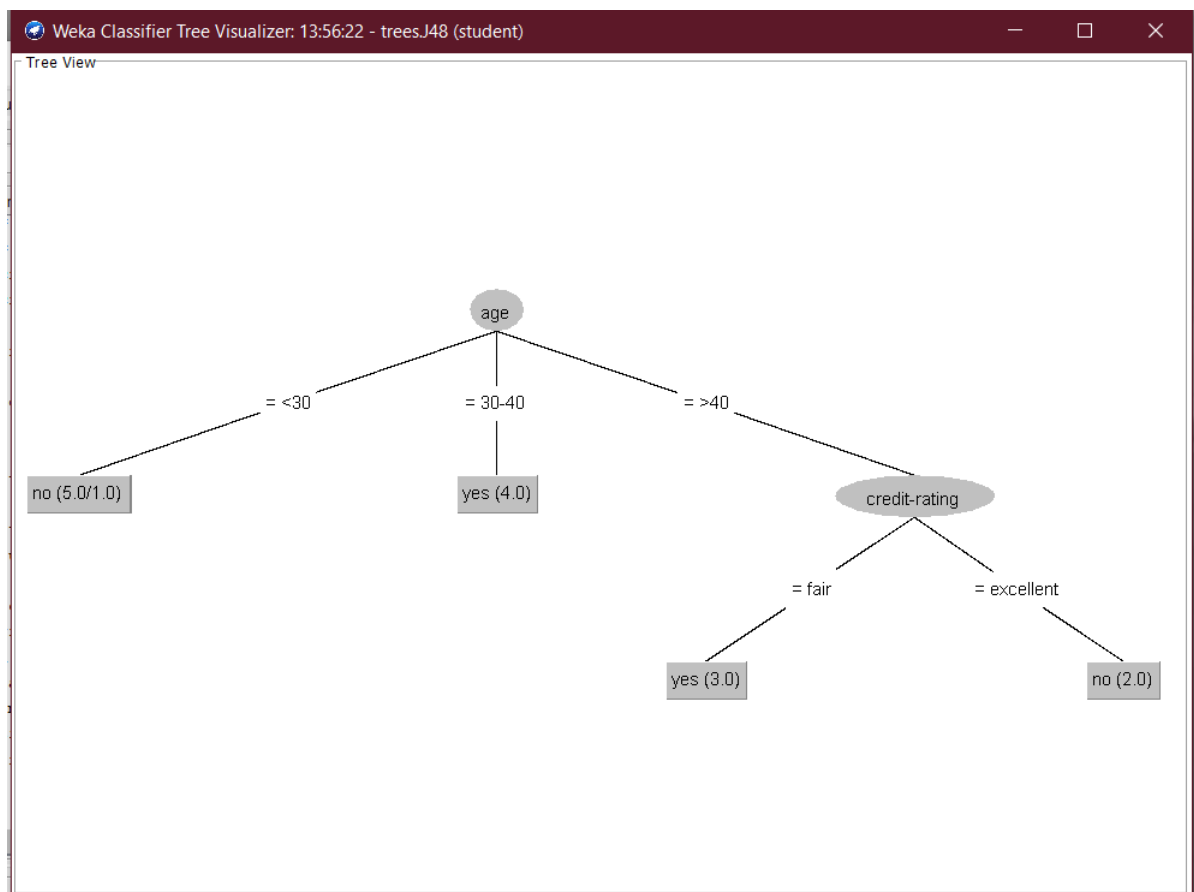
Number of Leaves : 4
Size of the tree : 6

Time taken to build model: 0.01 seconds

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

Correctly Classified Instances      11      78.5714 %
Incorrectly Classified Instances    3      21.4286 %
Kappa statistic                    0.5532
Mean absolute error                 0.25
Root mean squared error             0.4058
Relative absolute error             49.5283 %
Root relative squared error         79.6745 %
Total Number of Instances          14
  
```

Status: OK Log x 0



Experiment 5

Aim : Implementation of Clustering technique on ARFF files using WEKA.

This experiment illustrates the use of simple k-mean clustering with Weka explorer. The sample data set used for this example is based on the iris data available in ARFF format. This document assumes that appropriate preprocessing has been performed. This iris dataset includes 150 instances.

Steps involved in this Experiment

Step 1: Run the Weka explorer and load the data file iris.arff in preprocessing interface.

Step 2: 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.

Step 3 : In this case we select 'simple k-means'.

Step 4: Next click on the text button to the right of the choose button to get the 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.

Step 5 : Once the options have been specified. We run the clustering algorithm there and we must make sure that they are in the 'cluster mode' panel. The use of the training set option is selected and then we click the 'start' button. This process and resulting window are shown in the following screenshots.

Step 6 : 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 are means vectors for each clusters. This clusters can be used to characterized the cluster. For eg, the centroid of cluster1 shows the class iris.versicolor mean value of the sepal length is

5.4706, sepal width 2.4765, petal width 1.1294, petal length 3.7941.

Step 7: 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.

The following screenshot shows the clustering rules that were generated when simple k means algorithm is applied on the given dataset.

```
15:54:43 - EM

=== Run information ===

Scheme:      weka.clusterers.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
Relation:    student
Instances:    10
Attributes:   3
              name
              percentage
              play
Test mode:    evaluate on training data

=== Clustering model (full training set) ===

EM
==

Number of clusters selected by cross validation: 2
Number of iterations performed: 0
```

```
15:54:43 - EM

Attribute      Cluster
              0      1
              (0.6) (0.4)
=====
name
Alex           1      2
Twinkle        2      1
Geetu          2      1
Shalu          1      2
Kanika         2      1
Karishma       2      1
Ayush          1      2
Akshay         2      1
Sarvesh        2      1
Mahesh         1      2
[total]        16     14
percentage
mean           82     83.25
std. dev.      6.7231  4.113

play
yes            7      1
no             1      5
[total]        8      6

Time taken to build model (full training data) : 0.09 seconds

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

Clustered Instances

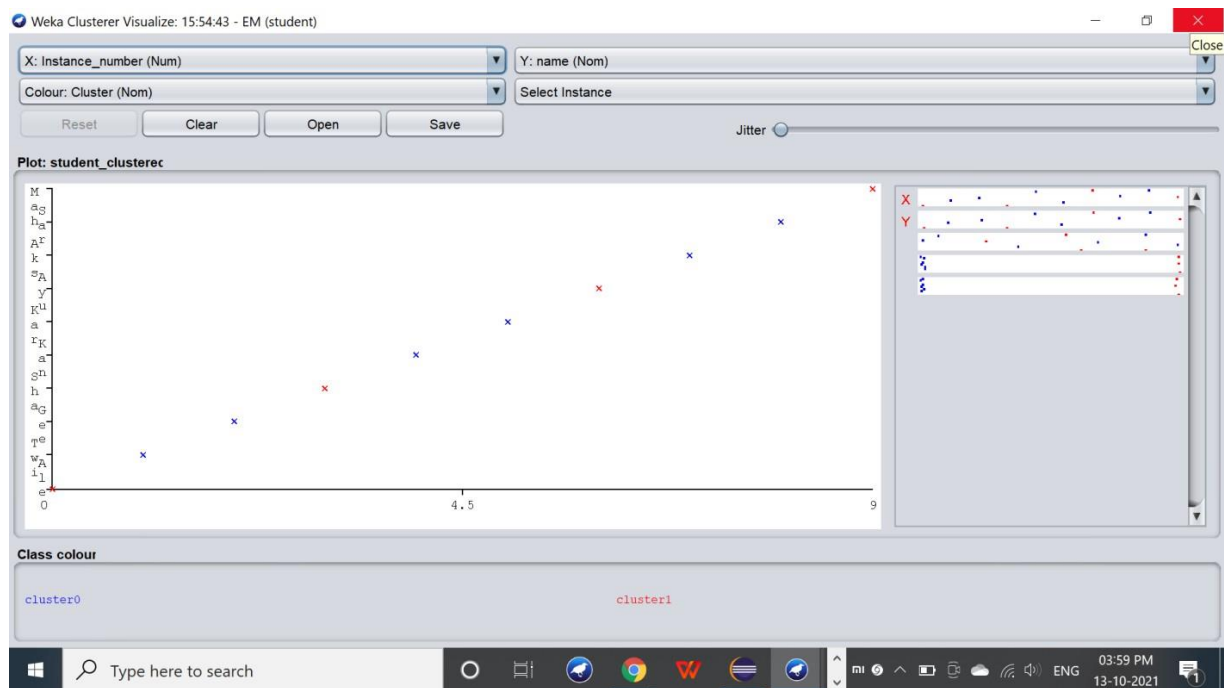
0      6 ( 60%)
1      4 ( 40%)

