INDUSTRIAL UNIVERSITY OF HO CHI MINH CITY
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE

PRACTICAL MANUALS
DATA MINING AND APPLICATIONS

2023

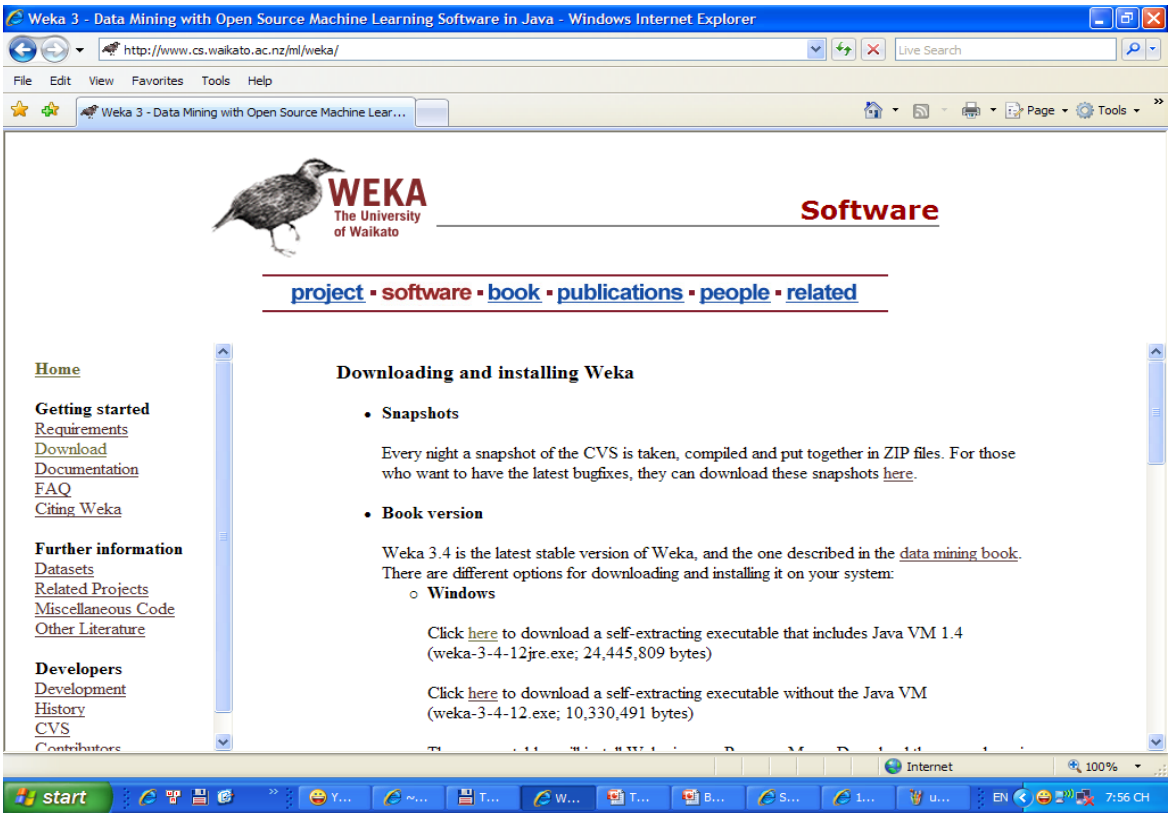
Lab 1 - 2	**Data Mining with WEKA** **Data Preprocessing**

I. Introduction to WEKA

- Weka (**WEKA** – **W**aikato **E**nvironment for **K**nowledge **A**nalysis is an open-source Java development environment for data mining from the University of Waikato in New Zealand. It can be downloaded freely from http://www.cs.waikato.ac.nz/ml/weka/ . Weka is really an asset for learning data mining because it is freely available, students can study how the different data mining models are implemented and develop customized Java data mining applications.
- A Collection of Machine Learning algorithms for data tasks. Using Weka to build models, predict class, evaluate model, compare different models.
- Java Code:
  - Insert into your java application.
  - Use GUI for your own classifier.

II. How to install and use Weka program

1. Download Weka program on the website: http://www.cs.waikato.ac.nz/ml/weka/

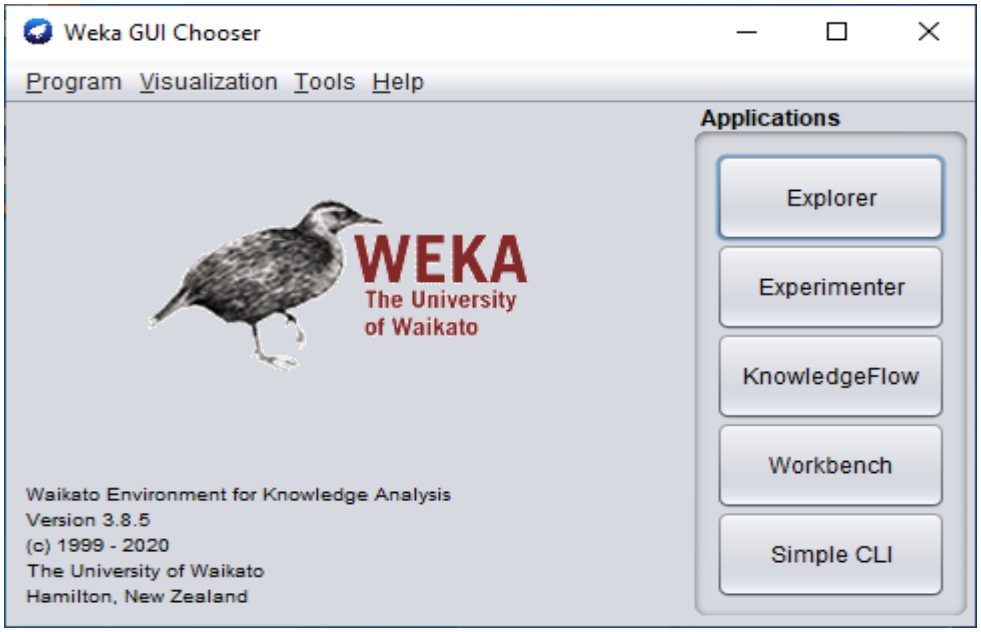


2. Install

- File: weka-3-8-6jre.exe
- Size: about 30MB
- Noted:
  - Install with full option.
  - After installing Weka, may be install J2SE

- Setting folder: Weka-3-8

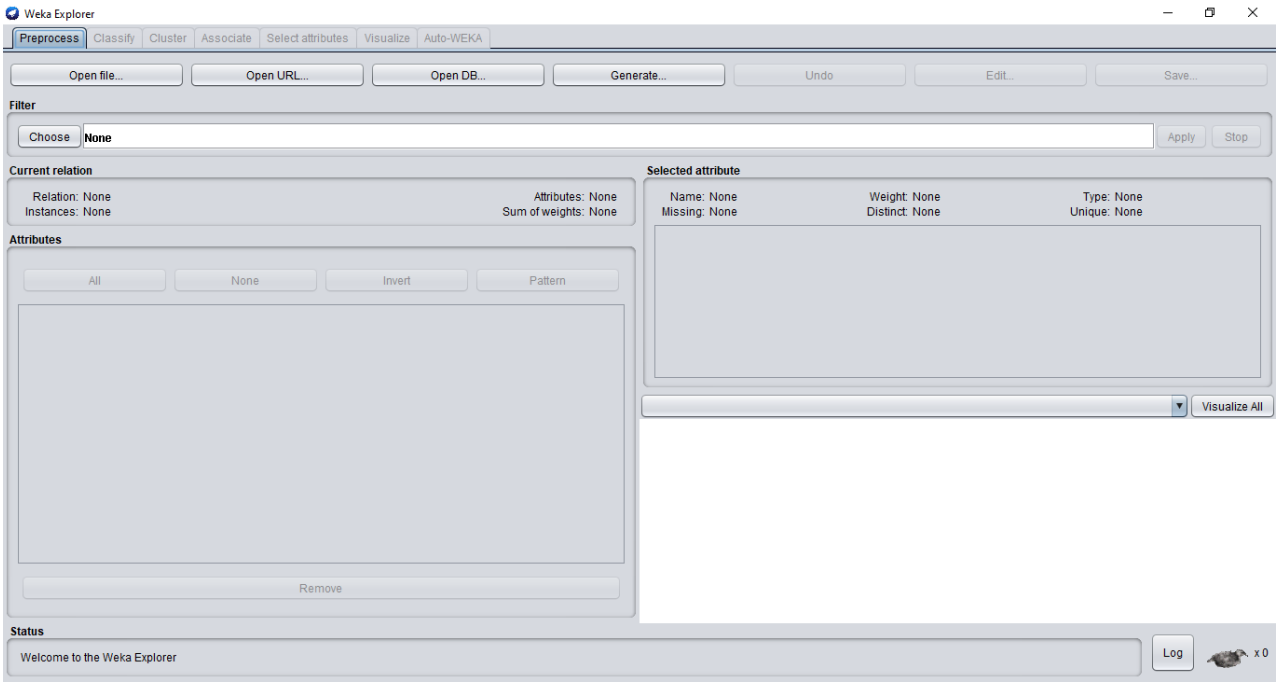
3. GUI



This is the GUI you get when started. You have 5 options **Explorer** (graphical user interface), **Experimenter**, **KnowledgeFlow, Workbench** and *Simple CLI* (command line interface). The buttons can be used to start the following applications:

- **Explorer:** An environment for exploring data with WEKA (the rest of this Documentation deals with this application in more detail)
- **Experimenter:** An environment for performing experiments and conducting statistical tests between learning schemes.
- **KnowledgeFlow:** This environment supports essentially the same functions as Explorer but with a drag-and-drop interface. One advantage is that it supports incremental learning. The Knowledge Flow provides an alternative to the Explorer as a graphical front end to WEKA's core algorithms. The Knowledge Flow presents a data-flow inspired interface to WEKA. The user can select WEKA components from a palette, place them on a layout canvas and connect them together in order to form a knowledge flow for processing and analyzing data. At present, all of WEKA's classifiers, filters, clusterers, associators, loaders and savers are available in the Knowledge Flow along with some extra tools.
- **Workbench**: The Weka workbench is a set of tools for pre-processing data, classification, regression, clustering, association rules.
- **SimpleCLI:** Provides a simple command-line interface that allows direct execution of WEKA commands for operating systems that do not provide their own command line interface.

3.1 Weka Explorer



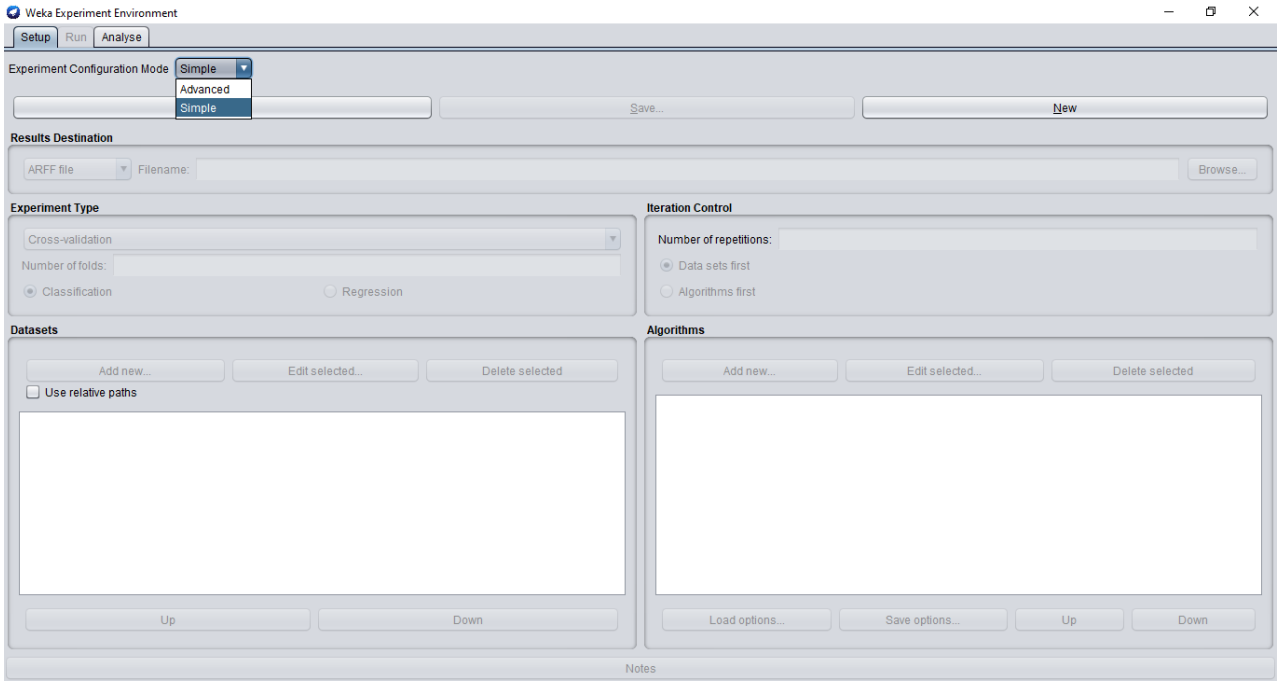
At the very top of the window, just below the title bar, is a row of tabs. When Explorer is first started only the first tab is active; the others are grayed out. This is because it is necessary to open (and potentially pre-process) a data set before starting to explore the data. The tabs are as follows:

1. **Preprocess**: Choose and modify the data being acted on.
2. **Classify**: Train & test learning schemes that classify or perform regression
3. **Cluster**: Learn clusters for the data.
4. **Associate**: Learn association rules for the data.
5. **Select Attributes**: Select the most relevant attributes in the data.
6. **Visualize**: View an interactive 2D plot of the data.

Once the tabs are active, clicking on them flicks between different screens, on which the respective actions can be performed. The bottom area of the window (including the status box, the log button, and the Weka bird) stays visible regardless of which section you are in. The Explorer can be easily extended with custom tabs.

3.2. Weka Experimenter

The Weka Experiment Environment enables the user to create, run, modify, and analyze experiments in a more convenient manner than is possible when processing the schemes individually. For example, the user can create an experiment that runs several schemes against a series of datasets and then analyze the results to determine if one of the schemes is (statistically) better than the other schemes.



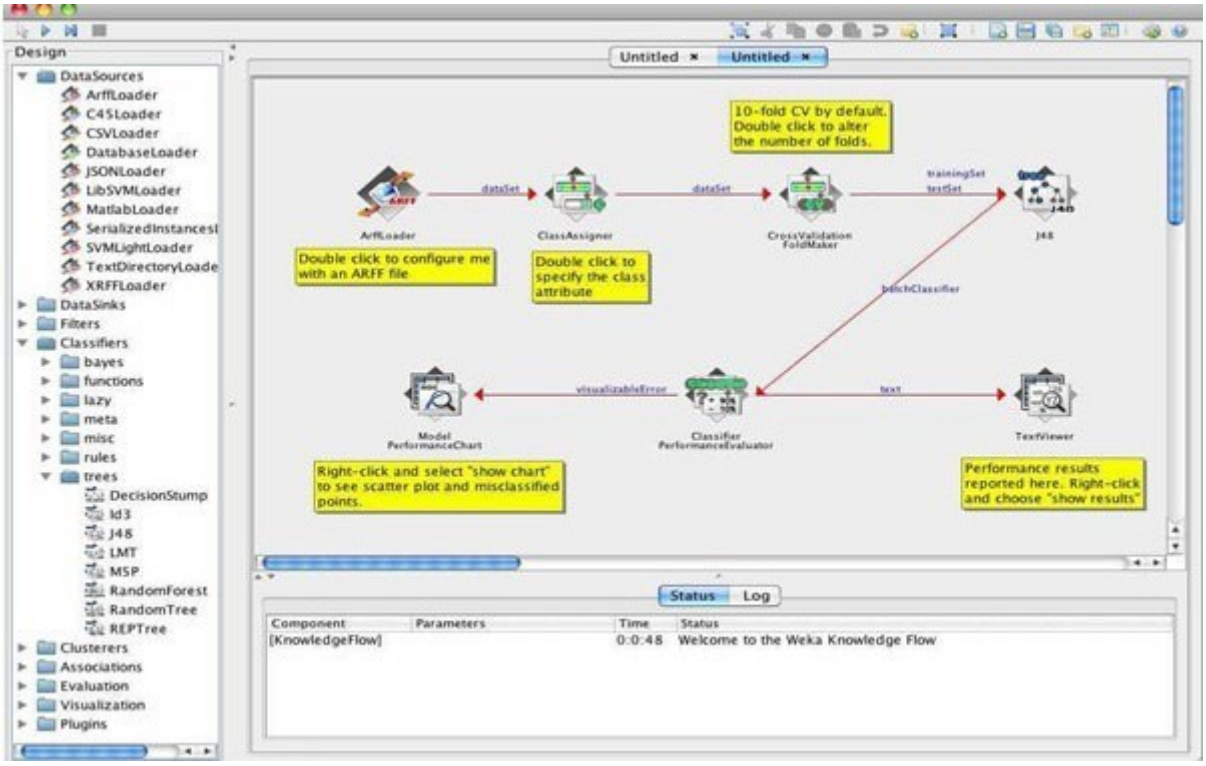
The Experiment Environment can be run from the command line using the Simple CLI. For example, the following commands could be typed into the CLI to run the OneR scheme on the Iris dataset using a basic train and test process. (Note that the commands would be typed on one line into the CLI.) While commands can be typed directly into the CLI, this technique is not particularly convenient and the experiments are not easy to modify. The Experimenter comes in two flavors', either with a simple interface that provides most of the functionality one needs for experiments, or with an interface **with full access to the Experimenter's capabilities. You can** choose between those two with the Experiment Configuration Mode radio buttons:

- Simple
- Advanced

Both setups allow you to setup standard experiments, that are run locally on a single machine, or remote experiments, which are distributed between several hosts. The distribution of experiments cuts down the time the experiments will take until completion, but on the other hand the setup takes more time. The next section covers the standard experiments (both simple and advanced), followed by the remote experiments and finally the analyzing of the results.

3.3. Knowledge Flow

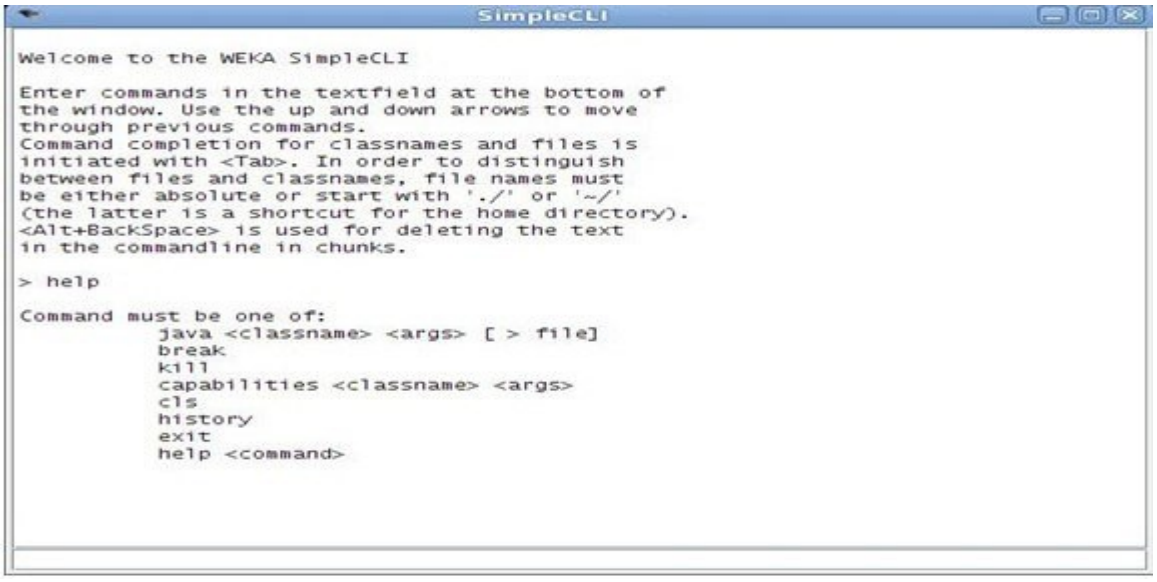
The Knowledge Flow provides an alternative to the Explorer as a graphical front end to **WEKA's core algorithms.** The Knowledge Flow presents a data-flow inspired interface to WEKA. The user can select WEKA components from a palette, place them on a layout canvas and connect them together in order to form a knowledge flow for processing and analyzing data. At present, all of **WEKA's classifiers, filters, clusterers, associators, loaders and savers a**re available in the Knowledge Flow along with some extra tools.