Log likelihood: -5.81082
```

Interpretation of the above visualization

From the above visualization, we can understand the distribution of sepal length and petal length in each cluster. For instance, for each cluster is dominated by petal length. In this case by changing the color dimension to other attributes we can see their distribution within each of the cluster.

Step 8: We can assure that resulting dataset which included each instance along with its assigned cluster. To do so we click the save button in the visualization window and save the result iris k-mean. The top portion of this file is shown in the following figure.



Experiment 6

Aim : Implementation of Association Rule technique on ARFF files using WEKA.

This experiment illustrates some of the basic elements of association rule mining using WEKA. The sample dataset used for this example is contactlenses.arff

Step1: Open the data file in Weka Explorer. It is presumed that the required data fields have been discretized. In this example it is age attribute.

Step2: Clicking on the associate tab will bring up the interface for association rule algorithm.

Step3: We will use apriori algorithm. This is the default algorithm.

Step4: Inorder 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.

Dataset contactlenses.arff

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: contact-lenses, Instances: 24, Attributes: 5, Sum of weights: 24

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

Attributes: All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> age
2	<input type="checkbox"/> spectacle-prescrip
3	<input type="checkbox"/> astigmatism
4	<input type="checkbox"/> tear-prod-rate
5	<input type="checkbox"/> contact-lenses

Remove

Class: contact-lenses (Nom) Visualize All

Status: OK Log x 0

ARFF-Viewer - C:\Users\DEEPIKA SINGH\OneDrive\Desktop\contactlenses.arff

File Edit View

contactlenses.arff

Relation: contact-lenses

No.	1: age Nominal	2: spectacle-prescrip Nominal	3: astigmatism Nominal	4: tear-prod-rate Nominal	5: contact-lenses Nominal
1	young	myope	no	reduced	none
2	young	myope	no	normal	soft
3	young	myope	yes	reduced	none
4	young	myope	yes	normal	hard
5	young	hypermetrope	no	reduced	none
6	young	hypermetrope	no	normal	soft
7	young	hypermetrope	yes	reduced	none
8	young	hypermetrope	yes	normal	hard
9	pre-...	myope	no	reduced	none
10	pre-...	myope	no	normal	soft
11	pre-...	myope	yes	reduced	none
12	pre-...	myope	yes	normal	hard
13	pre-...	hypermetrope	no	reduced	none
14	pre-...	hypermetrope	no	normal	soft
15	pre-...	hypermetrope	yes	reduced	none
16	pre-...	hypermetrope	yes	normal	none
17	pres...	myope	no	reduced	none
18	pres...	myope	no	normal	none
19	pres...	myope	yes	reduced	none
20	pres...	myope	yes	normal	hard
21	pres...	hypermetrope	no	reduced	none
22	pres...	hypermetrope	no	normal	soft
23	pres...	hypermetrope	yes	reduced	none
24	pres...	hypermetrope	yes	normal	none

The following screenshot shows the association rules that were generated when apriori algorithm is applied on the given dataset.

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 to expand/collapse)

12:04:13 - Apriori

Associator output

```

=== Run information ===

Scheme:      weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:    contact-lenses
Instances:    24
Attributes:   5
    age
    spectacle-prescrip
    astigmatism
    tear-prod-rate
    contact-lenses