The Knowledge Flow can handle data either incrementally or in batches (the Explorer handles batch data only). Of course, learning from data incremen- tally requires a classifier that can be updated on an instance-by-instance basis. Currently in WEKA there are ten classifiers that can handle data incrementally.

3.4. Simple CLI

The Simple CLI provides full access to all Weka classes, i.e., classifiers, filters, clusterers, etc., but without the hassle of the CLASSPATH (it facilitates the one, with which Weka was started). It offers a simple Weka shell with separate command line and output.



4. *ARFF File Format*

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 are not the only format one can load, but all files that can be converted with Weka's "core converters". The following formats are currently supported:

- *ARFF*
- CVS files
- C4.5 file
- XRFF file

5. *Structure ARFF*

- ARFF –Attribute Relation File Format
- Structure *.ARFF:
    - @relation
    - @attribute
    - @data
    - % - comments line
    - ? – missing value

Syntax @relation

@relation <relation-name>

where <relation-name> is a string. The string must be quoted if the name includes spaces.

Example:

@relation golfWeatherMichigan_1988/02/10_14days

@relation 'golfWeatherMichigan 1988/02/10 14 days'

Syntax @attribute

@attribute <attribute-name> <datatype> The <datatype> can be:

1. numeric
2. nominal
3. string
4. date

Numeric and nominal attributes

1. Numeric:

@attribute <attribute-name> <numeric || integer || real>

@attribute salary *numeric*

@attribute salary *real*
@attribute age *integer*

2. Nominal:

@attribute <attribute-name> <nominal-specification>

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

String and date attributes

3. String – text of unlimited length:

@attribute <attribute-name> *string*

@attribute comments *string*

4. Date:

@attribute <attribute-name> *date* [<date-format>]

@ attribute timestamp *date* Default format:

ISO-8601 combined date and time format: yyyy-MM-d'T'HH:mm:ss

@ attribute timestamp *date "yyyy-MM-dd"*

Syntax @data

The @data declaration is a single line denoting the start of the data segment in the file. The format is:

@data

The instance data. Each instance is represented on a single line, with carriage returns denoting the end of the instance. Attribute values for each instance are delimited by commas. They must appear in the order that they were declared in the header section (i.e. the data corresponding to the nth @attribute declaration is always the nth field of the attribute)

ARFF file example 1:

```
% This is a relation about weather
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature real
@attribute humidity real
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
```

ARFF file example 2:

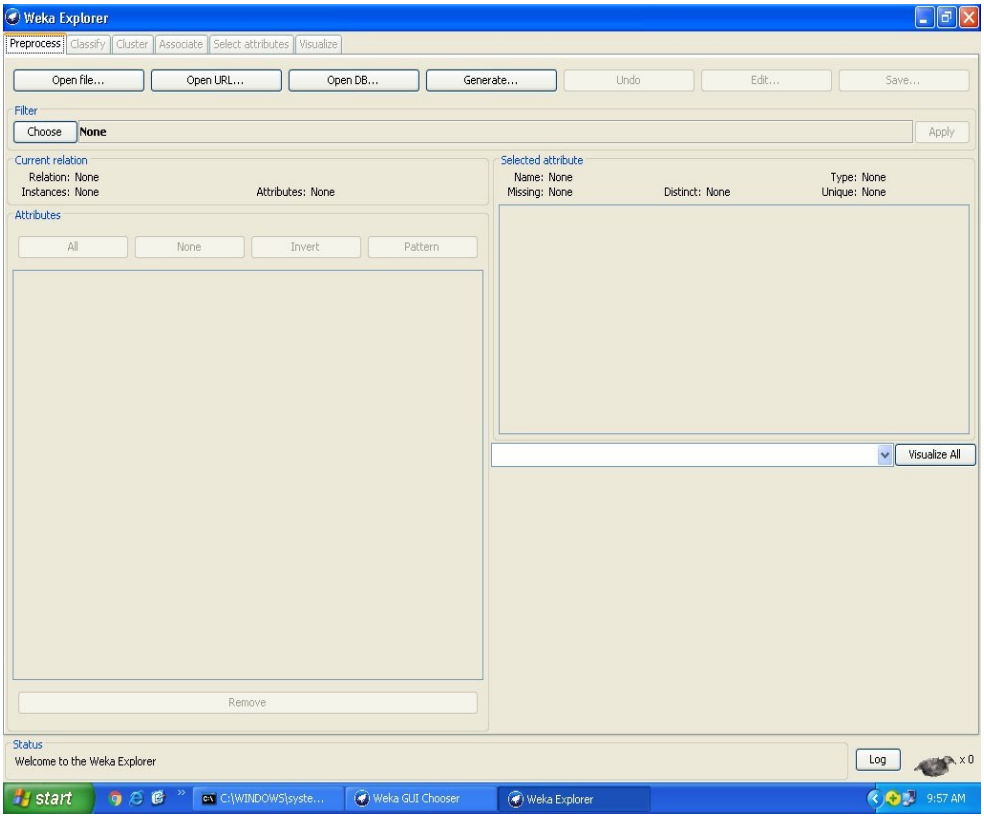
```
@relation nhanvien
@attribute hoten string
@attribute ngaysinh date "dd/MM/yy"
@attribute gioitinh {nam, nu}
@attribute hesoluong real

@data
'Nguyen Van A', 10/12/1957, nam, 1.34
'Tran Thi B', ?, nu, 1.5
```

6. Explore the available data sets in WEKA.

Steps for identifying data sets in WEKA:

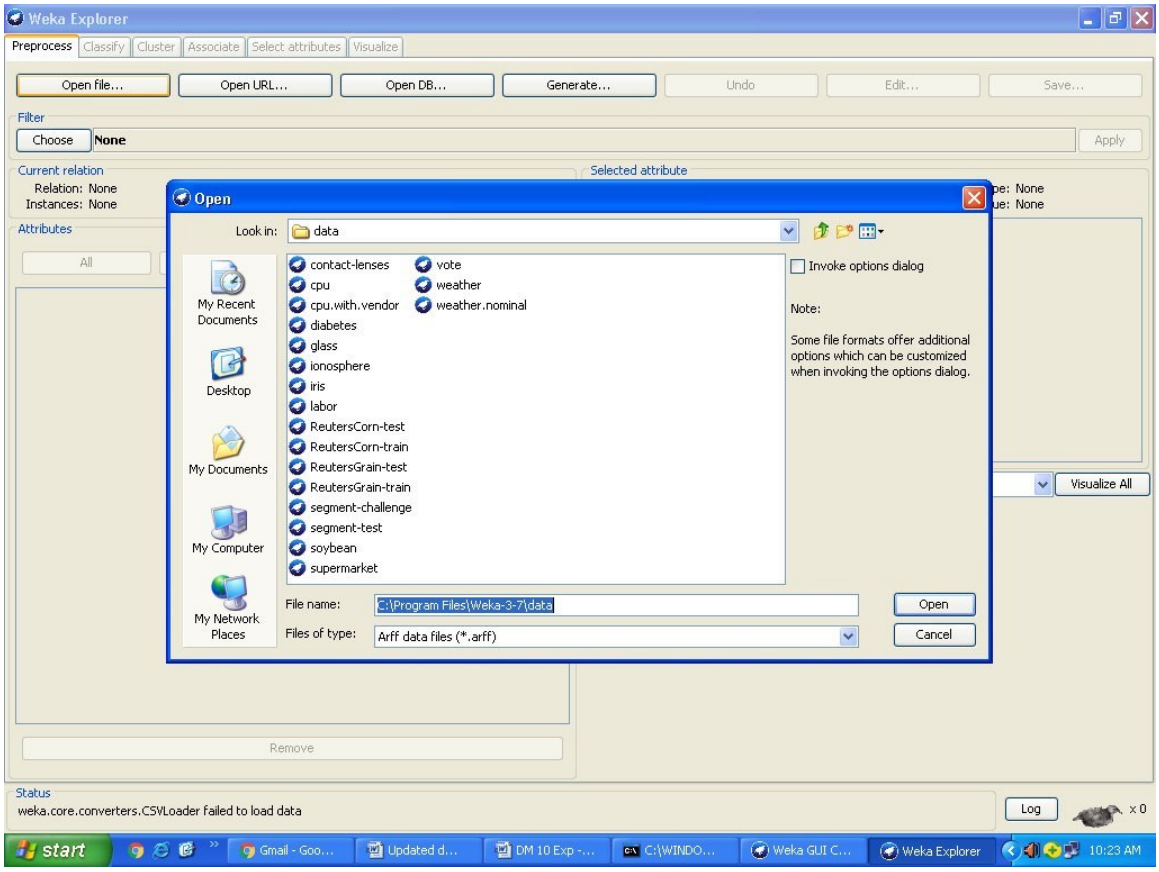
1. Open WEKA Tool.
2. Click on WEKA Explorer.



Current Relation: Once some data has been loaded, the Preprocess panel shows a variety of in**formation. The Current relation box (the "current relation" is the** currently loaded data, which can be interpreted as a single relational table in database terminology) has three entries:

- **Relation.** The name of the relation, as given in the file it was loaded from. Filters (described below) modify the name of a relation.
- **Instances.** The number of instances (data points/records) in the data.
- **Attributes.** The number of attributes (features) in the data.

3. Click on **Open file** button.
4. Choose WEKA folder in C drive.
5. Select and Click on data option button.



Sample Weka Data Sets

Below are some sample WEKA data sets, in arff format.

- contact-lens.arff
- cpu.arff
- cpu.with-vendor.arff
- diabetes.arff
- glass.arff
- ionospehre.arff
- iris.arff
- labor.arff
- ReutersCorn-train.arff
- ReutersCorn-test.arff
- ReutersGrain-train.arff
- ReutersGrain-test.arff
- segment-challenge.arff
- segment-test.arff

- soybean.arff
- supermarket.arff
- vote.arff
- weather.arff
- weather.nominal.arff

7. Exercises

Exercise 1: Load each dataset and exam dataset the following:

- What's the name of the relation?
- Number of records in each dataset
- How many instances does each dataset have?
- How many attributes?
- List attribute names and their types.
- Click on each attribute, observe, and tell:
    - What are the values that each attribute can have?
    - Missing: The number (and percentage) of instances in the data for which this attribute is missing (unspecified)
    - Distinct: The number of different values that the data contains for this attribute.
    - Unique: The number (and percentage) of instances in the data having a value for this attribute that no other instances have.
    - Show the basic statistics on the attributes for the categorical attributes the frequency of each attribute, or for the continuous attributes, show the value of min, max, mean, standard deviation and deviation etc.,
- Identify the class attribute (if any). Determine the number of records for each class
- Plot Histogram
- Visualize the data in various dimensions. What happens if the class attribute is left on the first column? Compare the visualizations of the 2 datasets (the one with class on the first column, before any manipulation, and the one with the class on the last column) as they appear on the right side of the Preprocessing panel: can you notice any differences?
- Examining data throught a viewer for editing

(Loading any 5 dataset)

Exercises 2: Create Dataset student .arff

no	age	income	student	credit-rating	buyspc
1	<30	high	no	fair	no
2	<30	high	no	excellent	no
3	30-40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	30-40	low	yes	excellent	yes
8	<30	medium	no	fair	no
9	<30	low	yes	fair	no
10	>40	medium	yes	fair	yes
11	<30	medium	yes	excellent	yes
12	30-40	medium	no	excellent	yes
13	30-40	high	yes	fair	yes
14	>40	medium	no	excellent	no

Exercises 3: Preprocessing with WEKA-Explorer on the dataset student.arff

- Loading the dataset into Explorer.
- Examming dataset with requirements in exercise 1
- What is the class value of instance number 8 in the student data?
- Preprocessing data by selecting or filtering attributes.
  - Removing an attribute
  - Saving the new working relation as an arff file
  - Discretizing the age attribute with the number of bins is 3
  - Saving the new working relation in a file called student-data-discretized.arff

Exercises 4: Preprocessing with WEKA-Explorer on Nominal data: the Weather (nominal) dataset.

- Loading the dataset into Explorer.
- How many instances are listed in the dataset? How many attributes? What's the attributes names? What type of attributes are these? Which one is the class?
- What is the function of the first column in the *Viewer* window? What is the class value of instance number 8 in the weather data?

- Proprocessing data
  - Removing an attribute from the dataset
  - Saving the new working relation as an arff file
  - Using the filter weka.unsupervised.instance.RemoveWithValues to remove all instances in which the humidity attribute has the value high.
  - Observing changes throught a viewer for editing

Exercises 5: Apply Preprocessing techniques to the dataset of Weather.

- Add a Climate attribute.
- Remove an attribute (select any attribute)
- Normalization: select numeric attributes to normalize

Exercises 6:

- Create a dataset Employee as:

ID	Name	Salary	Experience	Department
101	Mark	10000	4	Human Resources
102	Brian	15000	5	Sales
103	Alan	12000	3	Legal
104	Tony	13000	3	Retail
105	Agatha	16000	6	Sales
106	Lana	15000	5	Accounting
107	Heather	12000	3	Accounting
108	Ben	11000	5	Sales
109	Caitlyn	12000	3	Retail
110	Gibbs	11000	5	Retail
111	Anderson	12000	3	Sales
112	Michael	13000	4	Retail
113	David	14000	5	Sales
114	Jacob	12000	3	Support
115	John	13000	3	Business Development
116	Leonardo	16000	6	Business Development
117	Matthew	14000	5	Human Resources

- Apply Preprocessing techniques to this dataset.
  - Add a phone number attribute.
  - Remove an attribute (select any attribute)
  - Normalization: select numeric attributes to normalize

Exercises 7: Download and Exam dataset horse-colic

- Data: http://archive.ics.uci.edu/ml/machine-learning-databases/horse-colic/horsecolic.data
- Descriptions: http://archive.ics.uci.edu/ml/machine-learning-databases/horse-colic/horsecolic.names (*)

Lab 3-4-5	ASSOCIATION RULES MINING

APPLICATION OF DATA MINING

- Data mining can typically be used with transactional databases (for ex. in shopping cart analysis)
- Aim can be to build association rules about the shopping events
- Based on **item sets**, such as
  {milk, cocoa powder} 2-itemset
  {milk, corn flakes, bread} 3-itemset

ASSOCIATION RULES

- Items that occur often together can be associated to each other
- These together occuring items form a **frequent itemset**
- Conclusions based on the frequent itemsets form **association rules**
- For ex. {milk, cocoa powder} can bring a rule *cocoa powder*^{è}*milk*

SUPPORT AND CONFIDENCE

- If confidence gets a value of 100 % the rule is an **exact rule**
- Even if confidence reaches high values the rule is not useful unless the support value is high as well
- Rules that have both high confidence and support are called **strong rules**
- Some competing alternative approaches can generate useful rules even with low support values

GENERATING ASSOCIATION RULES

- Usually consists of two sub problems:
  1) Finding frequent itemsets whose occurrences exceed a predefined minimum support threshold
  2) Deriving association rules from those frequent itemsets (with the constrains of minimum confidence threshold)
- These two sub problems are solved iteratively until new rules no more emerge
- The second sub problem is quite straight- forward and most of the research focus is on the first sub problem.

USE OF APRIORI ALGORITHM

- Initial information: transactional database D and user-defined numeric minimum support threshold *min_sup*
- Algorithm uses knowledge from previous iteration phase to produce frequent itemsets
- This is reflected in the Latin origin of the name that means ‖from what comes before.

CREATING FREQUENT SETS

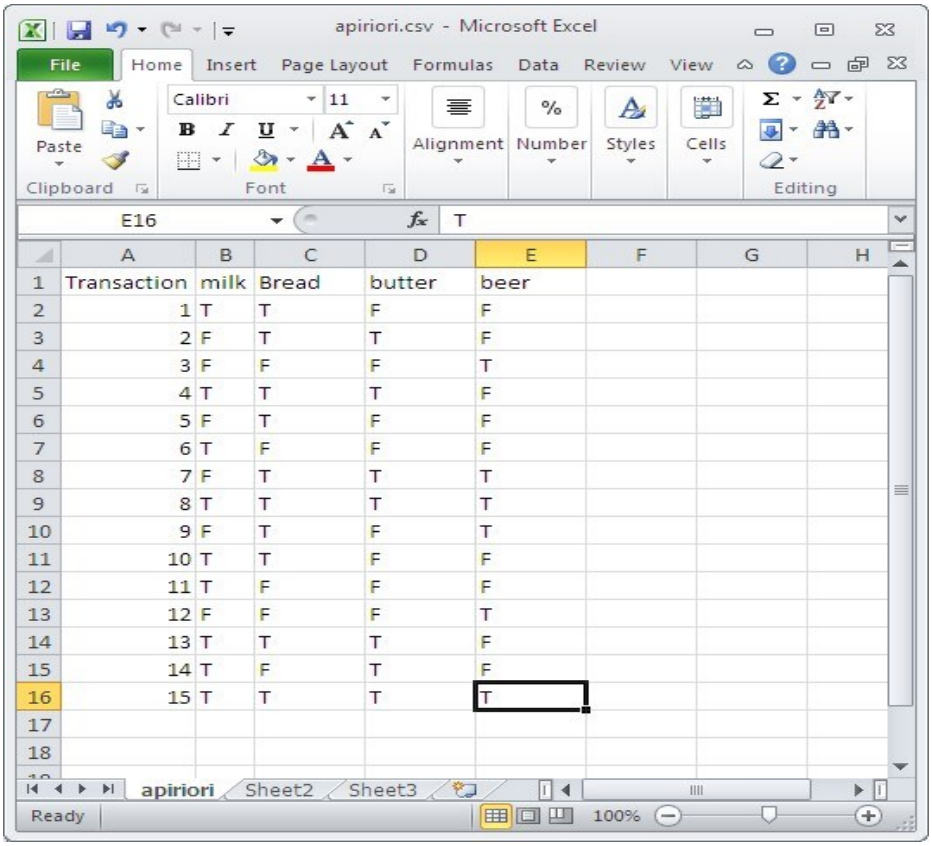
- Let's define:$C_k$as a candidate itemset of size kL_k as a frequent itemset of size k
- Main steps of iteration are:
    1) Find frequent set L_{k-1}
    2) Join step: C_k is generated by joining L_{k-1} with itself (Cartesian product Lk-1 x Lk-1)
    3) Prune step (apriori property): Any (k − 1) size itemset that is not frequent cannot be a subset of a frequent k size itemset, hence should be removed 4) Frequent set L_k has been achieved.
- Algorithm uses breadth-first search and a hash tree structure to make candidate itemsets efficiently
- Then occurrence frequency for each candidate itemset is counted
- Those candidate itemsets that have higher frequency than minimum support threshold are qualified to be frequent itemsets.

APRIORI ALGORITHM IN PSEUDOCODE

L_1= {frequent items};
for(k= 2; L_{k-1}!=Ø; k++) **do begin**
    C_k= candidates generated from L_{k-1} (that is: Cartesian product L_{k-1} x L_{k-1} and eliminating any k-1 size itemset that is not frequent);
    for each transaction t in database **do** increment the count of all candidates in C_k that are contained in t
        L_k = candidates in C_k with *min_sup*
    end
return $_kL_k$;

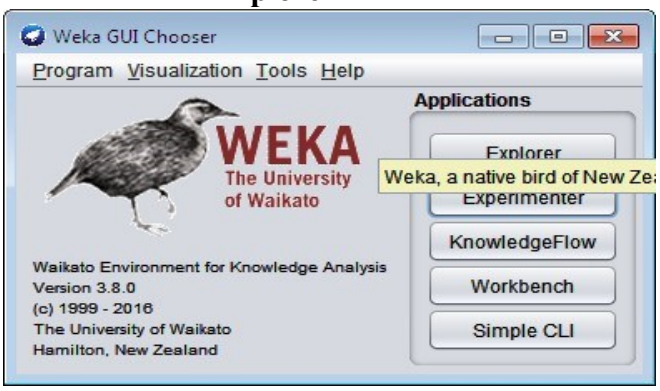
Exercise 1: Illustrate the Apriori algorithm in Weka to find association rules from the transaction database.

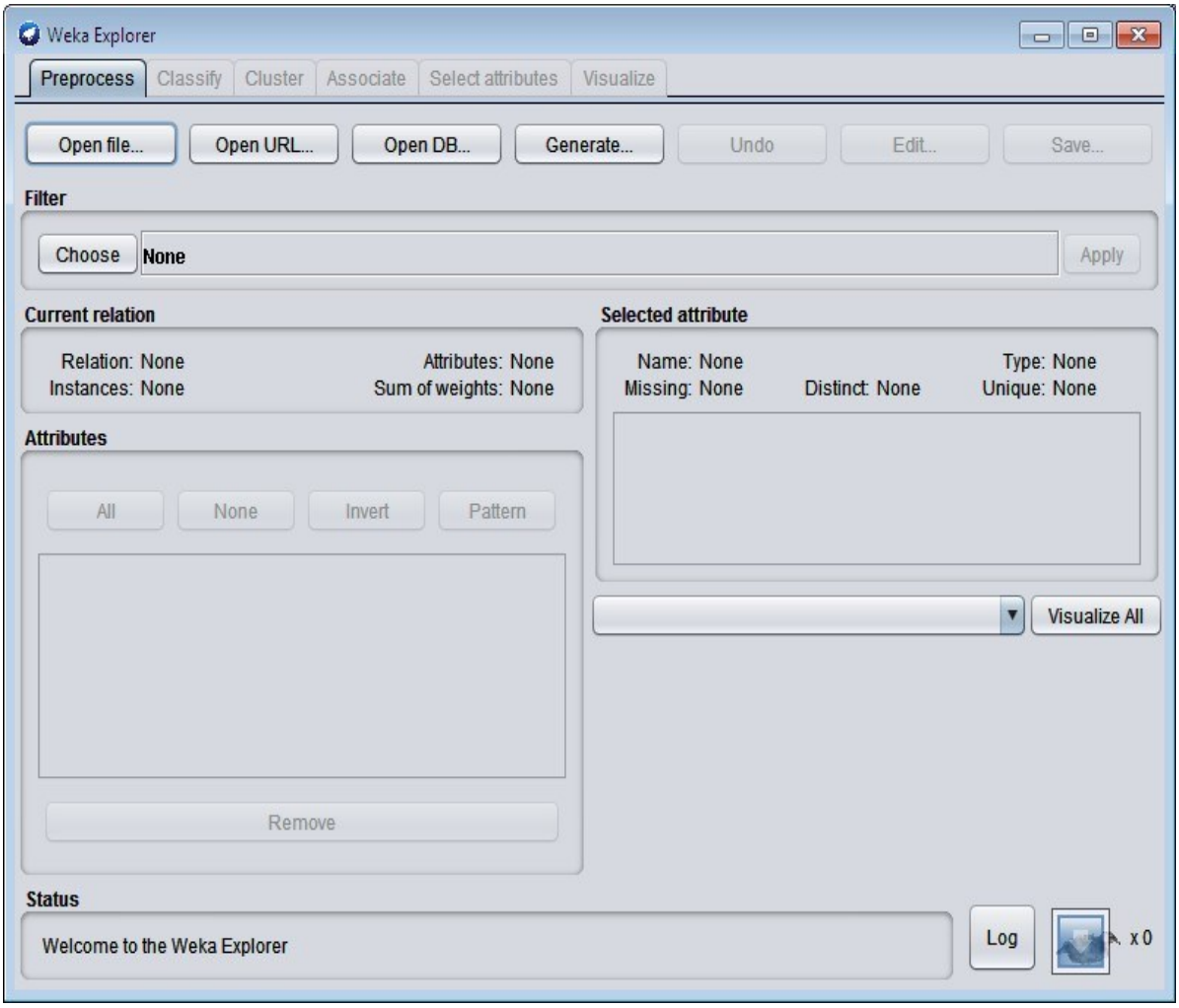
1. Create the dataset **apriori.csv** as:



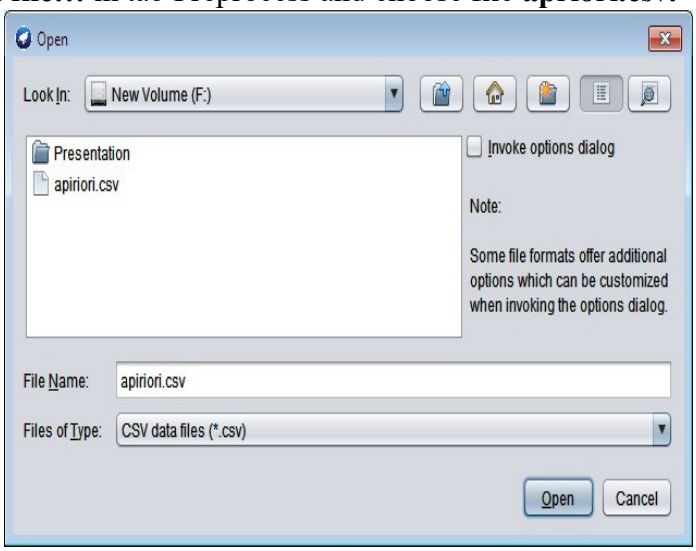
	A	B	C	D	E	F	G	H
1	Transaction	milk	Bread	butter	beer			
2	1	T	T	F	F			
3	2	F	T	T	F			
4	3	F	F	F	T			
5	4	T	T	T	F			
6	5	F	T	F	F			
7	6	T	F	F	F			
8	7	F	T	T	T			
9	8	T	T	T	T			
10	9	F	T	F	T			
11	10	T	T	F	F			
12	11	T	F	F	F			
13	12	F	F	F	T			
14	13	T	T	T	F			
15	14	T	F	T	F			
16	15	T	T	T	T			
17								
18								

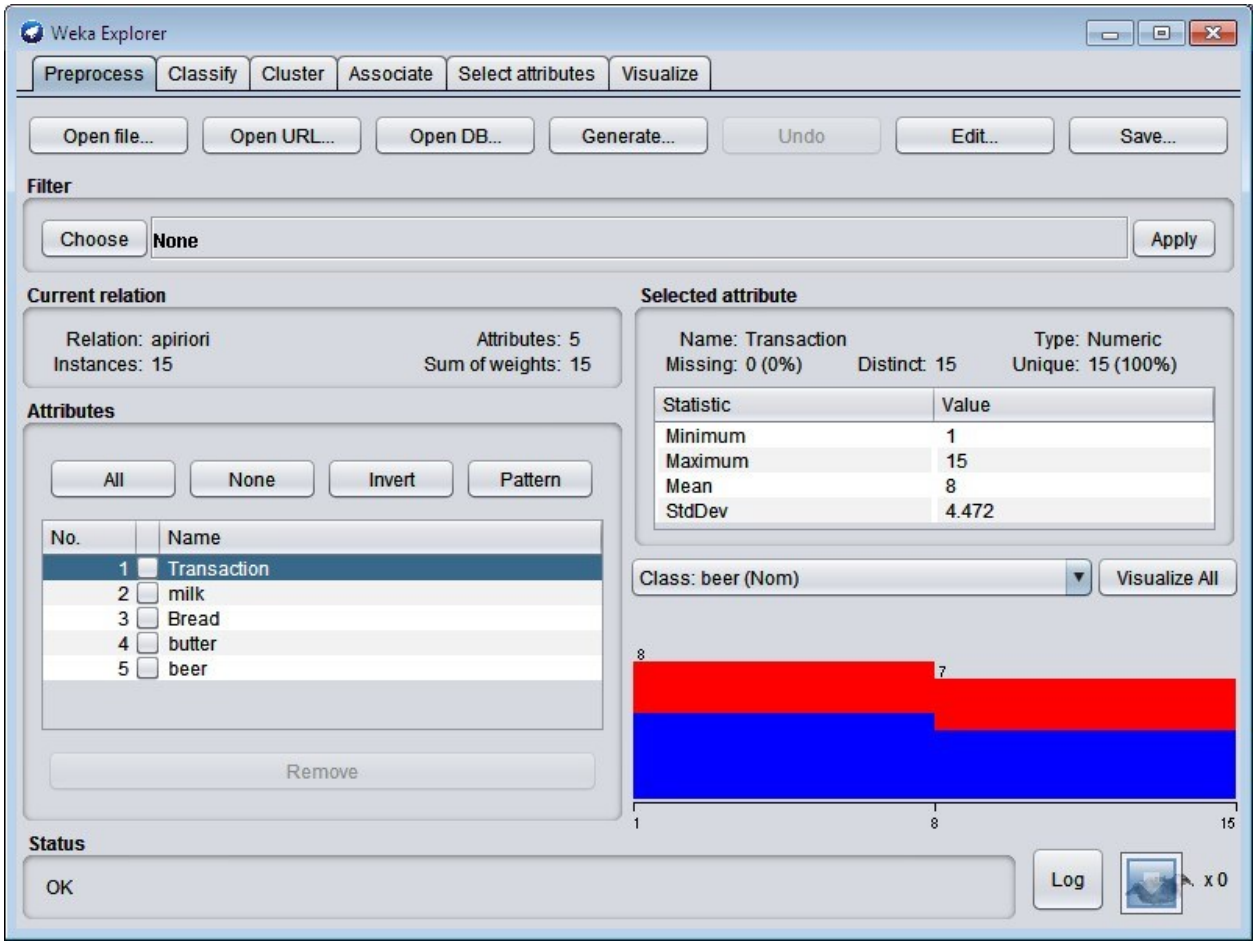
2. Open **weka tool** and click on button **Explorer**.



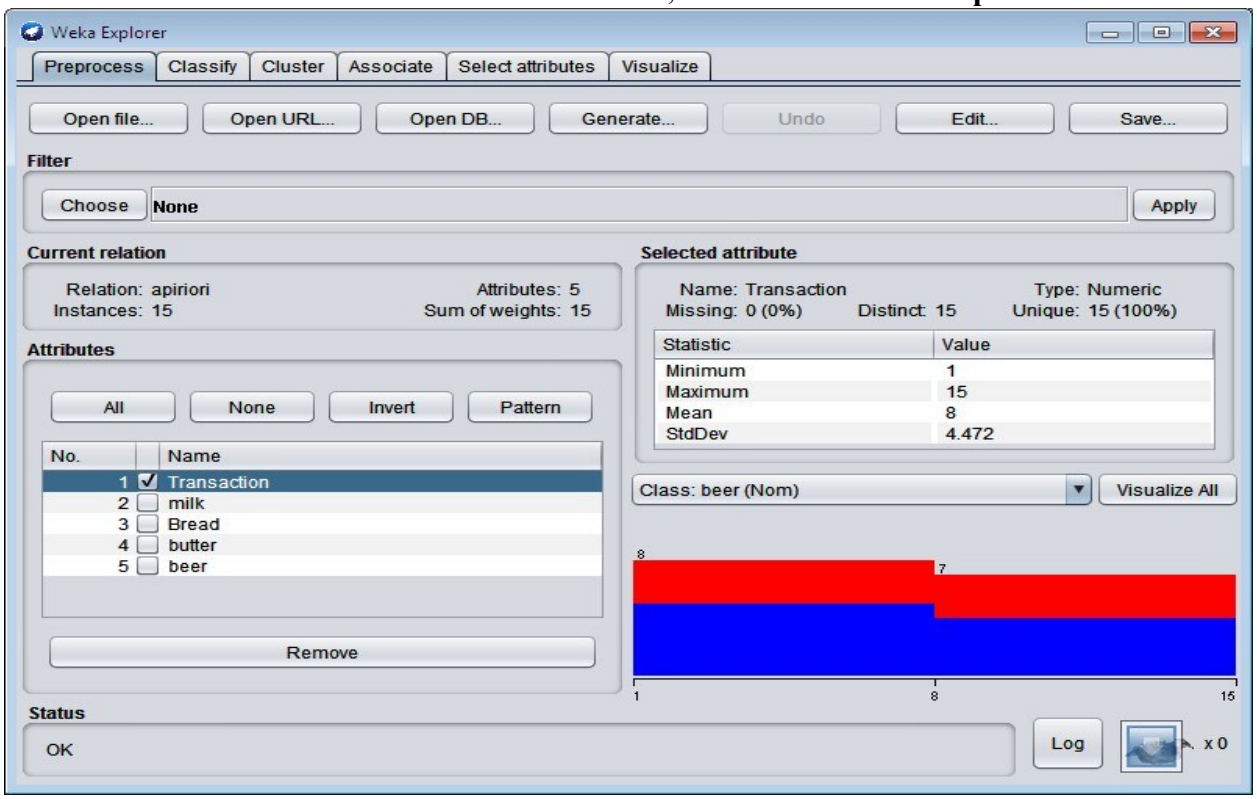


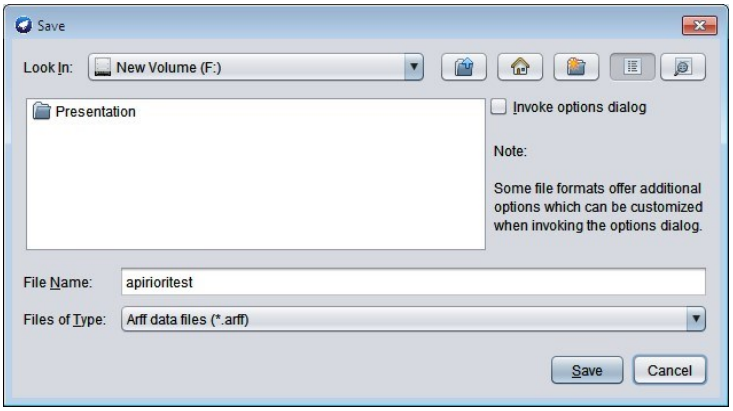
Click on button **open file…** in tab Preprocess and choose file **apriori.csv.**



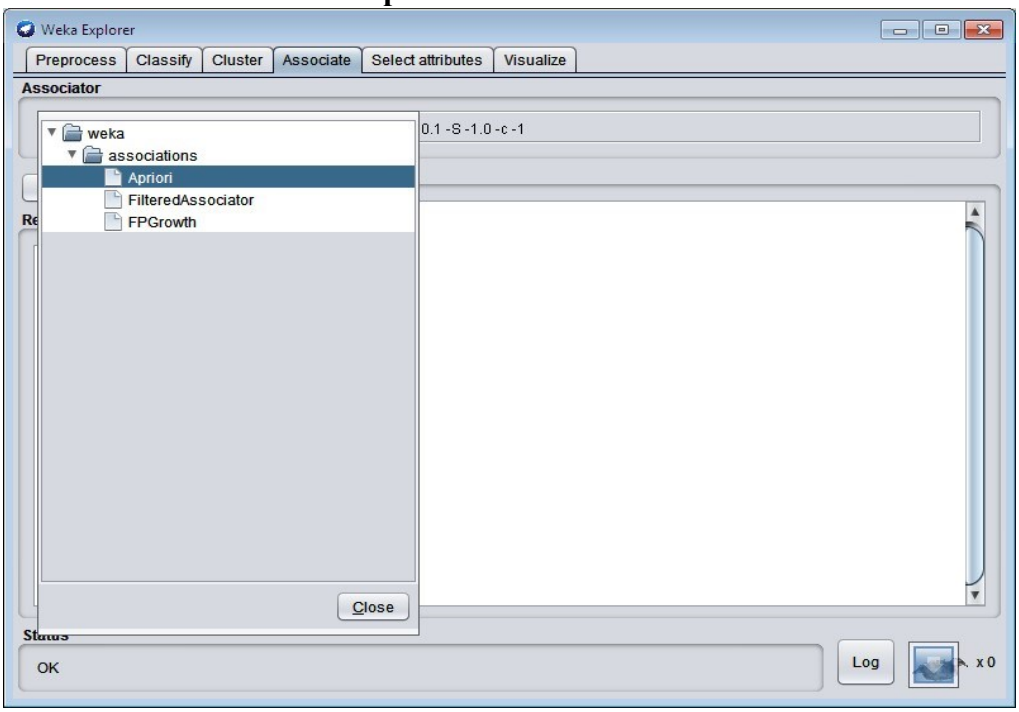


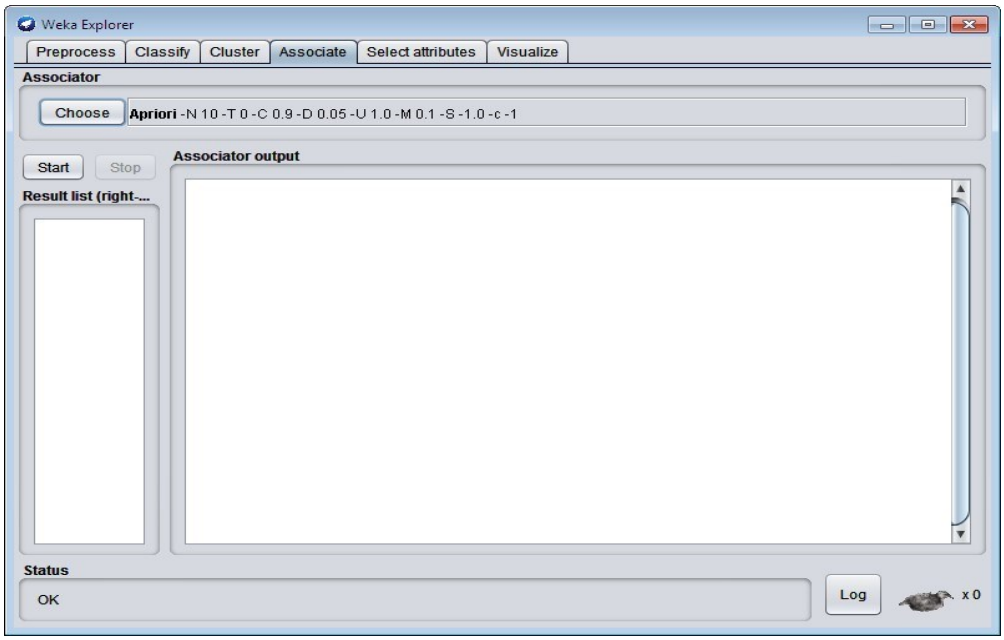
Choose field **Transaction** and click button Remove, after that save as file **aprioritest.arff**