=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.2 (5 instances)
Minimum metric <confidence>: 0.9

```

Status

OK Log

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Associate

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

Start Stop

Result list (right-click to open)

12:04:13 - Apriori

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)

Status

OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Associate

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

Start Stop

Result list (right-click to open)

12:04:13 - Apriori

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

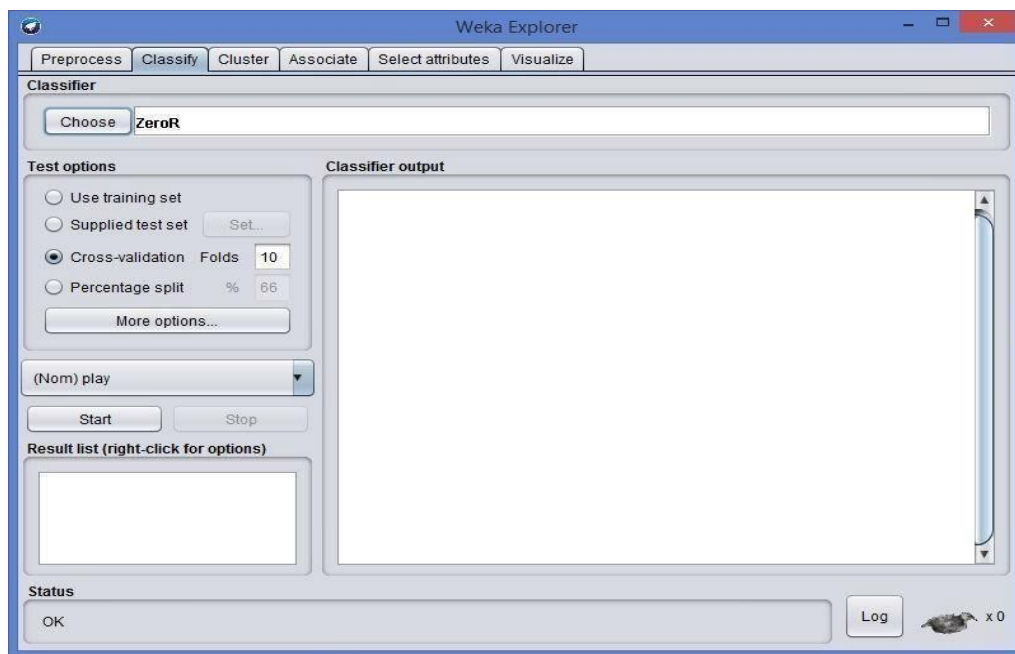
Experiment 7: Implementation of Visualization technique on ARFF files using WEKA

To search through all possible combinations of attributes in the data and find which subset of attributes works best for prediction, make sure that you set up attribute evaluator to „Cfs Subset Val“ and a search method to „Best First“. The evaluator will determine what method to use to assign a worth to each subset of attributes. The search method will determine what style of search to perform. The options that you can set for selection in the „Attribute Selection Mode“ fig no: 3.2

1. **Use full training set.** The worth of the attribute subset is determined using the full set of training data.

2. **Cross-validation.** The worth of the attribute subset is determined by a process of cross-validation. The „Fold“ and „Seed“ fields set the number of folds to use and the random seed used when shuffling the data.

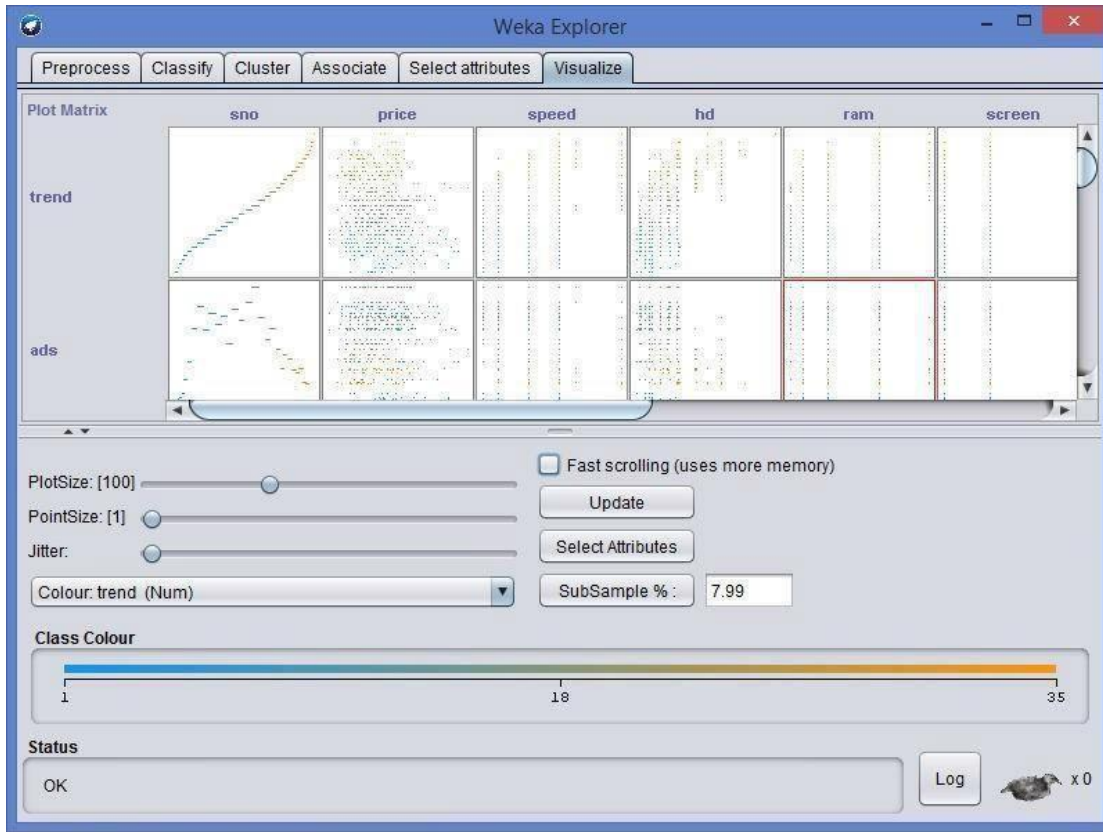
Specify which attribute to treat as the class in the drop-down box below the test options. Once all the test options are set, you can start the attribute selection process by clicking on „Start“ button.



Choosing Cross validation

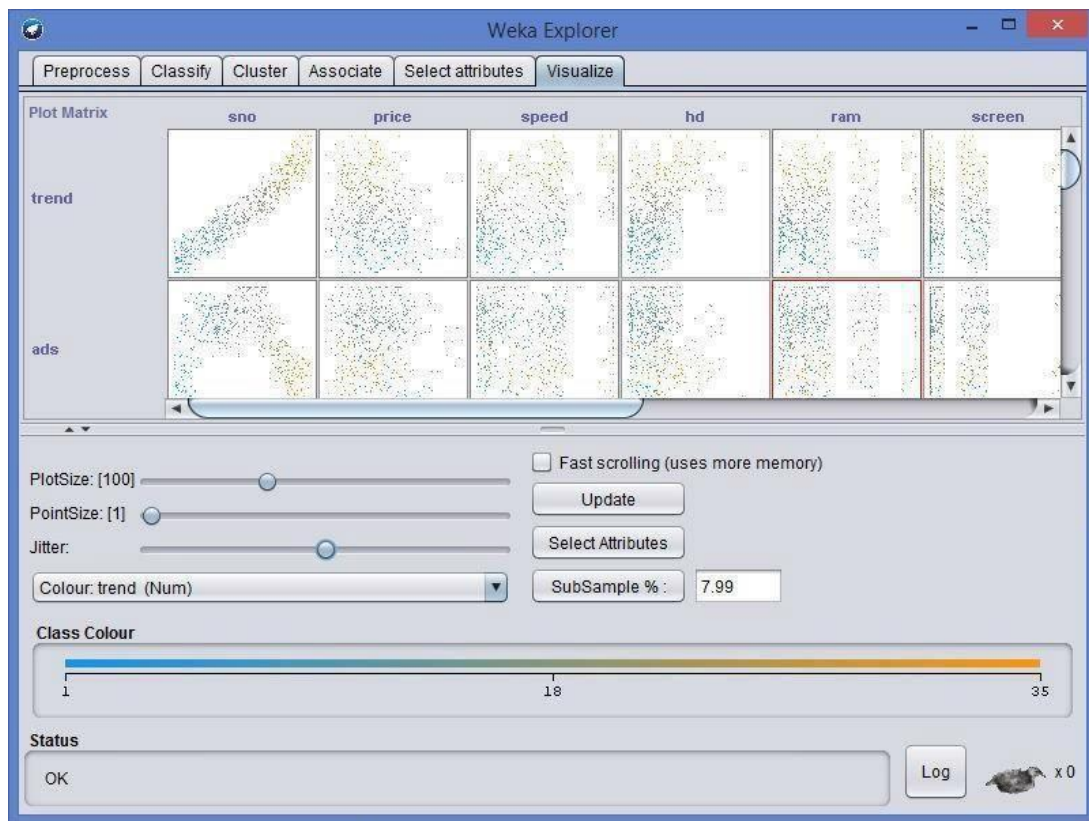
When it is finished, the results of selection are shown on the right part of the window and entry is added to the „Result list“.

2. Visualizing Results

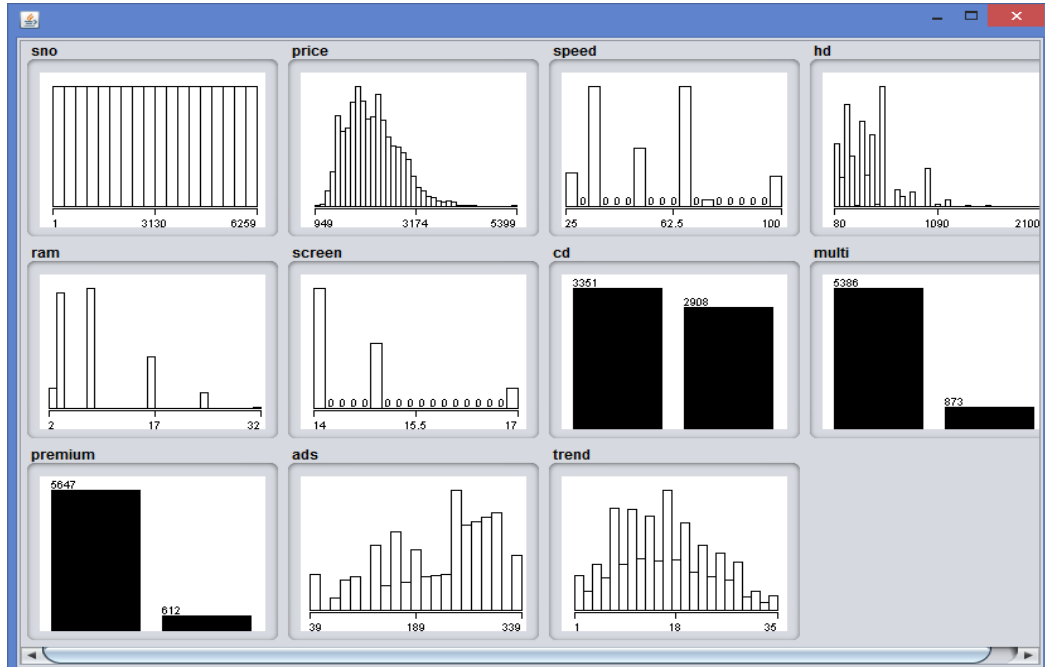


Data Visualization

WEKA's visualization allows you to visualize a 2-D plot of the current working relation. Visualization is very useful in practice; it helps to determine difficulty of the learning problem. WEKA can visualize single attributes (1-d) and pairs of attributes (2-d), rotate 3-d visualizations (Xgobi-style). WEKA has "Jitter" option to deal with nominal attributes and to detect "hidden" data points.



Preprocessing with jitter



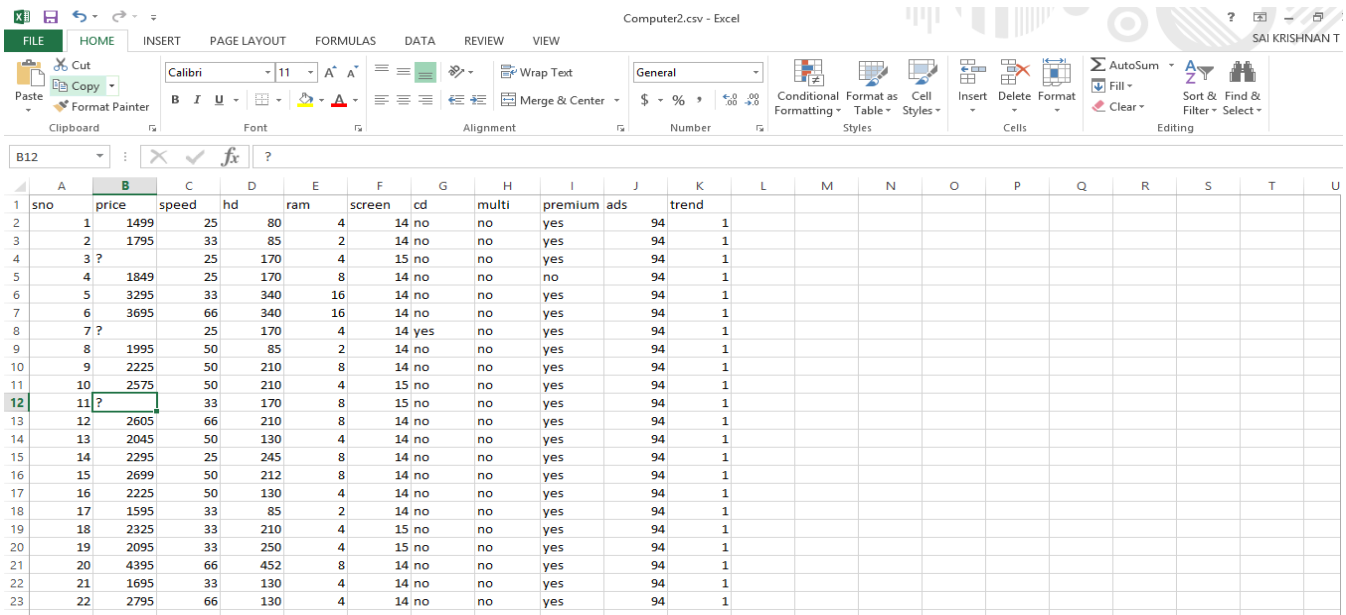
Data visualization

Exercise

1. Explain data preprocessing steps for heart disease dataset.

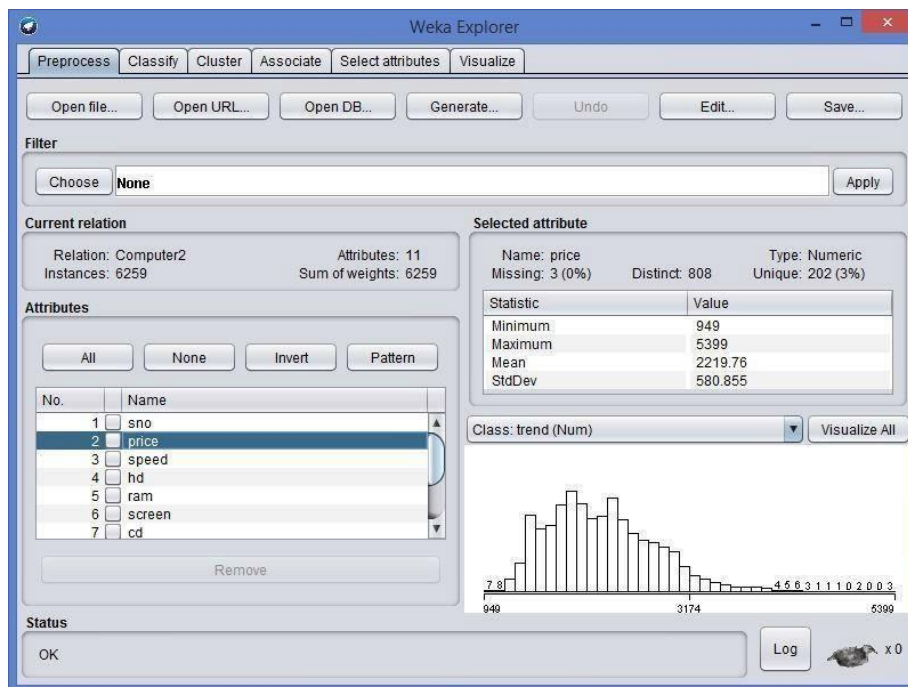
Aim: B. Pre-process a given dataset based on Handling Missing Values

Process: Replacing Missing Attribute Values by the Attribute Mean. This method is used for data sets with numerical attributes. An example of such a data set is presented in fig no: 3.4



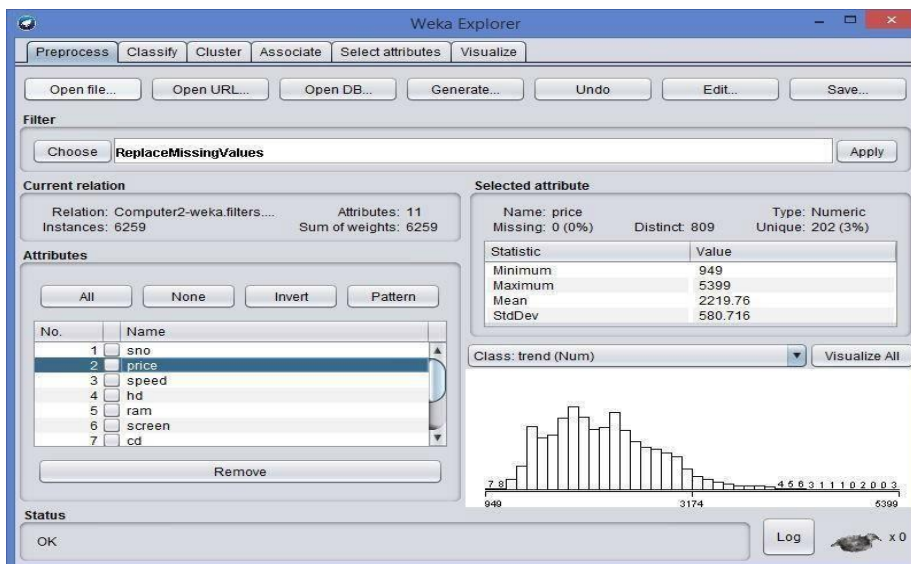
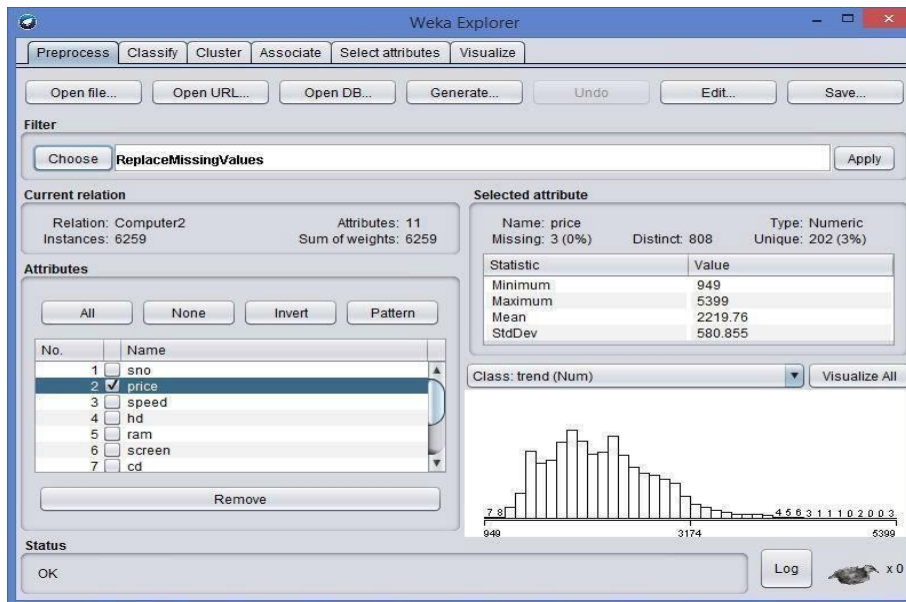
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	sno	price	speed	hd	ram	screen	cd	multi	premium	ads	trend										
2	1	1499	25	80	4	14	no	no	yes	94	1										
3	2	1795	33	85	2	14	no	no	yes	94	1										
4	3	?	25	170	4	15	no	no	yes	94	1										
5	4	1849	25	170	8	14	no	no	no	94	1										
6	5	3295	33	340	16	14	no	no	yes	94	1										
7	6	3695	66	340	16	14	no	no	yes	94	1										
8	7	?	25	170	4	14	yes	no	yes	94	1										
9	8	1995	50	85	2	14	no	no	yes	94	1										
10	9	2225	50	210	8	14	no	no	yes	94	1										
11	10	2575	50	210	4	15	no	no	yes	94	1										
12	11	?	33	170	8	15	no	no	yes	94	1										
13	12	2605	66	210	8	14	no	no	yes	94	1										
14	13	2045	50	130	4	14	no	no	yes	94	1										
15	14	2295	25	245	8	14	no	no	yes	94	1										
16	15	2699	50	212	8	14	no	no	yes	94	1										
17	16	2225	50	130	4	14	no	no	yes	94	1										
18	17	1595	33	85	2	14	no	no	yes	94	1										
19	18	2325	33	210	4	15	no	no	yes	94	1										
20	19	2095	33	250	4	15	no	no	yes	94	1										
21	20	4395	66	452	8	14	no	no	yes	94	1										
22	21	1695	33	130	4	14	no	no	yes	94	1										
23	22	2795	66	130	4	14	no	no	yes	94	1										

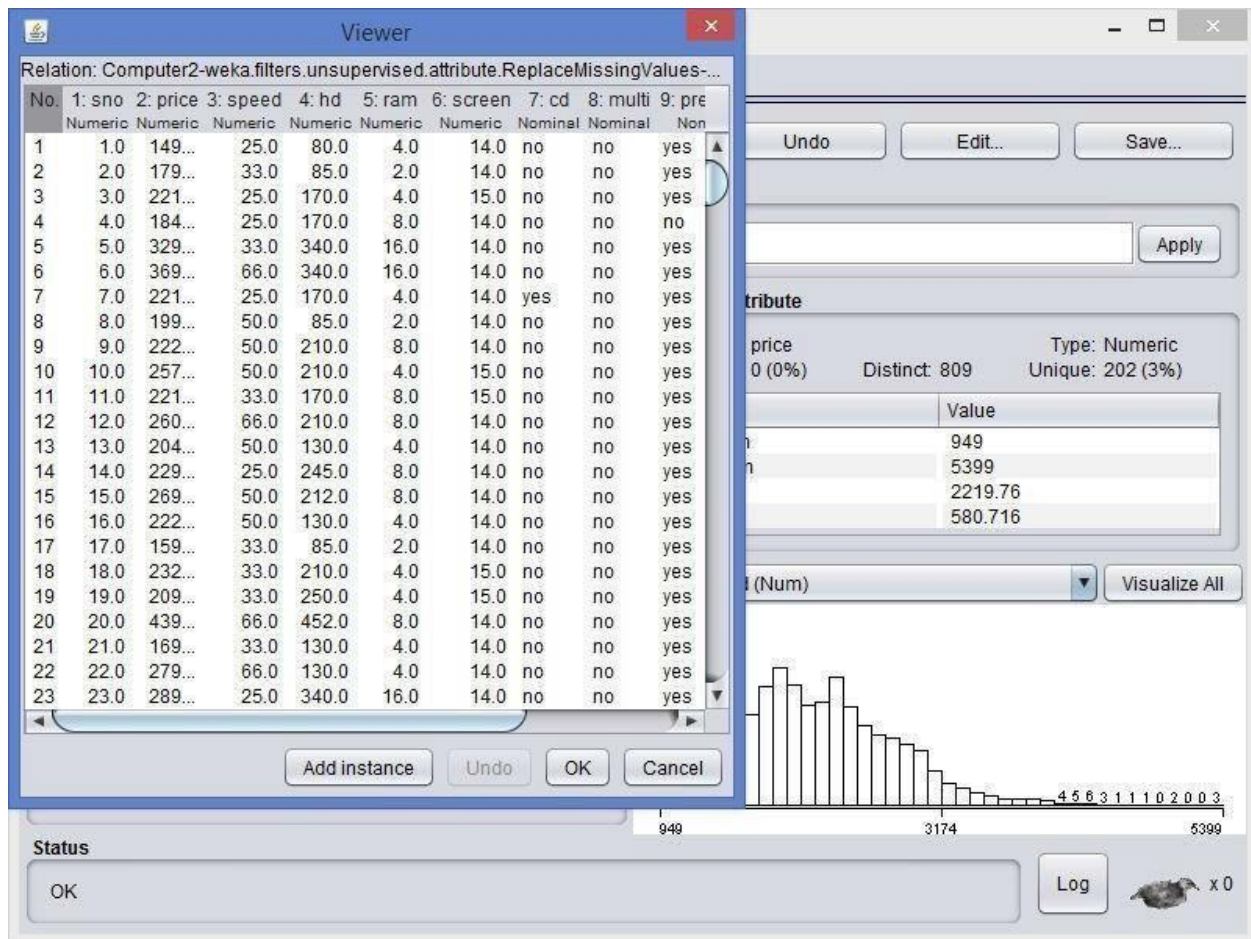
Missing values



Choosing a dataset

In this method, every missing attribute value for a numerical attribute is replaced by the arithmetic mean of known attribute values. In Fig, the mean of known attribute values for Temperature is 99.2, hence all missing attribute values for Temperature should be replaced by The table with missing attribute values replaced by the mean is presented in fig. For symbolic attributes Headache and Nausea, missing attribute values were replaced using the most common value of the Replace Missing Values.





ComputerReplaced.csv - Excel

SAI KRISHNAN T

	sno	price	speed	hd	ram	screen	cd	multi	premium	ads	trend
1	1	1499	25	80	4	14	no	no	yes	94	1
2	2	1795	33	85	2	14	no	no	yes	94	1
3	3	2219.76	25	170	4	15	no	no	yes	94	1
4	4	1849	25	170	8	14	no	no	no	94	1
5	5	3295	33	340	16	14	no	no	yes	94	1
6	6	3695	66	340	16	14	no	no	yes	94	1
7	7	2219.76	25	170	4	14	yes	no	yes	94	1
8	8	1995	50	85	2	14	no	no	yes	94	1
9	9	2225	50	210	8	14	no	no	yes	94	1
10	10	2575	50	210	4	15	no	no	yes	94	1
11	11	2219.76	33	170	8	15	no	no	yes	94	1
12	12	2605	66	210	8	14	no	no	yes	94	1
13	13	2045	50	130	4	14	no	no	yes	94	1
14	14	2295	25	245	8	14	no	no	yes	94	1
15	15	2699	50	212	8	14	no	no	yes	94	1
16	16	2225	50	130	4	14	no	no	yes	94	1