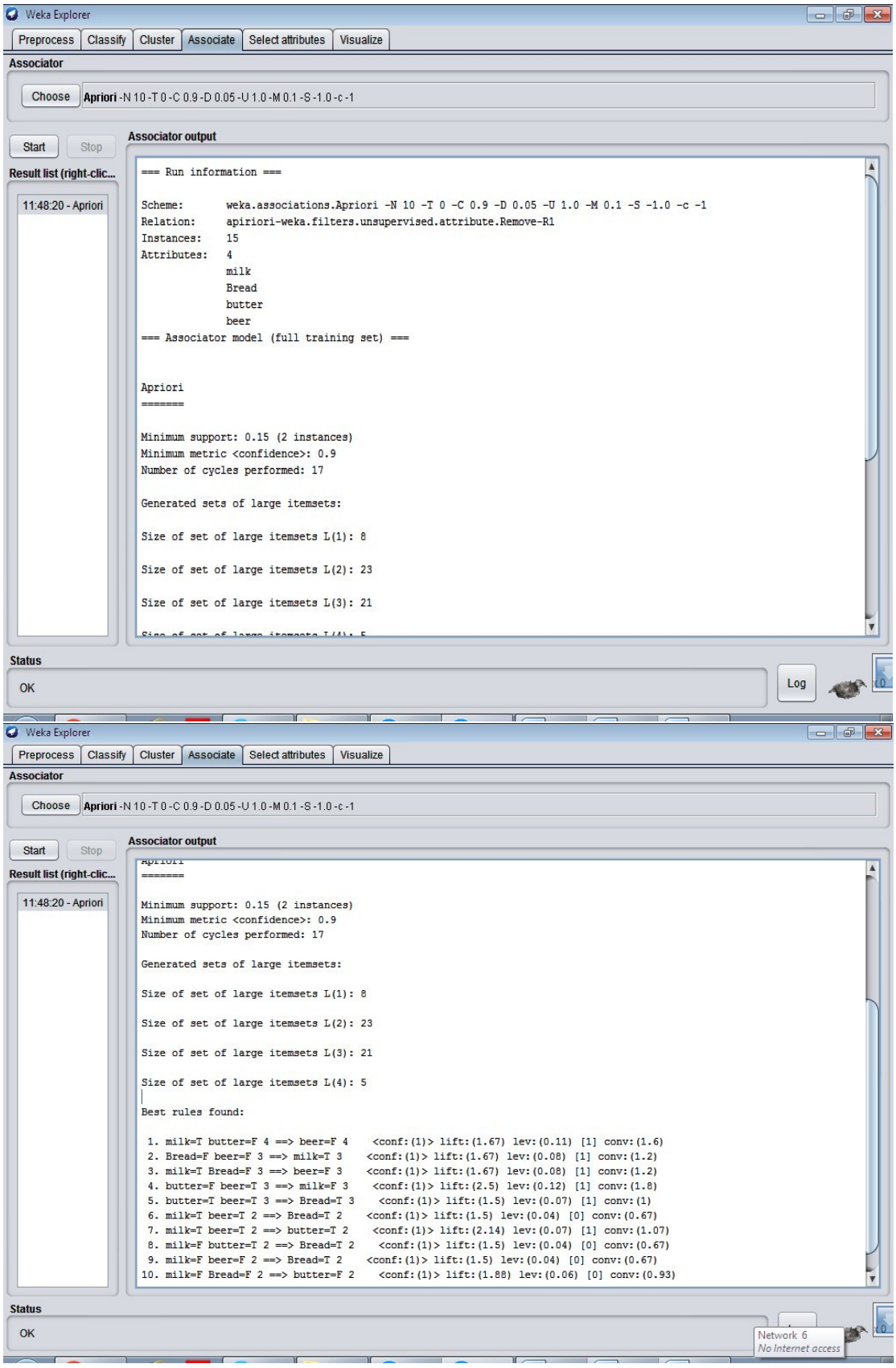


Change to the **Associate** tab – select **Apriori** and click the **Start** button.





The output field:



The first part of the output (‘Run information’) describes the options that have been set and the data set used. The rules that have been generated are listed at the end of the output.

- Analyse the association rules mined from the data set. What are their confidence and support values? Examine the number of large itemsets.
- Try changing different values for the following parameters \lowerBoundMinSupport" (min threshold for support), \minMetric" (min threshold for confidence). As you change these parameter values what do you notice about the rules that are found by the associator? Note that the parameter \numRules" limits the maximum number of rules that the associator looks for, you can try changing this value.

Exercise 2: Find association rules in the ***weather.nominal*** dataset by Weka explorer

a. Load the data in ***Weka Explorer***. Select the ***Associate tab.*** How many different associations rule mining algorithms are available?

b. Choose ***Apriori*** algorithm with the following parameters (which you can select by clicking on the chosen algorithm: support threshold = 15% (lowerBoundMinSupport = 0.15), confidence threshold = 90% (metricType = confidence, minMetric = 0.9), number of rules = 50 (numRules = 50). After starting the algorithm, how many rules do you find? Could you use the regular ***weather*** dataset to get the results? Explain why.

c. Paste a screenshot of the ***Explorer*** window showing at least the first 20 rules.

d. Based on the output, what is the support for this item set?

outlook = rainy humidity = normal windy = FALSE play = yes

e. ***Apriori*** algorithm generates association rules from frequent itemsets. How many itemsets of size 4 were found? Which rule(s) have been generated from itemset of size 4 (temperature=mild, windy=false, play=yes, outlook=rainy)? List their numbers in the list of rules.

f. What is the total number of possible rules for the weather data for each combination of values in the following table?

Minimun confidence	Minimum support	Number of rules
0.9	0.3	
0.9	0.2	
0.9	0.1	
0.8	0.3	
0.8	0.2	
0.8	0.1	
0.7	0.3	
0.7	0.2	
0.7	0.1	

Exercise 3: The 'database' below has four transactions:

Trans_id	Itemlist
T1	K, A, D, B
T2	D, A C, E, B
T3	C, A, B, E
T4	B, A, D

a. Use manually the Apriori algorithm to find all frequent itemsets and all association rules with minimum support is 60% and the minimum confidence is 80%.
b. What association rules can be found in this set using WEKA, if the minimum support is 60% and the minimum confidence is 80%.
c. Compare the association rules output from Apriori in weka and manually.

Repair dataset:
ARFF:
@relation exercise
@attribute A {TRUE, FALSE}
…
@data
TRUE,TRUE,FALSE,TRUE,FALSE,TRUE
… …
Or CSV:
A,B,C,D,E,K
TRUE,TRUE,FALSE,TRUE,FALSE,TRUE
… …

Exercise 4:

- Download the zoo dataset and load it into Weka.
- Examine the attributes. Are all attributes nominal?
- Preprocess data to make sure the dataset can be used directly with Apriori algorithm.
- Use the Apriori algorithm with the default parameters. Record the generated rules.
- Vary the number of rules generated (20, 30, ...). Record how many rules have to be generated before generating a rule containing *type=mammal*.
- Vary the maximum support until a rule containing *type=mammal* is the top rule generated. Record the maximum support needed.
- Select one generated rule that was interesting to you. Why was it interesting? What does it mean? Check its confidence and support – are they high enough?

Exercise 5: Using Weka KnowledgeFlow to find association rules for the above exercises.

LAB 6 -7: CLASSIFICATION DATA

Exercise 1: Using Weka Explorer to construct decision tree for the weather.nominal.arff with the J48 algorithm (or the other classification model) and classify it.

1. Launch the Weka tool and activate the **Explorer** environment.
2. Open the "**weather.nominal**" dataset
3. Go to the **Classify** tab. Select the **J48** classifier. Choose the "Cross-validation" (10 folds) test mode. Run the classifier and observe the results shown in the "Classifier output" window.
- How many instances are incorrectly classified?
- What is the MAE (mean absolute error) made by the classifier?
- What can you infer from the information shown in the Confusion Matrix?
- Visualize the classifier errors. In the plot, how can you differentiate between the correctly and incorrectly classified instances?
- How can you save the learned classifier to a file?
- How can you load a learned classifier from a file?

4. Choose the "Percentage split" (66% for training) test mode. Run the **J48** classifier and observe the results shown in the "Classifier output" window.
- How many instances are incorrectly classified? Why this number is smaller than that observed in the previous experiment (i.e., using the cross-validation test mode)?
- What is the MAE made by the classifier?
- Visualize the classifier errors to see the detailed information.

5. Choose the "Use training set" (66% for training) test mode. Run the **J48** classifier and observe the results shown in the "Classifier output" window.
- How many instances are incorrectly classified? Why this number is smaller than that observed in the previous experiment (i.e., using the cross-validation test mode)?
- What is the MAE made by the classifier?
- Visualize the classifier errors to see the detailed information.
6. Compare test modes provided in the J48 classifiers and fill-in the following table:

Test option	Kappa statistic	Mean absolute error	Root mean squared error	Relative absolute error	Root relative squared error
Use training set					
Cross-validation					
Percenttagesplit					

7. Compare the performance of other classification approaches in this dataset:
- J48
- OneR
- Naive Bayes

Exercise 2:

Let's assume that we have collected the following data set of users who decided to buy a computer and others who decided not. Each user record (i.e., example) is represented by the 5 attributes.
- Age, with the possible values {Young, Medium, Old}.
- Income, with the possible values {Low, Medium, High}.
- Student, with the possible values {Yes, No}.
- Credit_Rating, with the possible values {Fair, Excellent}.
- Buy_Computer – the classification attribute, with the possible values {Yes, No}.

UserID Age Income Student Credit_Rating Buy_Computer
1 Young High No Fair No
2 Young High No Excellent No
3 Medium High No Fair Yes
4 Old Medium No Fair Yes
5 Old Low Yes Fair Yes
6 Old Low Yes Excellent No
7 Medium Low Yes Excellent Yes
8 Young Medium No Fair No
9 Young Low Yes Fair Yes
10 Old Medium Yes Fair Yes
11 Young Medium Yes Excellent Yes
12 Medium Medium No Excellent Yes
13 Medium High Yes Fair Yes
14 Old Medium No Excellent No
15 Medium Medium Yes Fair No
16 Medium Medium Yes Excellent Yes
17 Young Low Yes Excellent Yes
18 Old High No Fair No
19 Old Low No Excellent No
20 Young Medium Yes Excellent Yes

We want to predict, for each of the following users, if s/he will buy a computer or not.
- User #21. A young student with medium income and fair credit rating.
- User #22. A young non-student with low income and fair credit rating.
- User #23. A medium student with high income and excellent credit rating.
- User #24. An old non-student with high income and excellent credit rating.

Requirement:
- Create the dataset containing 20 examples (i.e., Users #1-20) into the ARFF format and save it in the "buy_comp.arff" file.
- For each user in the set of Users #21-24, set the values of the Buy_Computer attribute by the predictions computed manually. Create the data of these four users into the ARFF format, and save it in the "buy_comp_extra.arff" file.
- Use the Weka Explorer tool Open the "buy_comp" dataset (i.e., saved in the "buy_comp.arff" file).
- Go to the "Classify" tab. Select the **J48** classifier **(or the other classification model)**. Choose "Percentage split" (66% for training) test mode. Run the classifier and observe the results shown in the "Classifier output" window.
- How many instances used for the training? How many for the test?
- Does the test set currently used include the four instances of Users #21-24?
- How many instances are incorrectly classified?
- What is the MAE (mean absolute error) made by the learned DT?
- What can you infer from the information shown in the Confusion Matrix?
- Visualize the errors made by the learned DT. In the plot, how can you differentiate between the correctly and incorrectly classified instances? In the plot, how can you see the detailed information of an incorrectly classified instance?
- How can you save the learned DT to a file?
- How can you visualize the structure of the learned DT?
- In the "Test options" panel select the "Supplied test set" option. Activate the nearby "Set..." button and locate the "buy_comp_extra.arff" file. Run the classifier and observe the results shown in the "Classifier output" window.

- How many instances used for the training? How many for the test?
- Does the test set currently used include the four examples (i.e., Users #21-24)?
- In the "Classifier output" window, where you can find the information that says for which of the four users (i.e., Users #21-24) the learned DT predicts correctly and for which others it predicts incorrectly?
- What is the MAE (mean absolute error) made by the learned DT?

Exercise 3: Using Weka Knowledge Flow to construct a decision tree for the weather.nominal.arff with the J-48 algorithm (or the other classification model) for the above exercises

Exercise 4: Using Java programming language to construct a decision tree with the J-48 algorithm (or the other classification model) for the above exercises

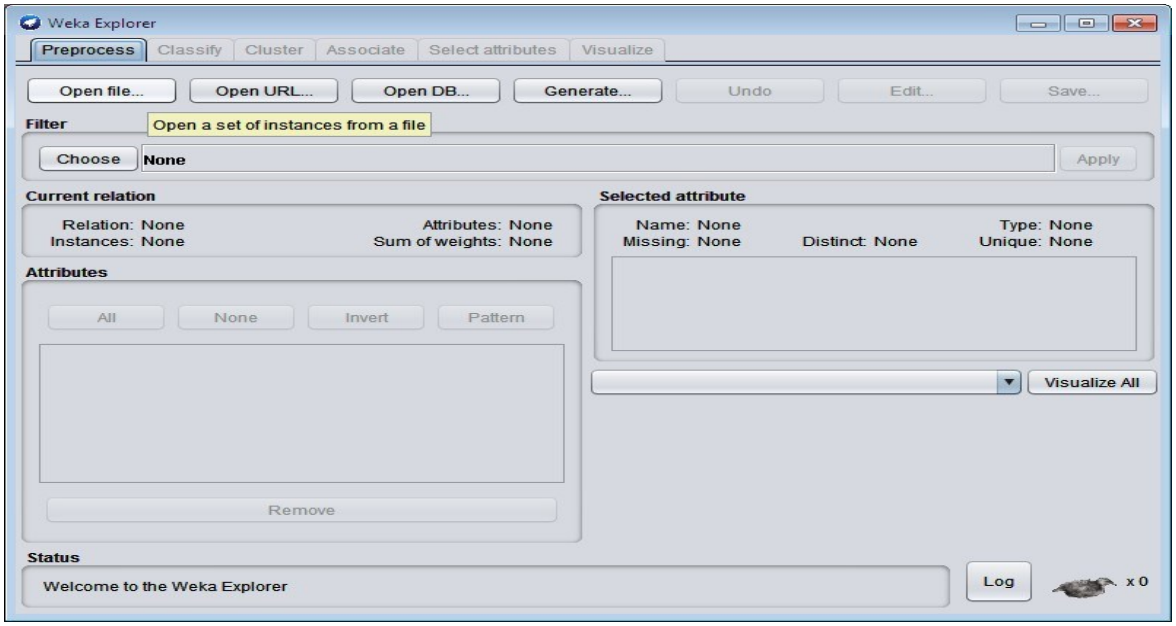
LAB 8 -9: CLUSTERING DATA

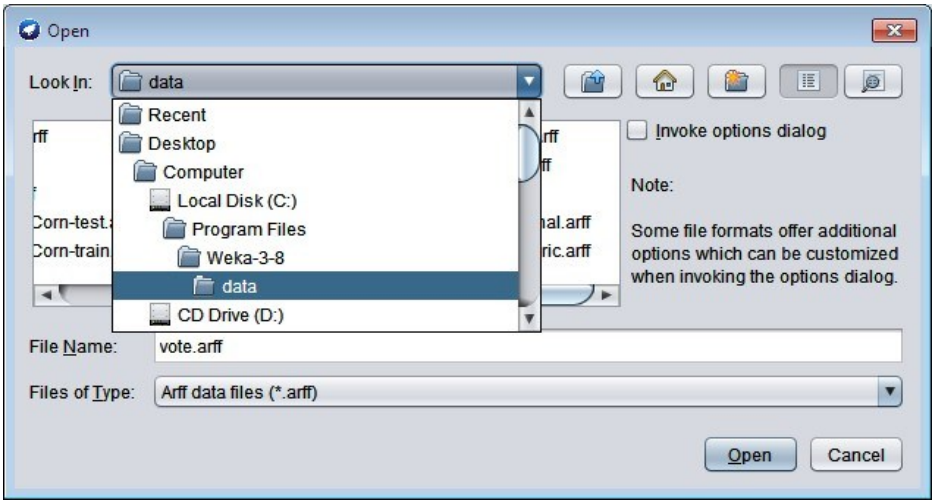
Exercise 1: This experiment illustrates the use of simple k-mean clustering with Weka. The sample data set used for this example is based on the vote.arff data set. This document assumes that appropriate pre-processing has been performed.

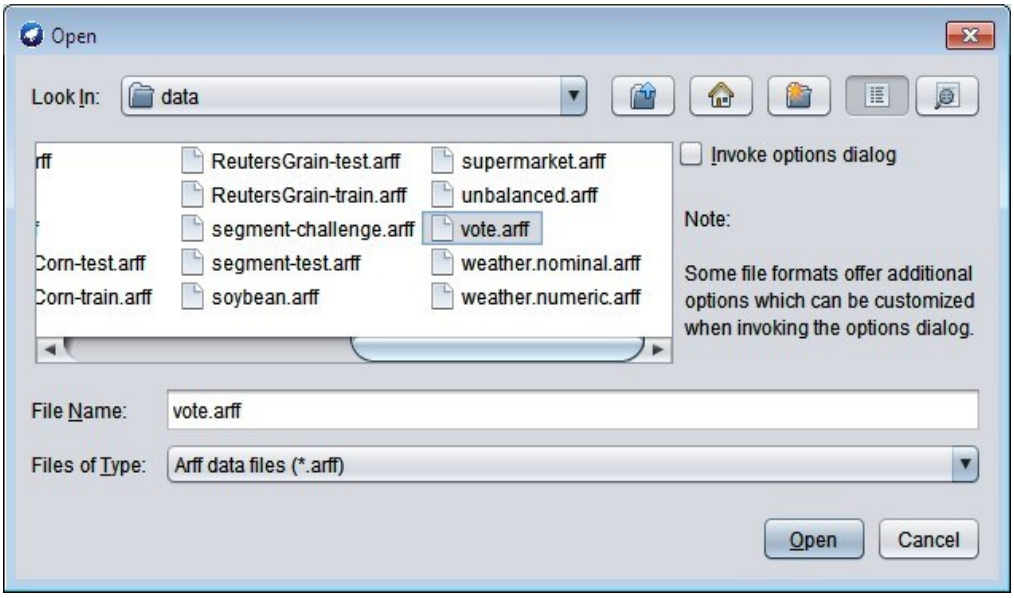
I. Using Weka Explore

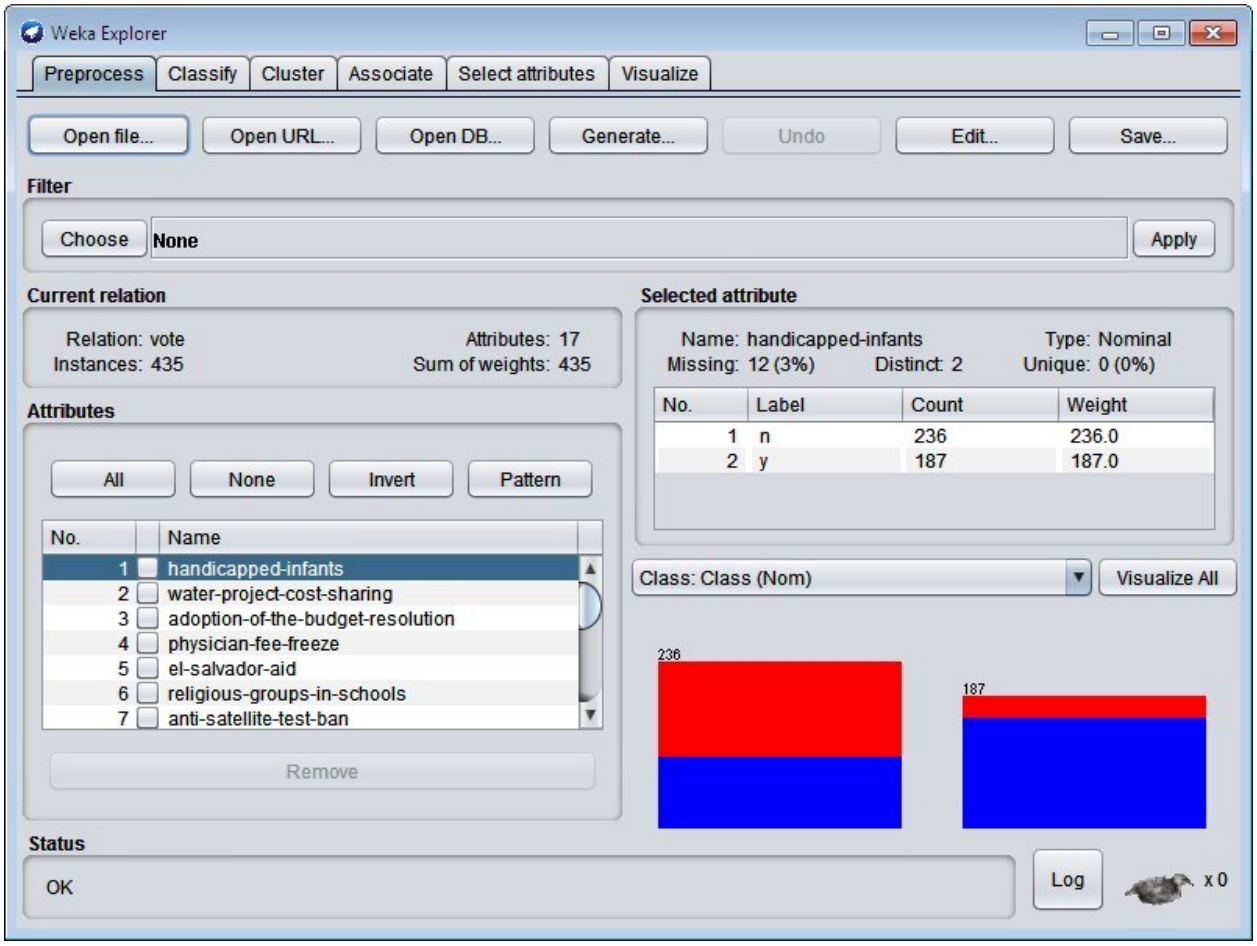
1. Open **weka** tool and click **Explorer.**