17	17	1595	33	85	2	14	no	no	yes	94	1
18	18	2325	33	210	4	15	no	no	yes	94	1
19	19	2095	33	250	4	15	no	no	yes	94	1
20	20	4395	66	452	8	14	no	no	yes	94	1
21	21	1695	33	130	4	14	no	no	yes	94	1
22	22	2795	66	130	4	14	no	no	yes	94	1
23	23	2895	25	340	16	14	no	no	yes	94	1

Replaced values

EXPERIMENT-8

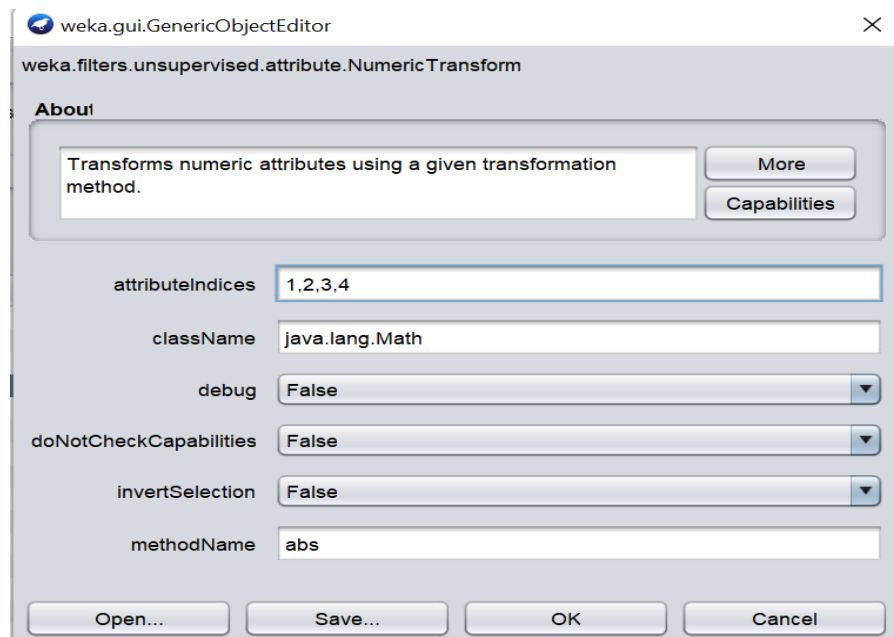
AIM: Implementation of Visualization technique on ARFF files using WEKA

Steps to be followed:-

1. Open segment-challenge.arff in the explorer in weka.



2. Then select the choose option after opening the file.
3. Perform the following:-
Choose -> filters -> unsupervised -> Numeric transform method
4. Click and fill the index of the column whose values are to be rounded off.



5. Apply these changes to all the columns by selecting all and click apply.
6. Click Edit to see the viewer as shown.

No.	1: region-centroid-col	2: region-centroid-row	3: region-pixel-count	4: short-line-density-5	5: short-line-density-2	6: vedge-mean	7: vedge-sd	8: hedge-mean	9: hedge-sd	10: hedge-sd
1	38.0	189.0	9.0	0.0	0.0	1.0	0.222222	6.22222	33.3185	
2	25.0	199.0	9.0	0.0	0.0	1.11111	0.607407	1.05556	0.462963	
3	49.0	139.0	9.0	0.0	0.0	0.166667	0.0777778	0.333333	0.0888889	
4	63.0	220.0	9.0	0.0	0.0	3.05556	15.263	3.66667	6.08889	
5	161.0	135.0	9.0	0.0	0.0	0.055556	0.136083	0.111111	0.172133	
6	235.0	88.0	9.0	0.0	0.0	0.611111	0.240741	0.944445	0.32963	
7	67.0	32.0	9.0	0.0	0.0	0.944444	1.06284	1.77778	1.31092	
8	188.0	182.0	9.0	0.0	0.0	1.61111	0.742868	4.16667	2.12655	
9	217.0	245.0	9.0	0.111111	0.111111	3.16667	3.01662	2.16667	1.24276	
10	9.0	171.0	9.0	0.0	0.0	1.5	1.00554	2.77778	1.64204	
11	149.0	117.0	9.0	0.222222	0.0	0.833333	0.255556	1.0	0.444445	
12	136.0	139.0	9.0	0.0	0.0	0.666667	0.177778	0.777778	0.207407	
13	214.0	79.0	9.0	0.0	0.0	1.27778	0.862963	1.33333	0.533333	
14	118.0	125.0	9.0	0.0	0.0	0.333333	0.298142	0.888889	0.344265	
15	83.0	53.0	9.0	0.0	0.0	0.555556	0.455419	0.722223	0.490652	
16	70.0	38.0	9.0	0.0	0.0	1.22222	0.272166	1.0	0.760116	
17	96.0	84.0	9.0	0.0	0.0	1.5	1.27778	1.61111	2.28519	
18	151.0	129.0	9.0	0.0	0.0	8.05556	8.14157	0.722222	0.827758	
19	98.0	144.0	9.0	0.0	0.0	0.222222	0.02963	0.333333	0.0888889	
20	146.0	140.0	9.0	0.0	0.0	1.05556	0.462963	1.0	0.577778	
21	162.0	237.0	9.0	0.111111	0.0	2.27778	1.14342	2.66667	2.56472	
22	155.0	27.0	9.0	0.0	0.0	1.05556	0.240737	1.44445	0.25185	
23	83.0	198.0	9.0	0.0	0.0	2.38889	1.49691	4.83333	3.00925	
24	94.0	104.0	9.0	0.0	0.0	6.61111	1.42075	1.0	0.516399	
25	220.0	111.0	9.0	0.0	0.0	2.55556	1.65552	2.27778	1.59745	
26	157.0	85.0	9.0	0.0	0.0	1.22222	1.24127	0.222222	0.172133	
27	18.0	145.0	9.0	0.0	0.0	0.388889	0.018519	0.611111	0.374074	
28	95.0	191.0	9.0	0.0	0.111111	3.16667	2.78687	6.0	5.18545	