2. click**Open file**…in Preprocess tab.- choose **vote.arff**

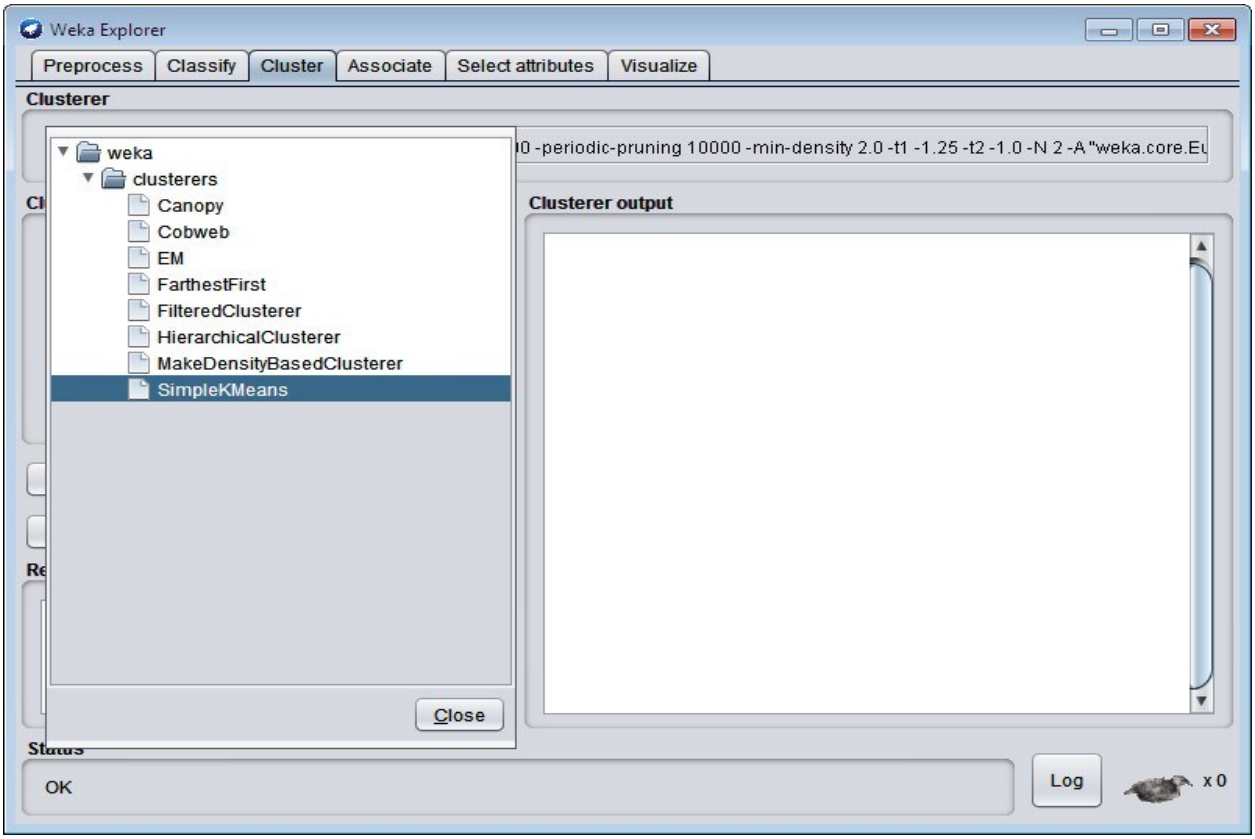


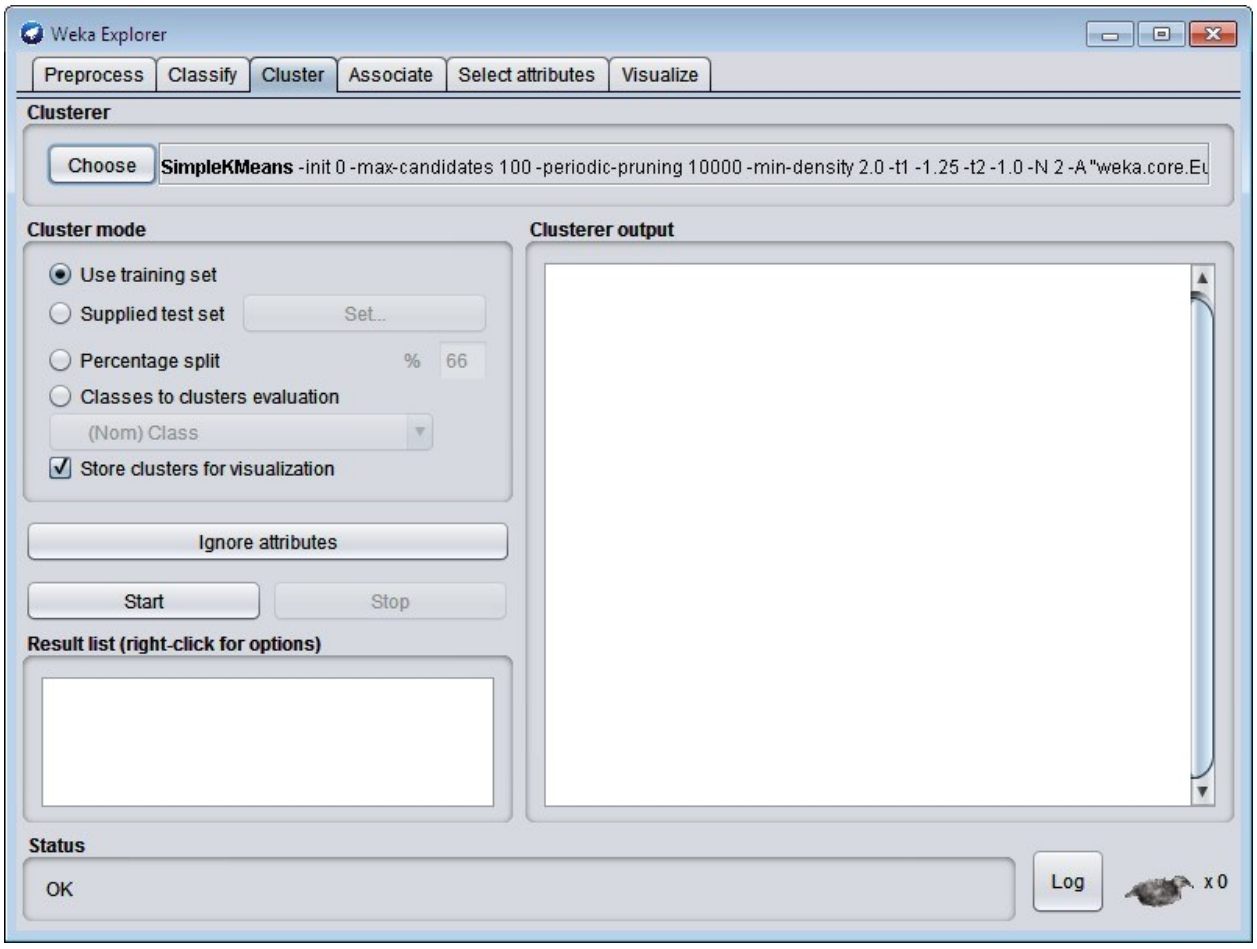






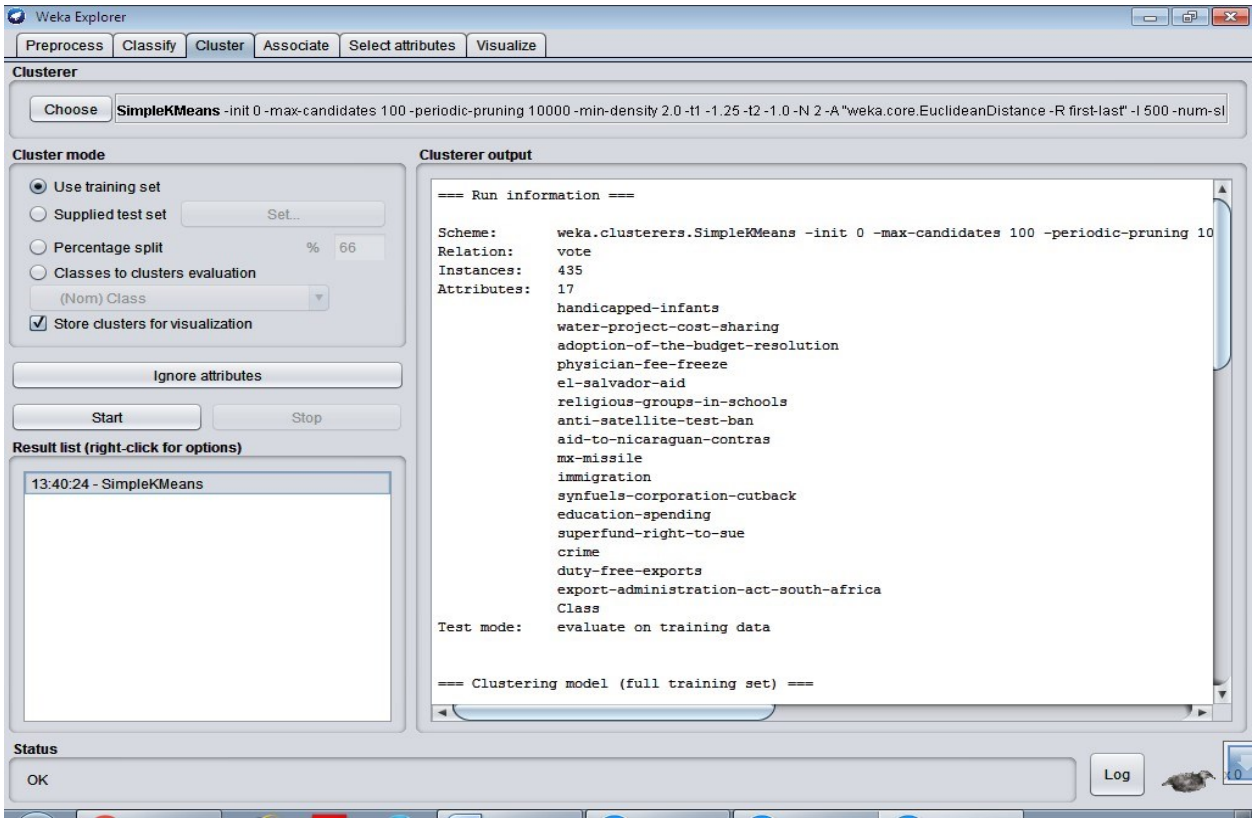
Choose **cluster** tab – click **choose** button – choose **SimpleKmeans.**

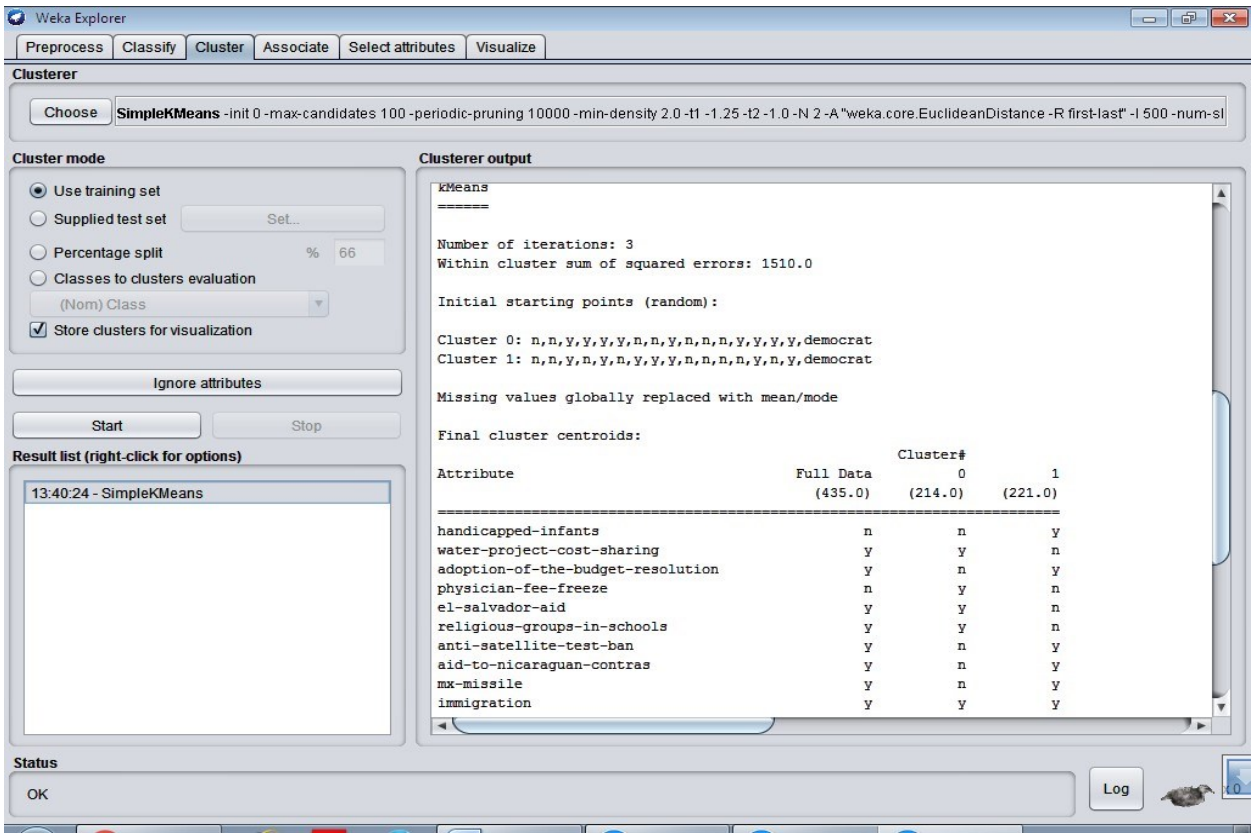


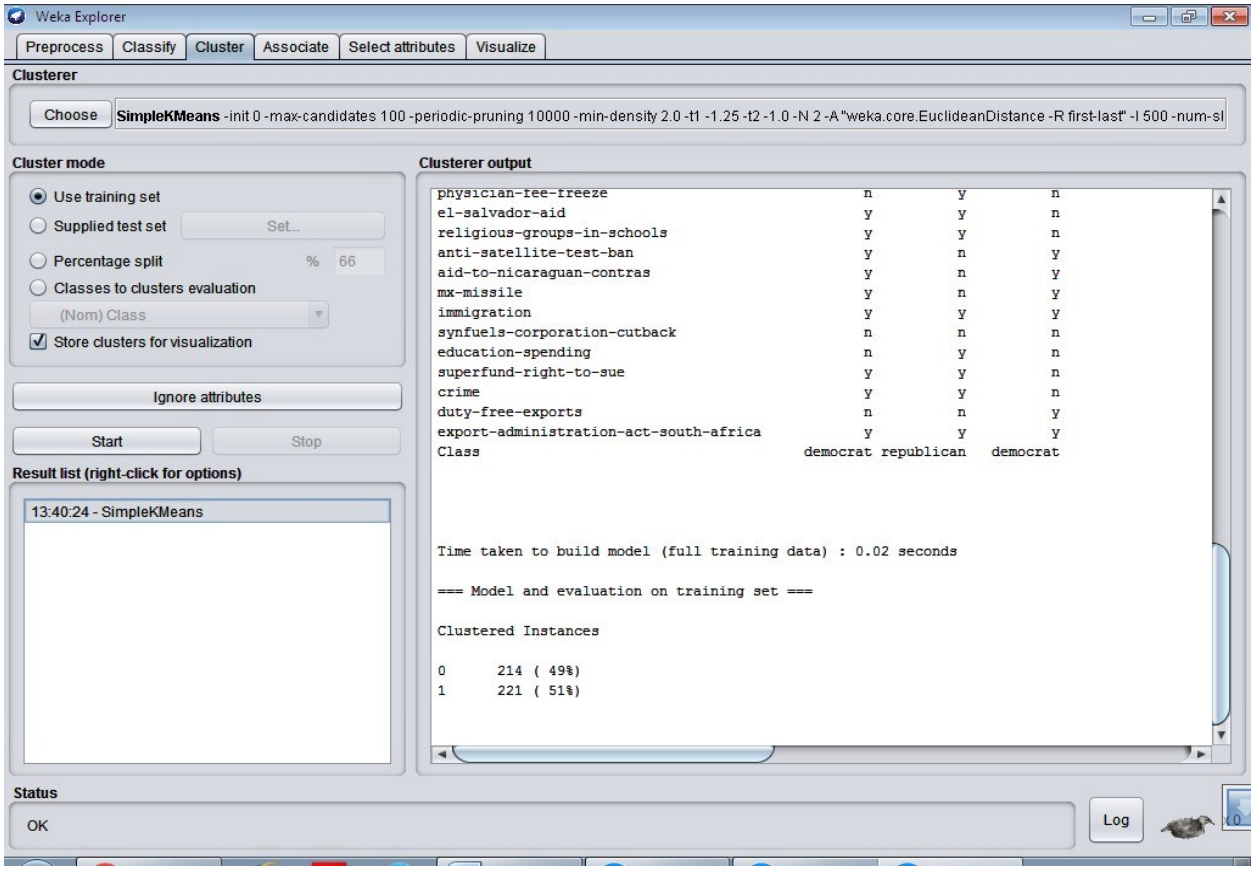


Click **Start** button

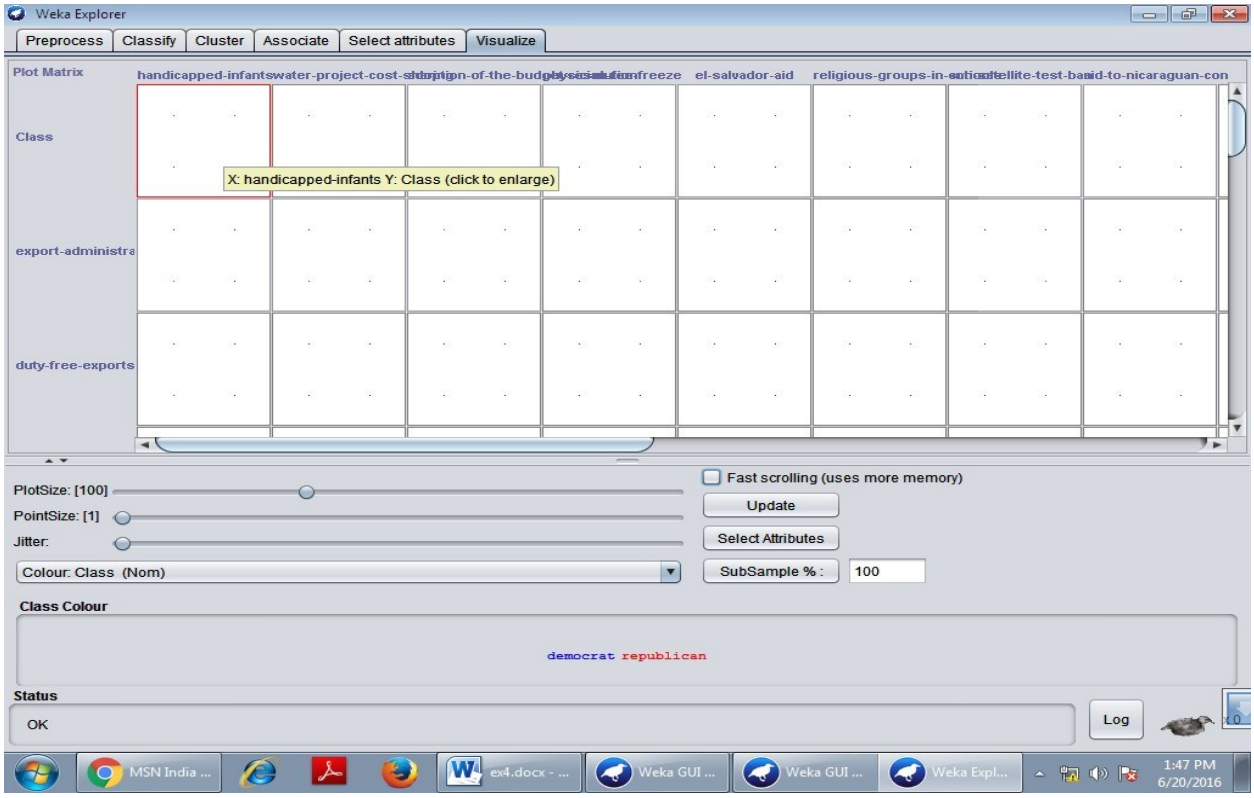
OUTPUT:

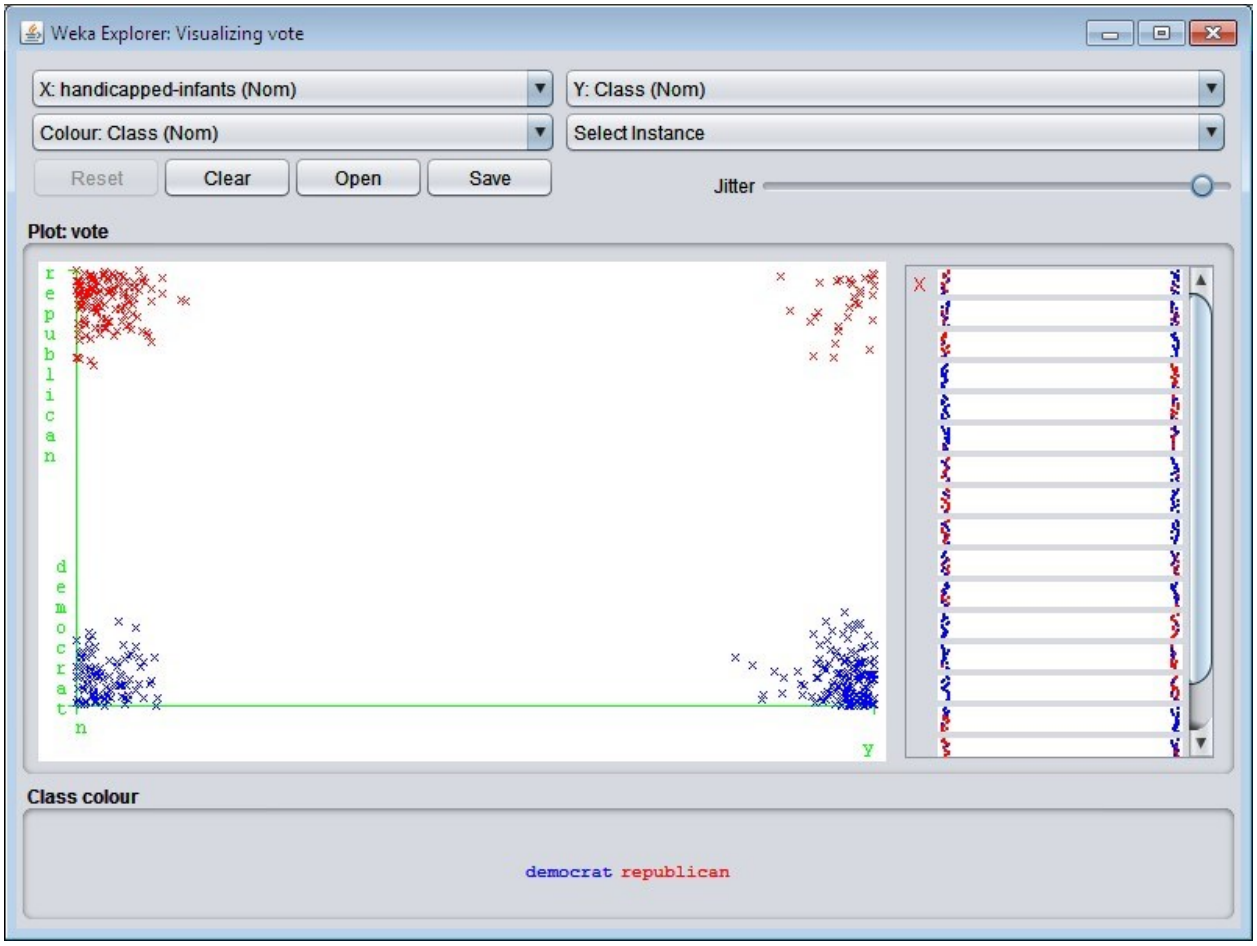

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-sl
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
13:40:24 - SimpleKMeans
Clusterer output
=== Run information ===
Scheme: weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10
Relation: vote
Instances: 435
Attributes: 17
handicapped-infants
water-project-cost-sharing
adoption-of-the-budget-resolution
physician-fee-freeze
el-salvador-aid
religious-groups-in-schools
anti-satellite-test-ban
aid-to-nicaraguan-contras
mx-missile
immigration
synfuels-corporation-cutback
education-spending
superfund-right-to-sue
crime
duty-free-exports
export-administration-act-south-africa
Class
Test mode: evaluate on training data
=== Clustering model (full training set) ===
Status
OK
Log

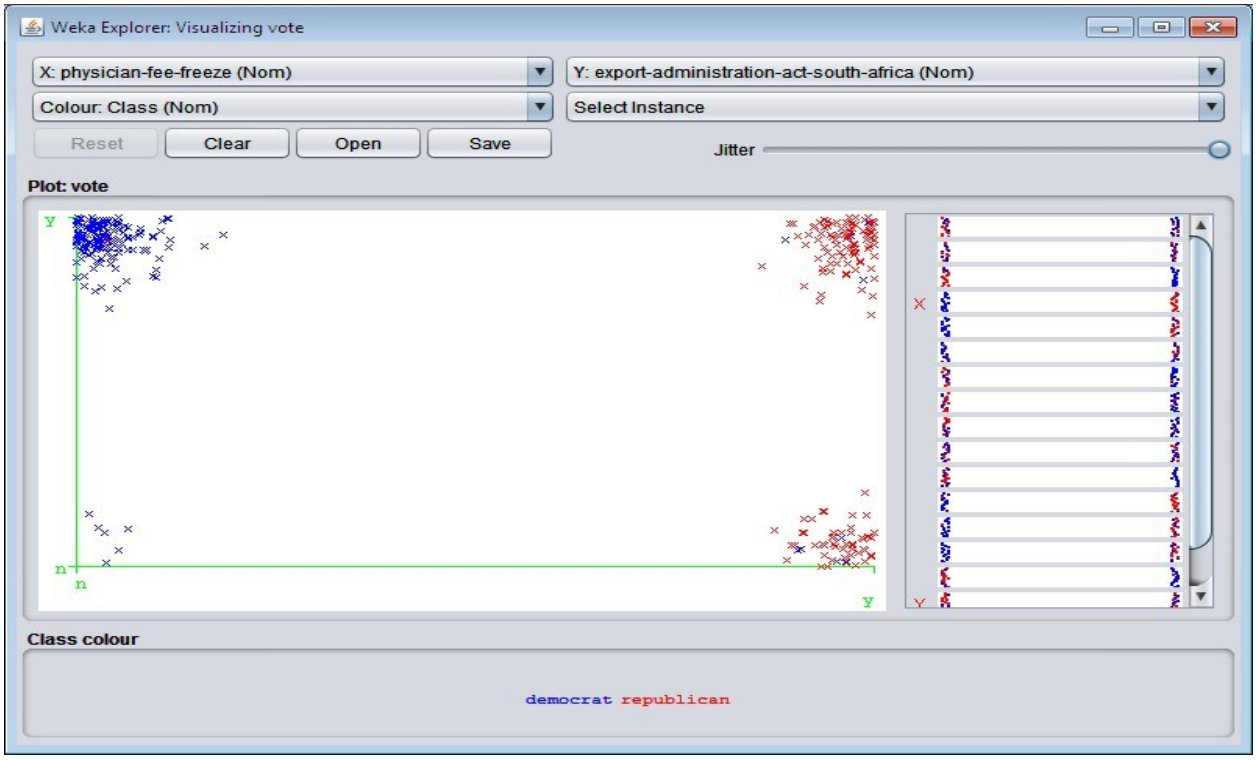

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-sl
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
13:40:24 - SimpleKMeans
Clusterer output
kMeans
======
Number of iterations: 3
Within cluster sum of squared errors: 1510.0
Initial starting points (random):
Cluster 0: n,n,y,y,y,y,n,n,y,n,n,n,y,y,y,y,democrat
Cluster 1: n,n,y,n,y,n,y,y,y,n,n,n,y,n,y,democrat
Missing values globally replaced with mean/mode
Final cluster centroids:
Cluster#
Attribute Full Data 0 1
(435.0) (214.0) (221.0)
handicapped-infants n n y
water-project-cost-sharing y y n
adoption-of-the-budget-resolution y n y
physician-fee-freeze n y n
el-salvador-aid y y n
religious-groups-in-schools y y n
anti-satellite-test-ban y n y
aid-to-nicaraguan-contras y n y
mx-missile y n y
immigration y y y
Status
OK
Log



Goto - **Visualize** tab- click one box any visualize.



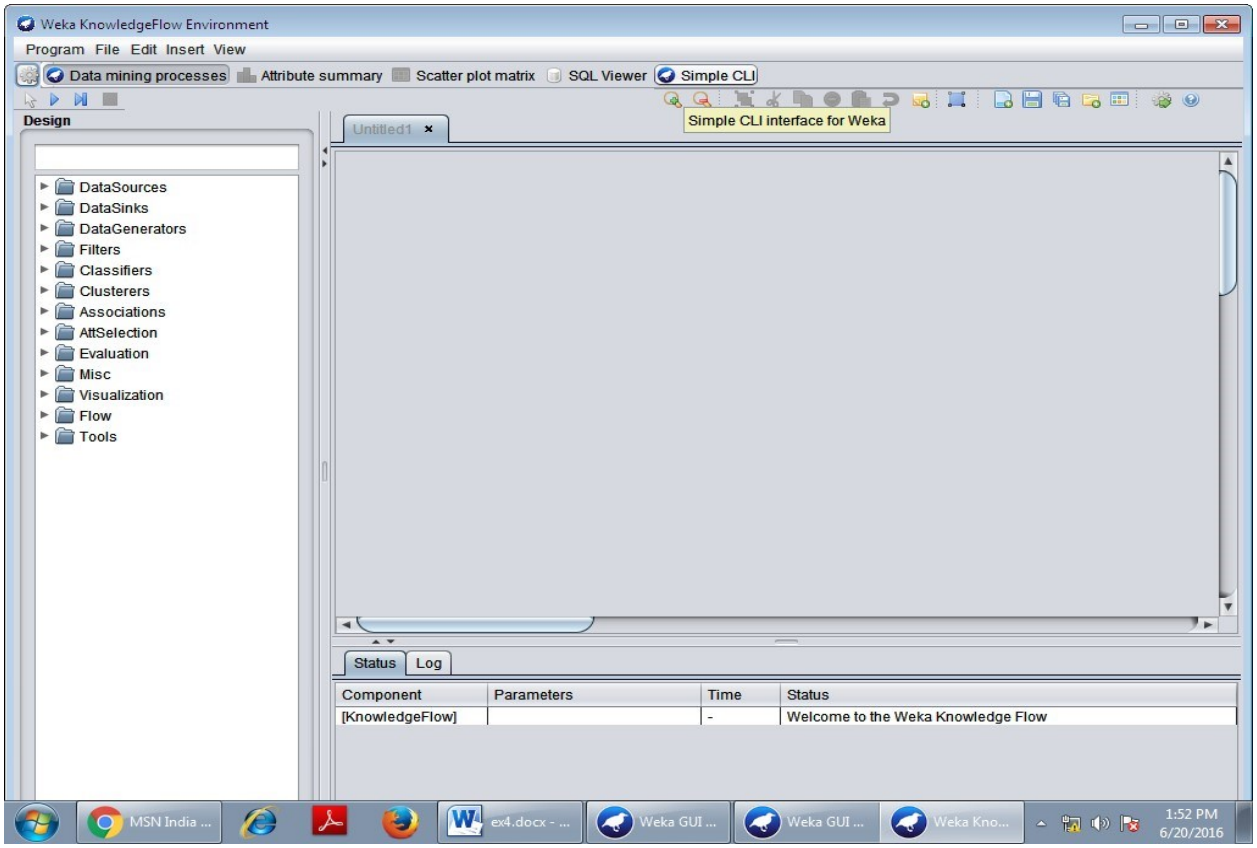




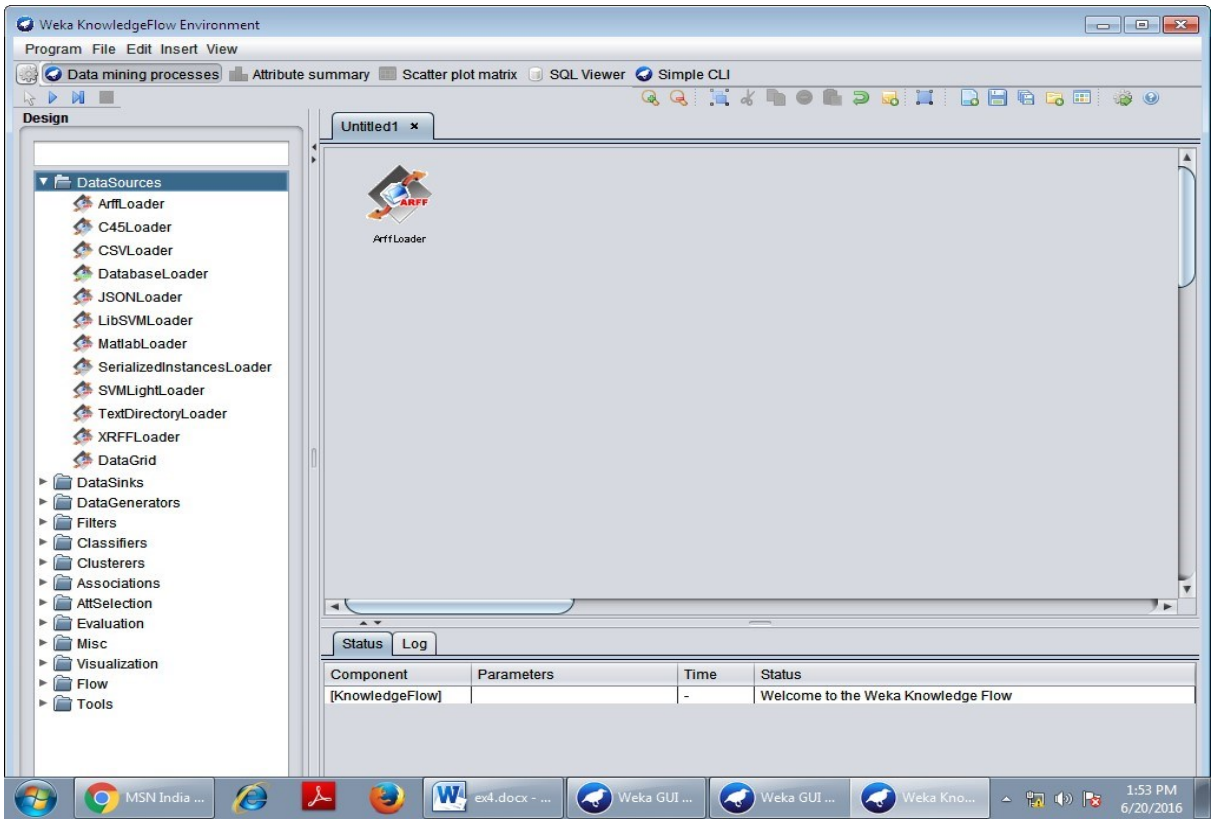
In the above move the jitter to last and to view the results of clustering.

II. Using Weka KnowledgeFlow

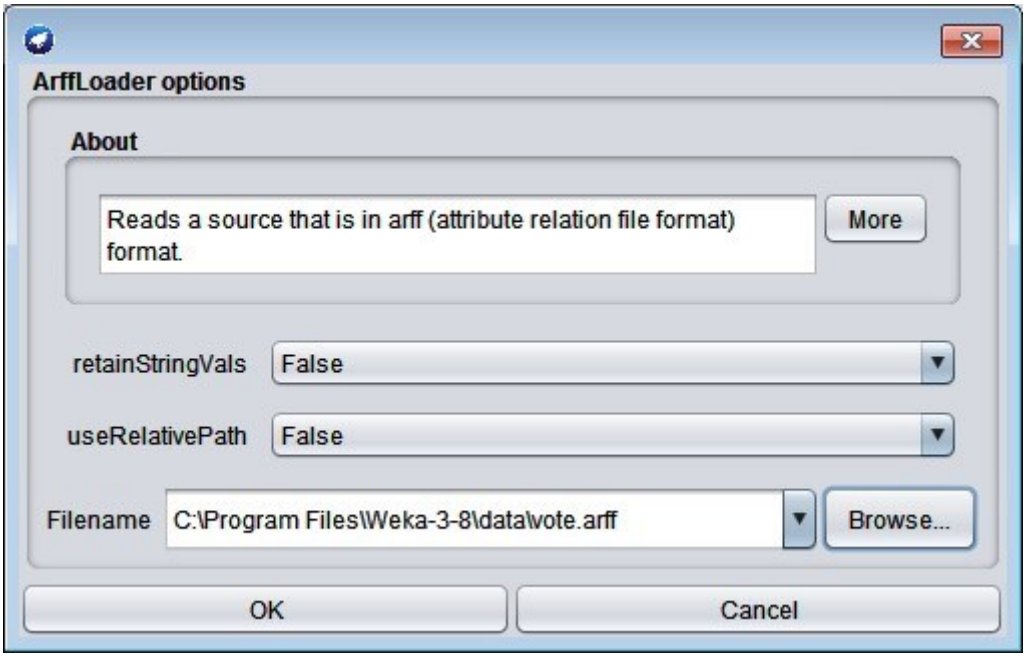
1. Open **weka** tool and click **KnowledgeFlow**.