29	112.0	38.0	9.0	0.0	0.0	11.0556	11.0743	1.11111	1.06805	

CONCLUSION:

By using Numeric transform filter and floor method, required values have been obtained.

Experiment 9: Implementation of Apriori Algorithm

Description:

The Apriori algorithm is an influential algorithm for mining frequent item sets for Boolean association rules. It uses a “bottom-up” approach, where frequent subsets are extended one at a time (a step known as candidate generation, and groups of candidates are tested against the data).

❖ Problem:

TID	ITEMS
100	1,3,4
200	2,3,5
300	1,2,3,5
400	2,5

To find frequent item sets for above transaction with a minimum support of 2 having confidence measure of 70% (i.e, 0.7).

Procedure:

Step 1:

Count the number of transactions in which each item occurs

TID	ITEMS
1	2
2	3
3	3
4	1
5	3

Step 2:

Eliminate all those occurrences that have transaction numbers less than the minimum support (2 in this case).

ITEM	NO. OF TRANSACTIONS
1	2
2	3
3	3
5	3

This is the single items that are bought frequently. Now let's say we want to find a pair of items that are bought frequently. We continue from the above table (Table in step 2).

Step 3:

We start making pairs from the first item like 1,2;1,3;1,5 and then from second item like 2,3;2,5. We do not perform 2,1 because we already did 1,2 when we were making pairs with 1 and buying 1 and 2 together is same as buying 2 and 1 together. After making all the pairs we get,

ITEM PAIRS
1,2
1,3
1,5
2,3
2,5
3,5

Step 4:

Now, we count how many times each pair is bought together.

ITEM PAIRS	NO.OF TRANSACTIONS
1,2	1
1,3	2
1,5	1
2,3	2
2,5	3
3,5	2

Step 5:

Again remove all item pairs having number of transactions less than 2.

ITEM PAIRS	NO. OF TRANSACTIONS
1,3	2
2,3	2
2,5	3
3,5	2

These pair of items is bought frequently together. Now, let's say we want to find a set of three items that are bought together. We use above table (of step 5) and make a set of three items.

Step 6:

To make the set of three items we need one more rule (It is termed as self-join), it simply means, from item pairs in above table, we find two pairs with the same first numeric, so, we get (2,3) and (2,5), which gives (2,3,5). Then we find how many times (2, 3, 5) are bought together in the original table and we get the following

ITEM SET	NO. OF TRANSACTIONS
(2,3,5)	2

Thus, the set of three items that are bought together from this data are (2, 3, 5).

Confidence:

We can take our frequent item set knowledge even further, by finding association rules using the frequent item set. In simple words, we know (2, 3, 5) are bought together frequently, but what is the association between them. To do this, we create a list of all subsets of frequently bought items (2, 3, 5) in our case we get following subsets:

- {2}
- {3}
- {5}
- {2,3}
- {3,5}
- {2,5}

Now, we find association among all the subsets.

$\{2\} \Rightarrow \{3,5\}$: (If „2“ is bought, what's the probability that „3“ and „5“ would be bought in same transaction)

$$\text{Confidence} = P(3 \square 5 \square 2) / P(2) = 2/3 = 67\%$$

$$\{3\} \Rightarrow \{2,5\} = P(3 \square 5 \square 2) / P(3) = 2/3 = 67\%$$

$$\{5\} \Rightarrow \{2,3\} = P(3 \square 5 \square 2) / P(5) = 2/3 = 67\%$$

$$\{2,3\} \Rightarrow \{5\} = P(3 \square 5 \square 2) / P(2 \square 3) = 2/2 = 100\%$$

$$\{3,5\} \Rightarrow \{2\} = P(3 \square 5 \square 2) / P(3 \square 5) = 2/2 = 100\%$$

$$\{2,5\} \Rightarrow \{3\} = P(3 \square 5 \square 2) / P(2 \square 5) = 2/3 = 67\%$$

Also, considering the remaining 2-items sets, we would get the following associations-

$$\{1\} \Rightarrow \{3\} = P(1 \square 3) / P(1) = 2/2 = 100\%$$

$$\{3\} \Rightarrow \{1\} = P(1 \square 3) / P(3) = 2/3 = 67\%$$

$$\{2\} \Rightarrow \{3\} = P(3 \square 2) / P(2) = 2/3 = 67\%$$

$$\{3\} \Rightarrow \{2\} = P(3 \square 2) / P(3) = 2/3 = 67\%$$

$$\{2\} \Rightarrow \{5\} = P(2 \square 5) / P(2) = 3/3 = 100\%$$

$$\{5\} \Rightarrow \{2\} = P(2 \square 5) / P(5) = 3/3 = 100\%$$

$$\{3\} \Rightarrow \{5\} = P(3 \square 5) / P(3) = 2/3 = 67\%$$

$$\{5\} \Rightarrow \{3\} = P(3 \square 5) / P(5) = 2/3 = 67\%$$

Eliminate all those having confidence less than 70%. Hence, the rules would be –

$\{2,3\} \Rightarrow \{5\}$, $\{3,5\} \Rightarrow \{2\}$, $\{1\} \Rightarrow \{3\}$, $\{2\} \Rightarrow \{5\}$, $\{5\} \Rightarrow \{2\}$.

- Now these manual results should be checked with the rules generated in WEKA.

Clipboard		Font				
A1		f _x				
		I1				
	A	B	C	D	E	F
1	I1	I2	I3	I4	I5	
2	t		t	t		
3		t	t		t	
4	t	t	t		t	
5		t			t	
6						
7						
8						
9						

So first create a csv file for the above problem, the csv file for the above problem will look like the rows and columns in the above figure. This file is written in excel sheet.

Procedure for running the rules in weka:

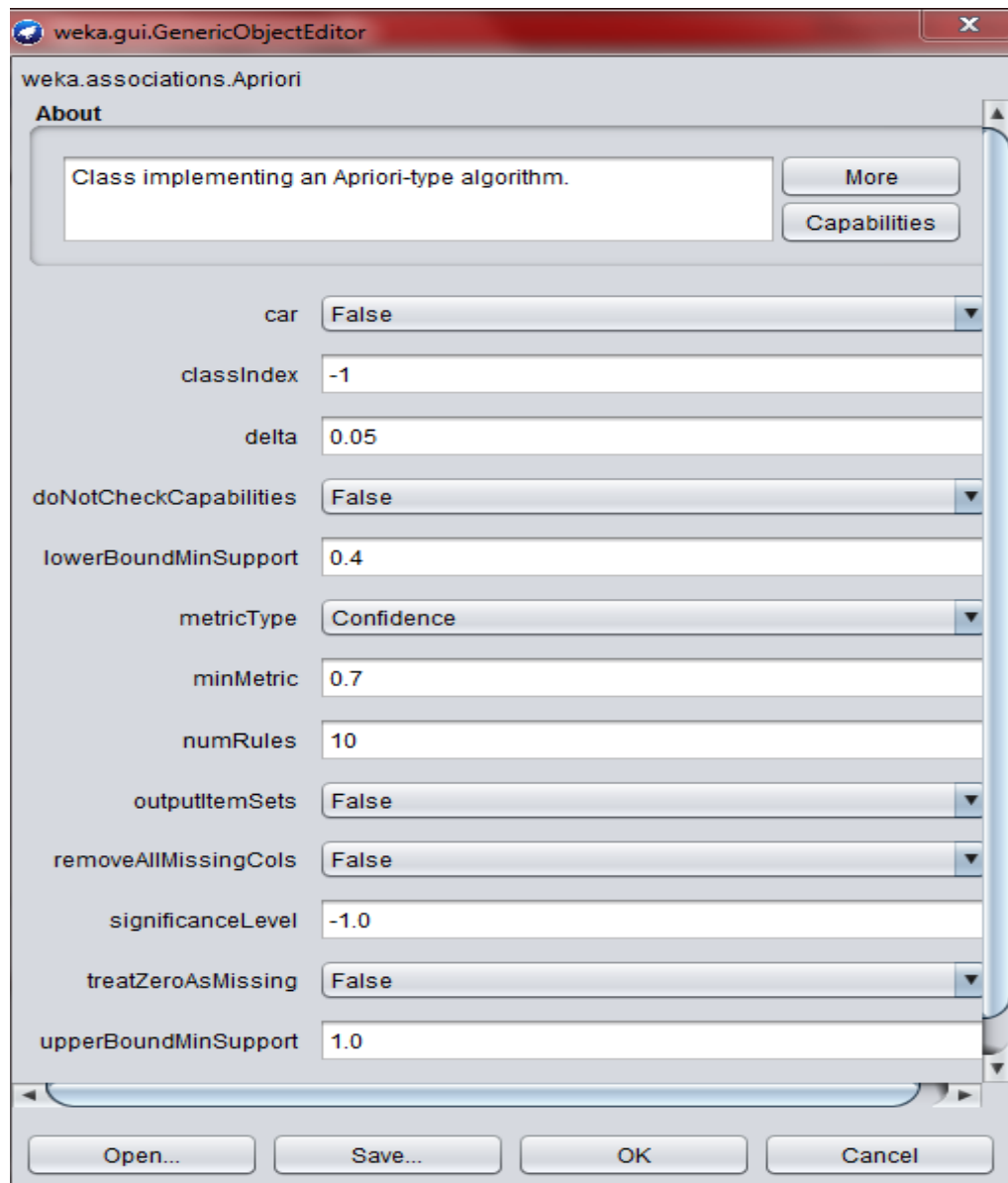
Step 1:

Open weka explorer and open the file and then select all the item sets. The figure gives a better understanding of how to do that.



Step 2:

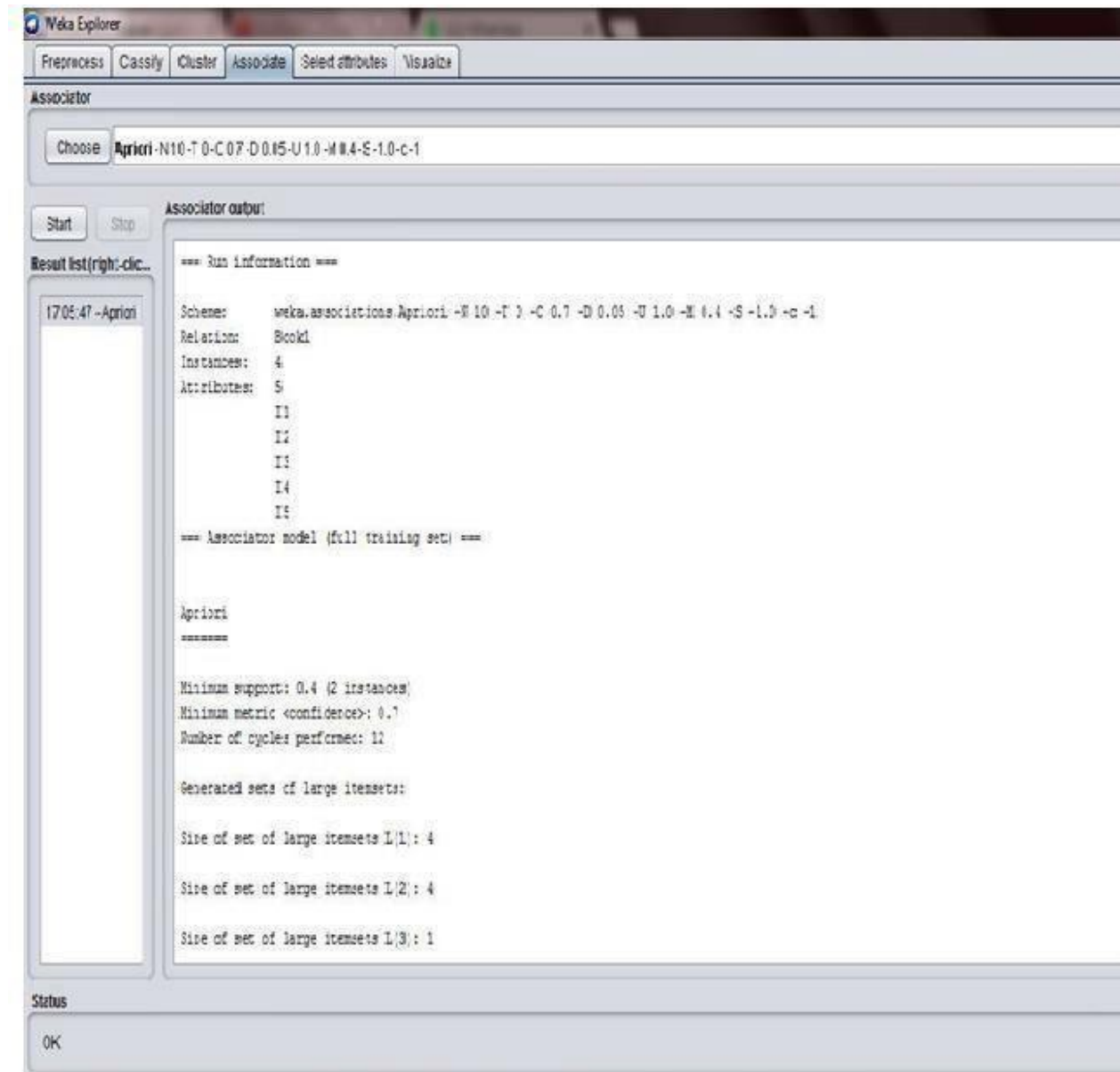
Now select the association tab and then choose apriori algorithm by setting the minimumsupport and confidence as shown in the figure



Step 3:

Now run the apriori algorithm with the set values of minimum support and the confidence. After running the weka generates the association rules and the respective confidence with minimum support as shown in the figure.

The above csv file has generated 5 rules as shown in the figure:



Associator output