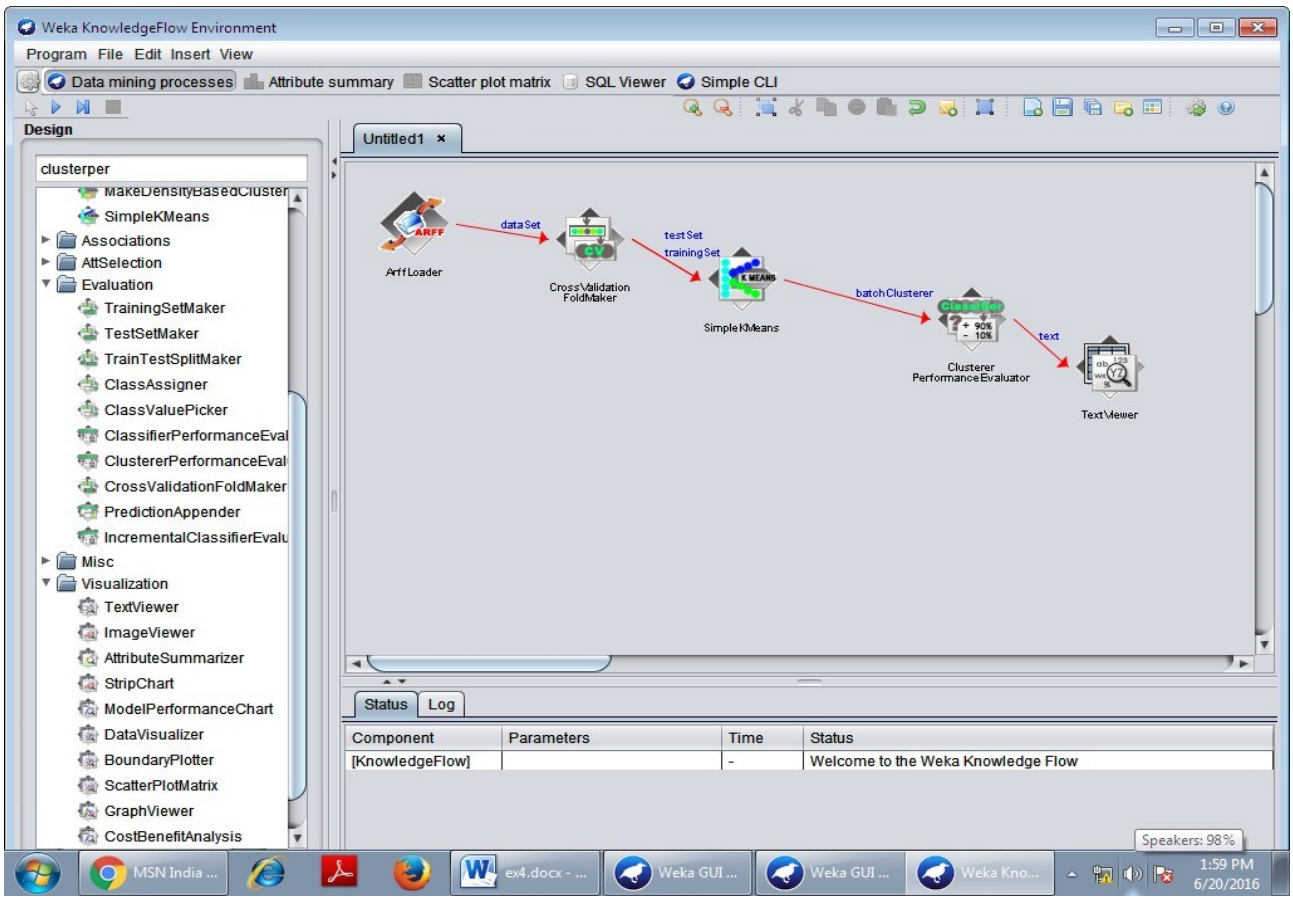
Click **DataSources** in the left side window and choose **ArffLoader** and draw in right side window



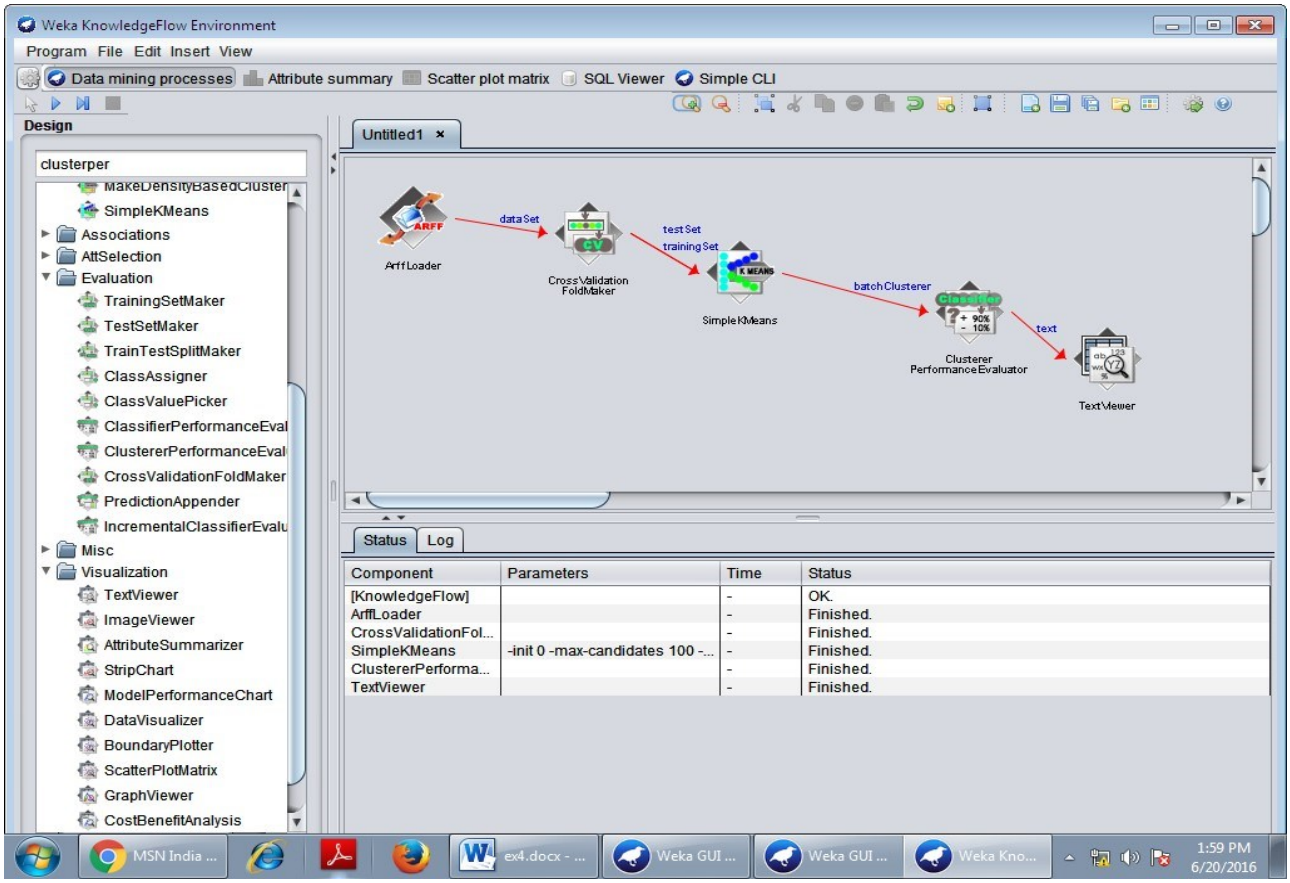
Double click **ArffLoader** and choose the file name.



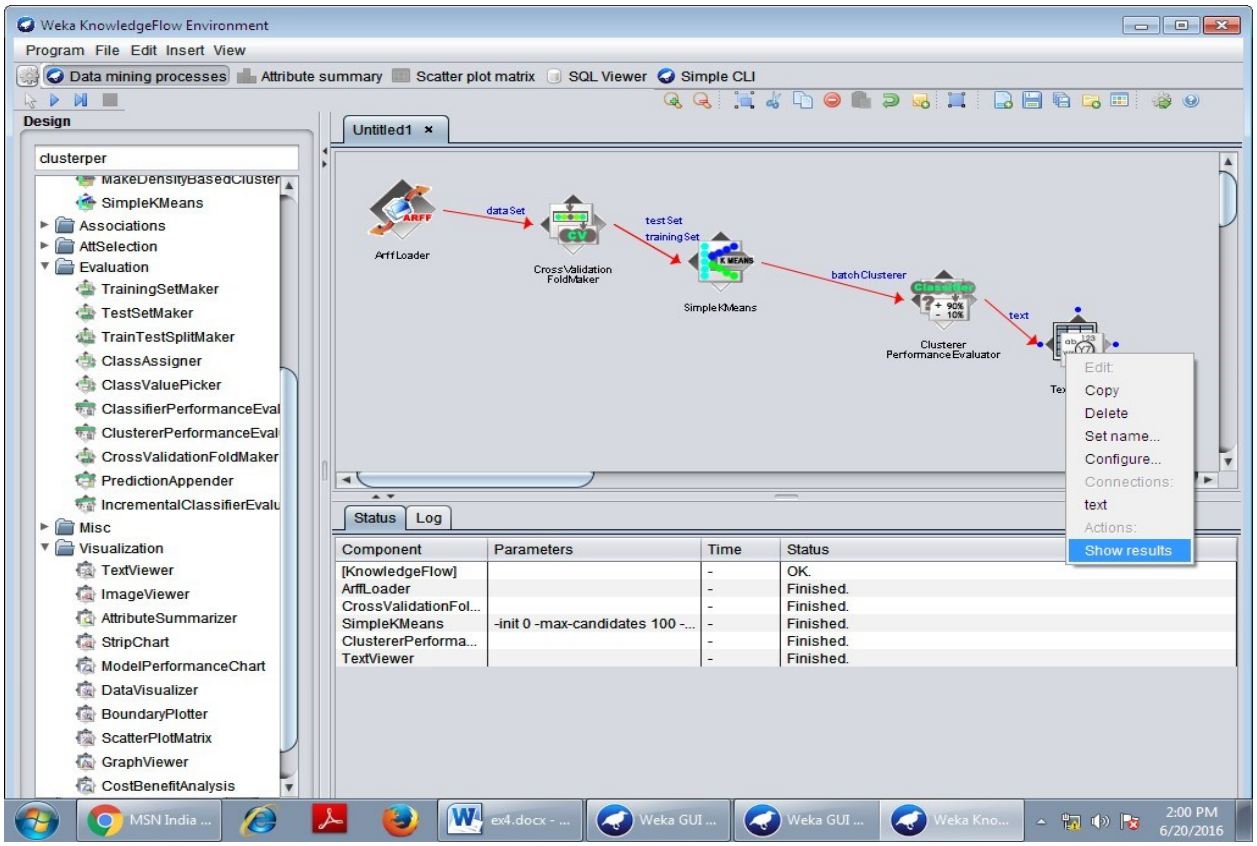
Similarly do the following.



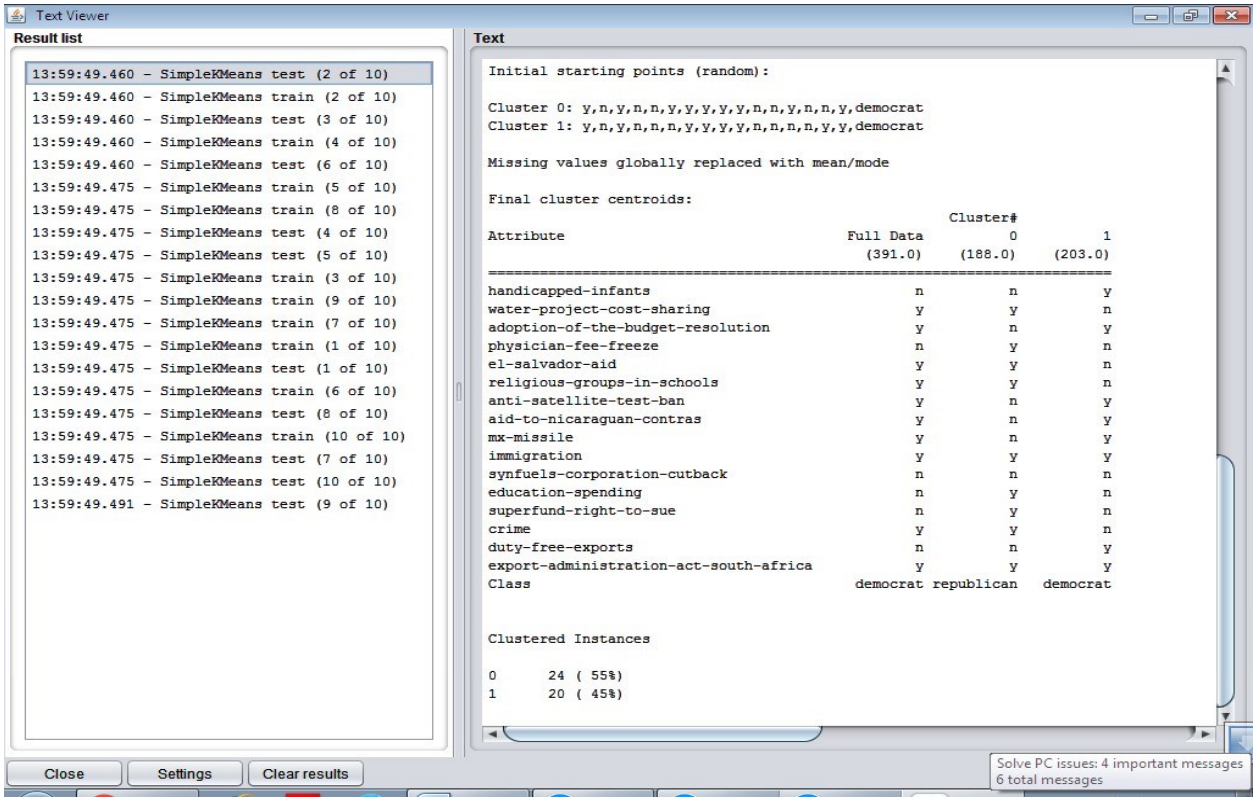
Click **Run** button.



Right click the **TextViwer** choose **Show results** option.



OUTPUT:



Exercise 2: This experiment illustrates the use of one hierarchical clustering with Weka. The sample data set used for this example is based on the vote.arff data set. This document assumes that appropriate pre-processing has been performed.

PROCEDURE:

1.Open the data file in Weka Explorer. It is presumed that the required data fields have been discretized.

2. Clicking on the cluster tab will bring up the interface for cluster algorithm.

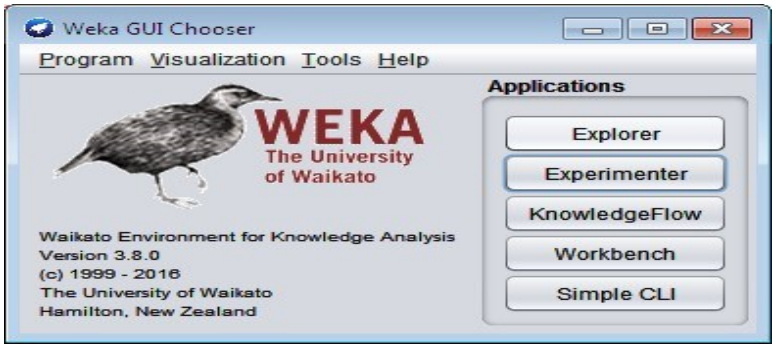
3. We will use hierarchical clustering algorithm.

4. Visualization of the graph

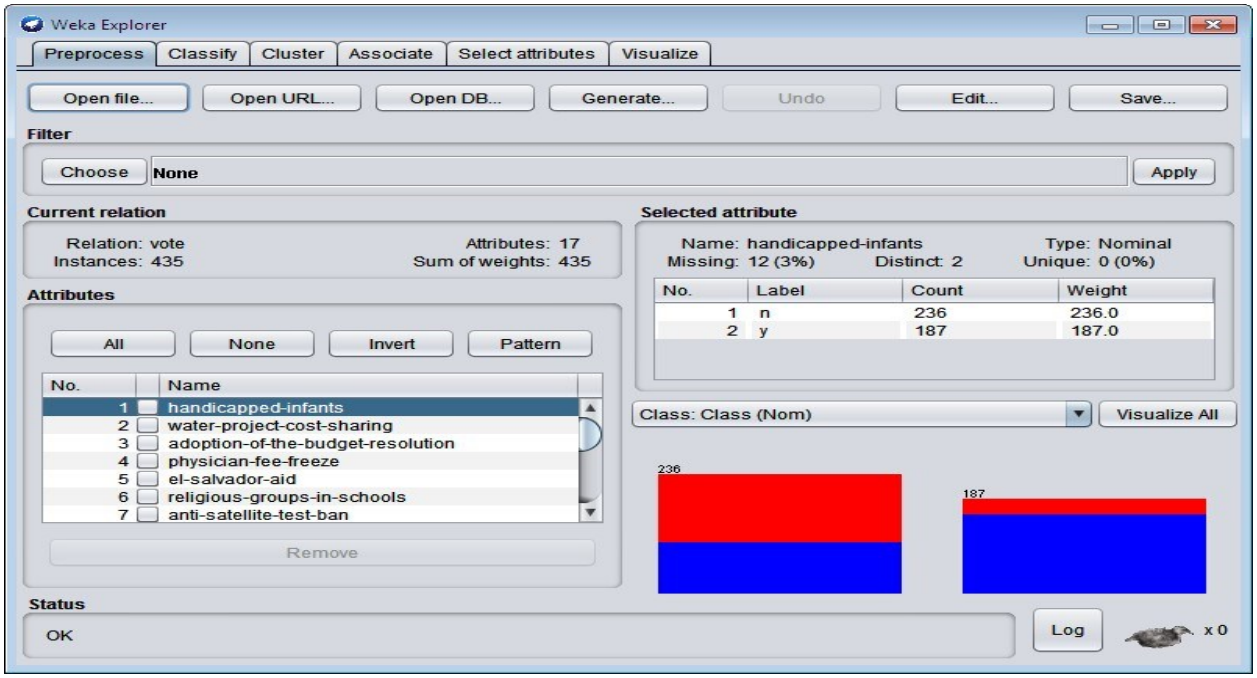
STEPS:

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

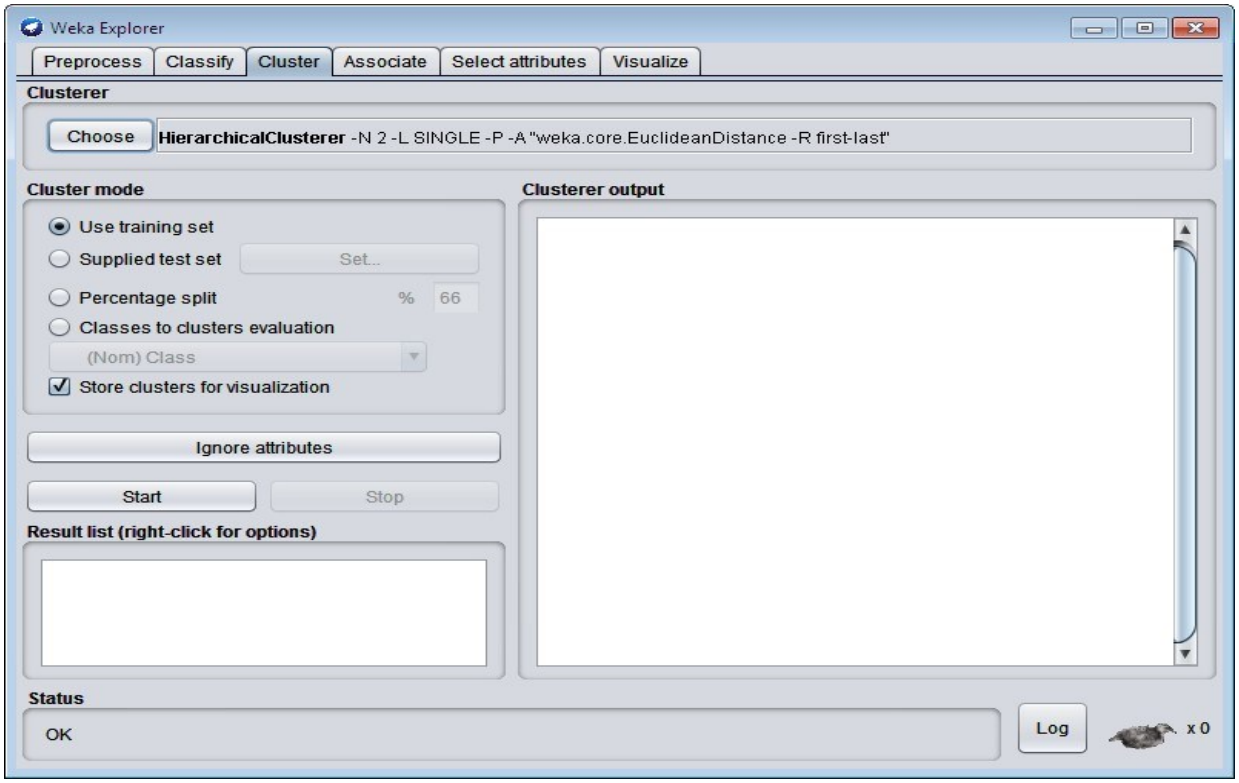
1.Open**Weka** tool and choose **Explorer.**



2.Click - **Open file**… in **preprocess** tab –choose **vote.arff.**

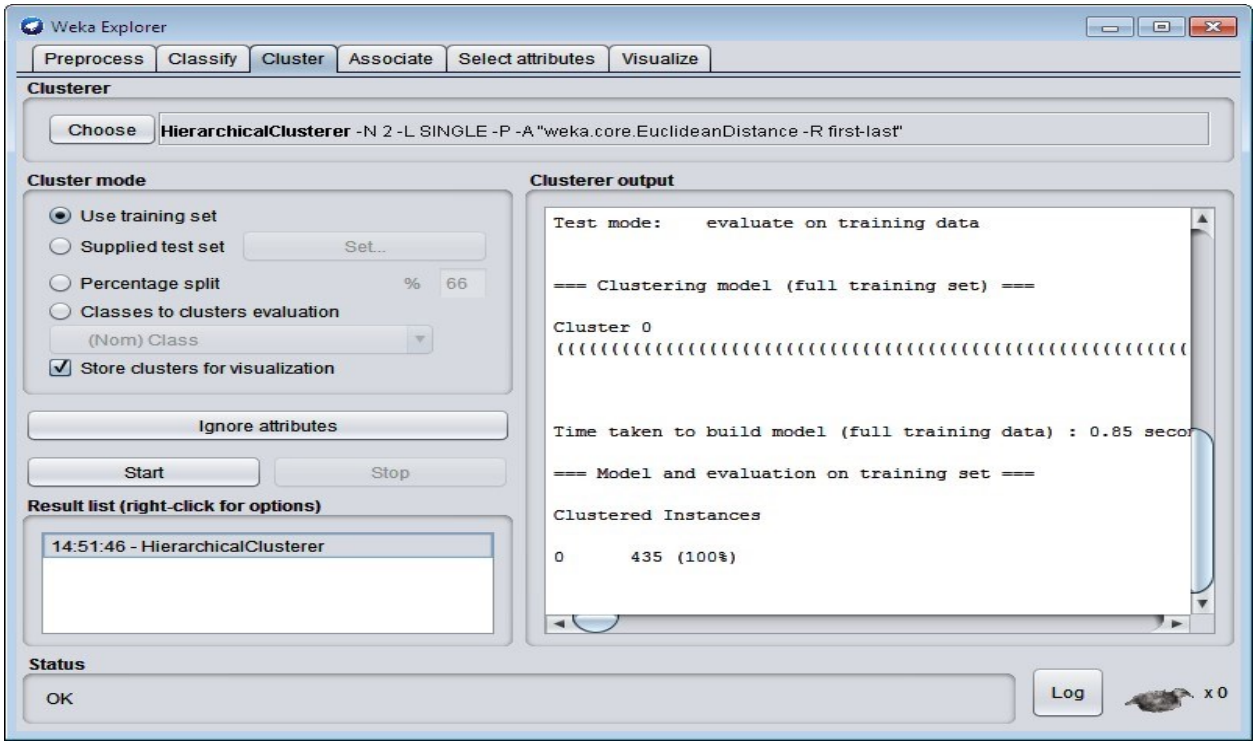


3.Goto **Cluster** tab – click **choose** button - select **HierarchicalClusterer.**

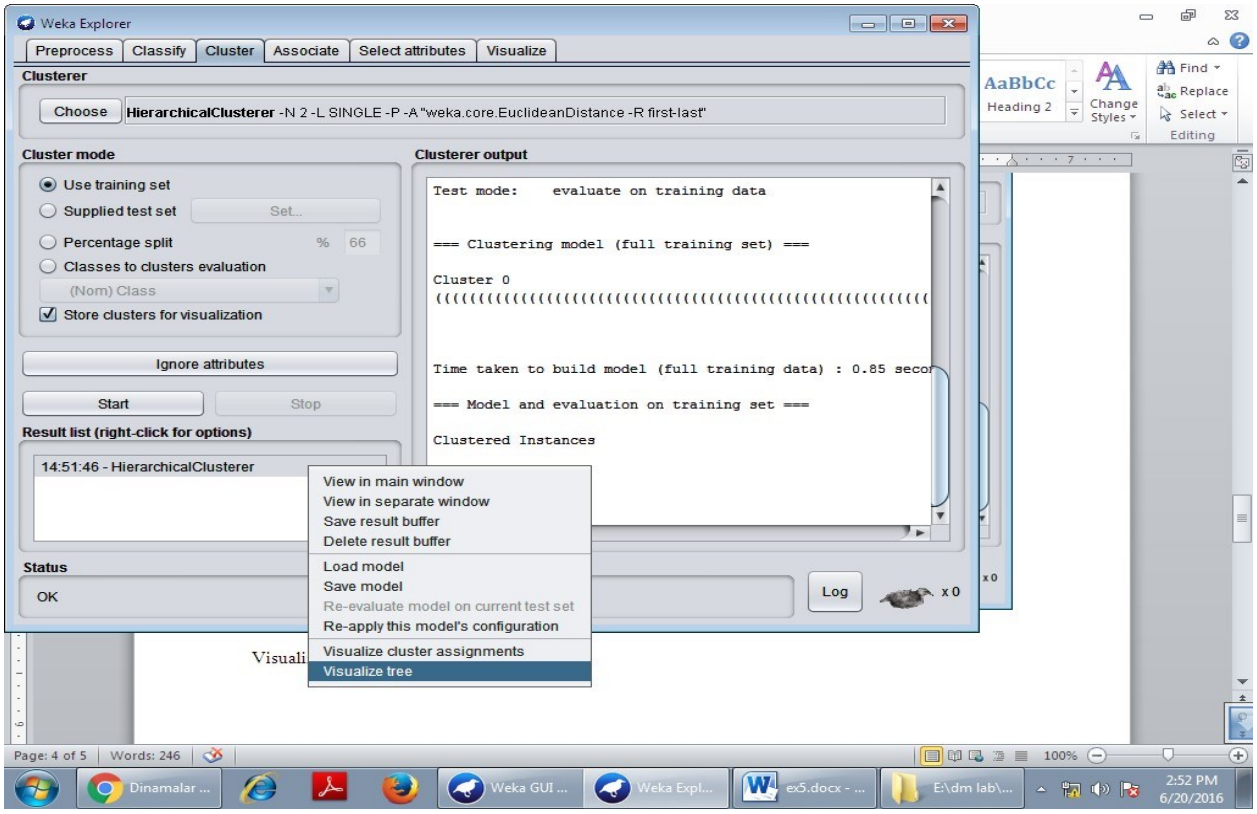


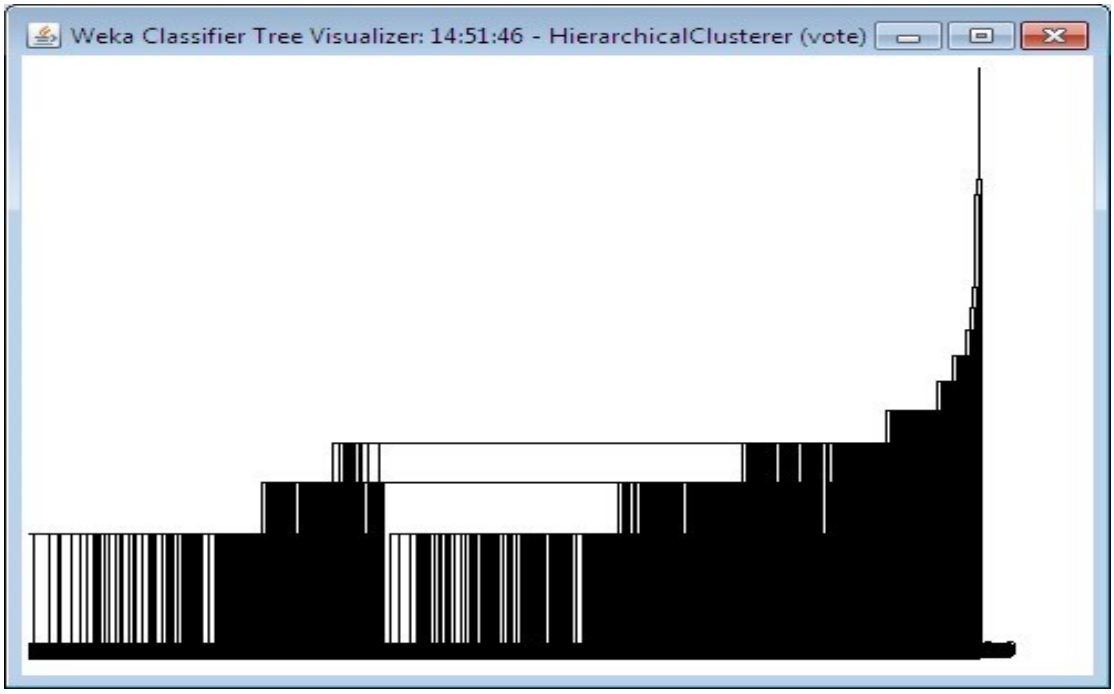
5.Click **Start** button.

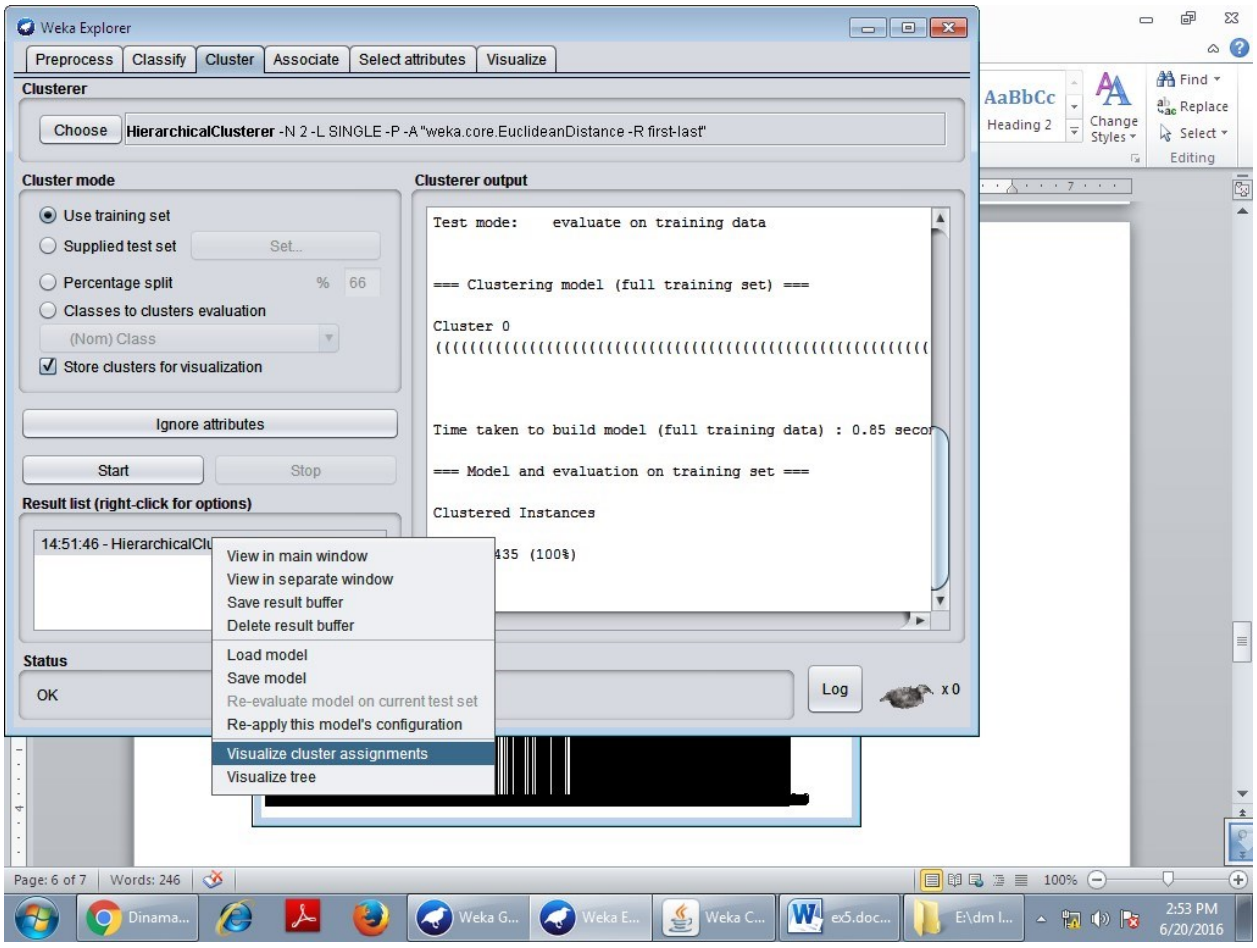
OUTPUT:

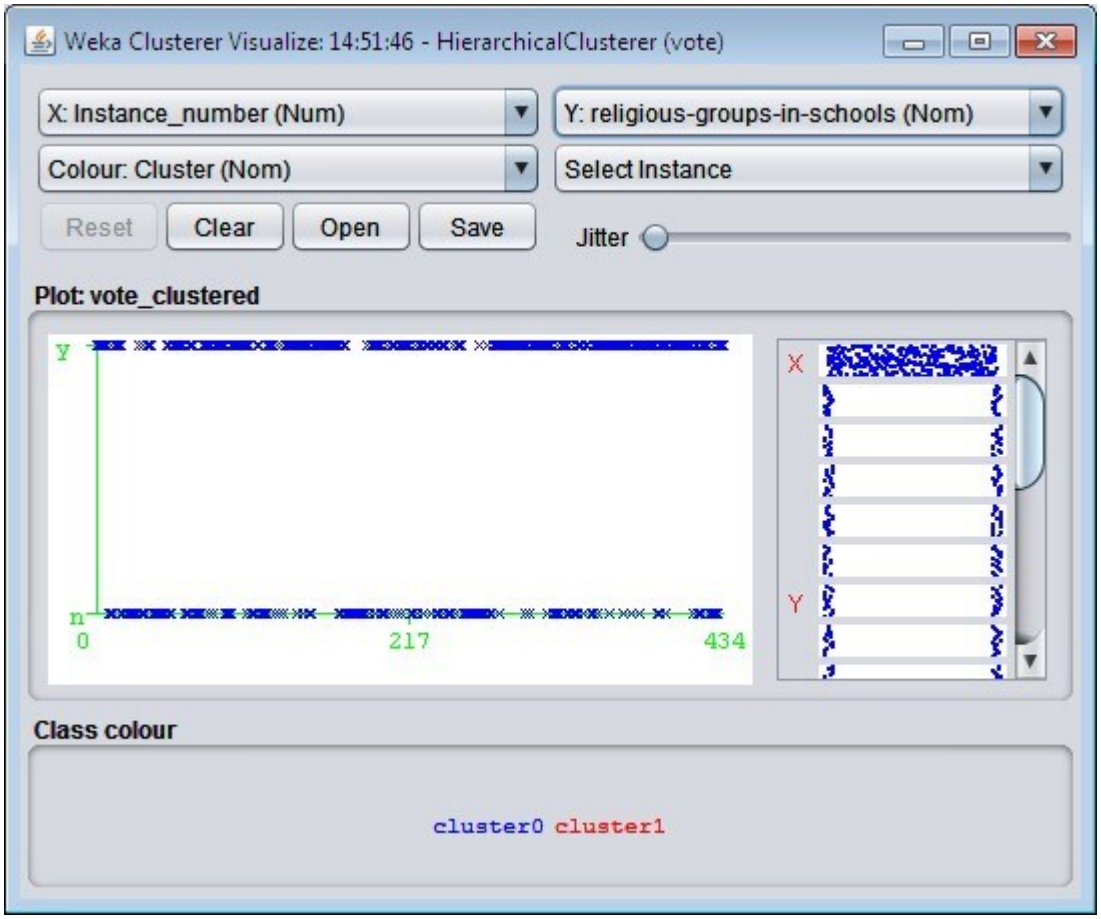


6. Visualize the tree by **right click**ing and choose **Visualize Tree** option.




Weka Classifier Tree Visualizer: 14:51:46 - HierarchicalClusterer (vote)


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
14:51:46 - HierarchicalClu
Clusterer output
Test mode: evaluate on training data
=== Clustering model (full training set) ===
Cluster 0
Time taken to build model (full training data) : 0.85 secor
=== Model and evaluation on training set ===
Clustered Instances
435 (100%)
View in main window
View in separate window
Save result buffer
Delete result buffer
Load model
Save model
Re-evaluate model on current test set
Re-apply this model's configuration
Visualize cluster assignments
Visualize tree
Status
OK
Log
x 0
Page: 6 of 7
Words: 246
100%
2:53 PM
6/20/2016



Exercise 3: The goal of this data mining study is to find groups of animals in the ***zoo*** dataset, and to check whether these groups correspond to the real animal types in the dataset.

a. What types of variables are in this dataset?
b. How many rows / cases are there?
c. How many animal types are represented in this dataset? List them here.
d. After removing the ***type*** attribute, go to the ***Cluster*** tab. How many clustering algorithms are available in **Weka**?
e. List the clustering algorithms seen in class, and map these to the ones provided in **Weka**.
f. Start using the ***SimpleKMeans*** clusterer choosing 7 clusters. Do the clusters learnt and their centroids seem to match the animal types?
g. Start using the ***HierarchicalClusterer*** with Single Link.
h. Start using the ***HierarchicalClusterer*** with Complete Link.
i. Start using the ***HierarchicalClusterer*** with Average Link.
j. Start using the ***HierarchicalClusterer*** with Mean Link.
k. Start using the ***HierarchicalClusterer*** with Centroid Link.

Exercise 4: Using Weka Knowledge Flow to cluster data for the above exercises.

Exercise 5: Using Java programming language to cluster data for the above exercises

LAB 10: ÔN TẬP – KIỂM TRA