```
I4
I5
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.4 (2 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 12

Generated sets of large itemsets:

Size of set of large itemsets L(1): 4

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

Size of set of large itemsets L(3): 1

Best rules found:

1. I5=t 3 ==> I2=t 3   <conf:(1)> lift:(1.33) lev:(0.19) [0] conv:(0.75)
2. I2=t 3 ==> I5=t 3   <conf:(1)> lift:(1.33) lev:(0.19) [0] conv:(0.75)
3. I1=t 2 ==> I3=t 2   <conf:(1)> lift:(1.33) lev:(0.13) [0] conv:(0.5)
4. I3=t I5=t 2 ==> I2=t 2   <conf:(1)> lift:(1.33) lev:(0.13) [0] conv:(0.5)
5. I2=t I3=t 2 ==> I5=t 2   <conf:(1)> lift:(1.33) lev:(0.13) [0] conv:(0.5)
```

Conclusion:

As we have seen the total rules generated by us manually and by the weka are matching, hence the rules generated are 5.

Experiment 10: Implementation of K-means algorithm

DESCRIPTION:

K-means algorithm aims to partition n observations into “ k clusters” in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in partitioning of the data into Voronoi cells.

ILLUSTRATION:

As a simple illustration of a k-means algorithm, consider the following data set consisting of the scores of two variables on each of the five variables.

I	X1	X2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

This data set is to be grouped into two clusters: As a first step in finding a sensible partition, let the A & C values of the two individuals furthest apart (using the Euclidean distance measure), define the initial cluster means, giving:

Cluster	Individual	Mean Vector(Centroid)
Cluster1	A	(1,1)
Cluster2	C	(0,2)

The remaining individuals are now examined in sequence and allocated to the cluster to which they are closest, in terms of Euclidean distance to the cluster mean. The mean vector is recalculated each time a new member is added. This leads to the following series of steps:

	A	C
A	0	1.4
B	1	2.5
C	1.4	0
D	3.2	2.82
E	4.5	4.2

Initial partitions have changed, and the two clusters at this stage having the following characteristics.

	Individual	Mean vector(Centroid)
Cluster 1	A,B	(1,0.5)
Cluster 2	C,D,E	(1.7,3.7)

But we cannot yet be sure that each individual has been assigned to the right cluster. So, we compare each individual's distance to its own cluster mean and to that of the opposite cluster. And, we find:

I	A	C
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

The individuals C is now relocated to Cluster 1 due to its less mean distance with the centroid points. Thus, its relocated to cluster 1 resulting in the new partition

	Individual	Mean vector(Centroid)
Cluster 1	A,B,C	(0.7,1)
Cluster 2	D,E	(2.5,4.5)

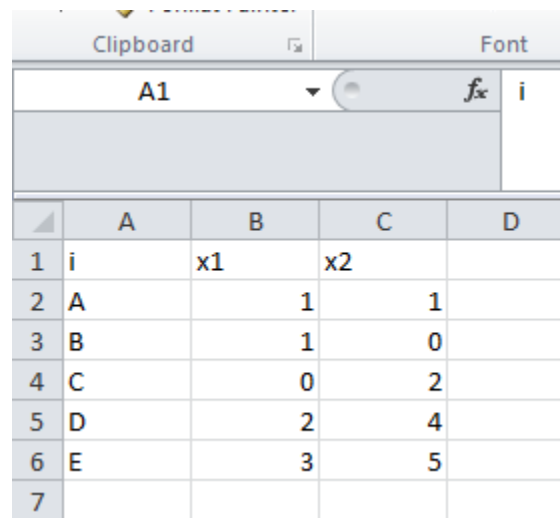
The iterative relocation would now continue from this new partition until no more relocation occurs. However, in this example each individual is now nearer its own cluster mean than that of the other cluster and the iteration stops, choosing the latest partitioning as the final cluster solution. Also, it is possible that the k-means algorithm won't find a final solution. In this case, it would be a better idea to consider stopping the algorithm after a pre-chosen maximum number of iterations.

Checking the solution in weka:

In order to check the result in the tool we need to follow a procedure.

Step 1:

Create a csv file with the above table considered in the example. the csv file will look as shown below:



	A	B	C	D
1	i	x1	x2	
2	A	1	1	
3	B	1	0	
4	C	0	2	
5	D	2	4	
6	E	3	5	
7				

Step 2:

Now open weka explorer and then select all the attributes in the table.

Filter

Choose **None** Apply

Current relation

Relation: menas
Instances: 5

Attributes: 3
Sum of weights: 5

Attributes

All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> i
2	<input checked="" type="checkbox"/> x1
3	<input checked="" type="checkbox"/> x2

Remove

Selected attribute


Name: i
Missing: 0 (0%)
Distinct: 5
Type: Nominal
Unique: 5 (100%)

No.	Label	Count	Weight
1	A	1	1.0
2	B	1	1.0
3	C	1	1.0
4	D	1	1.0

Class: x2 (Num) Visualize All

1 1 1 1 1

Status

OK Log  x 0

Step 3:

Select the cluster tab in the tool and choose normal k-means technique to see the result as shown below.

```
Clusterer output

=== Run information ===

Scheme:      weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "
Relation:    menas
Instances:    5
Attributes:   3
              i
              x1
              x2
Test mode:    evaluate on training data

=== Clustering model (full training set) ===

kMeans
=====

Number of iterations: 2
Within cluster sum of squared errors: 3.22962962962963

Initial starting points (random):

Cluster 0: D,2,4
Cluster 1: B,1,0

Missing values globally replaced with mean/mode
```

Final cluster centroids:

Attribute	Full Data (5.0)	Cluster#	
		0 (2.0)	1 (3.0)
=====			
i	A	D	A
x1	1.4	2.5	0.6667
x2	2.4	4.5	1

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

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

Clustered Instances

0	2 (40%)
1	3 (60%)