MALLA REDDY ENGINEERING COLLEGE

MAISAMMAGUDA(H), GUNDLAPOCHAMPALLY (V), MEDCHAL (M).
MEDCHAL - MALKAJGIRI DISTRICT TELANGANA – 500100.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is certify that it is a bonafide record of practical work done by Mr./Ms._____ bearing Regd. No. of _____B.Tech III Year II Semester, CSE Branch in the _____ Laboratory during the Academic Year _____ has satisfactorily completed the course of, programs prescribed by the Malla Reddy Engineering College supervision.

Lab In-charge

Head of the Department

Internal Examiner

External Examiner

DATA MINING RECORD

Index

The full form of WEKA is Waikato Environment for Knowledge Analysis.The Weka team has put a tremendous amount of effort into continuously developing and maintaining the system **since 1994**. The development of Weka was funded by a grant from the New Zealand Government's Foundation for Research, Science and Technology.

A collection of open source of many data mining and machine learning algorithms, including

- Pre-processing on data
- Classification:
- Clustering
- Association rule extraction

The **key features** responsible for Weka's success are:

- It provides many different algorithms for data mining and machine learning
- It is open source and freely available
- It is platform-independent
- It is easily useable by people who are not data mining specialists
- It provides flexible facilities for scripting experiments
- It has kept up-to-date, with new algorithms being added as they appear in the research.
- Portability, since it is fully implemented in the Java programming language and thus runs on almost any modern computing platform
- Free availability under the GNU General Public License
- A comprehensive collection of data preprocessing and modeling techniques
- Ease of use due to its graphical user interfaces

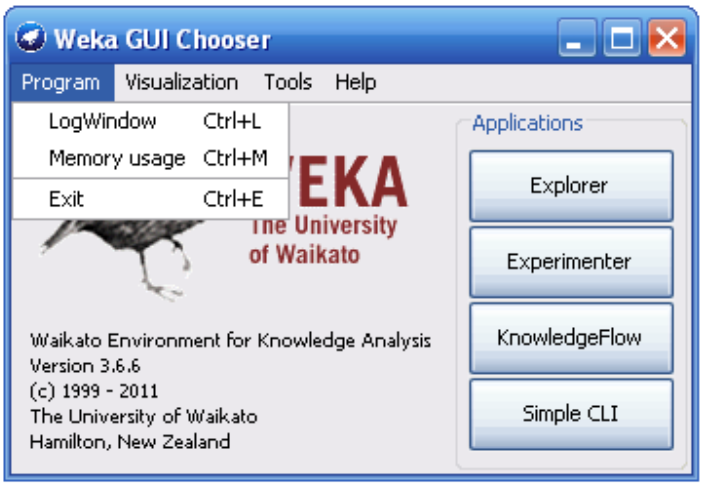
ENVIRONMENT

The menu consists of four sections:

Program Menu:

- LogWindow
  Opens a log window that captures all that is printed to stdout or stderr. Useful for environments like MS Windows, where WEKA is normally not started from a terminal.

- Exit Closes WEKA.



Visualization Menu:

It ways of visualizing data with WEKA.

- Plot
  It is used for plotting a 2D plot of a dataset.

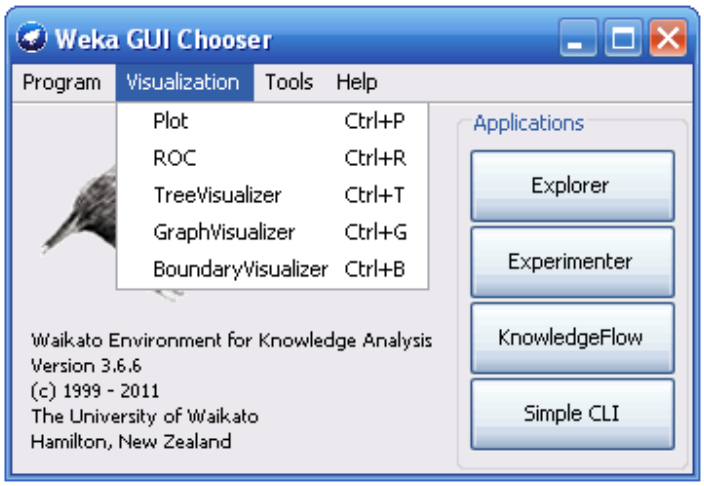
- ROC
  It displays a previously saved ROC curve.

- TreeVisualizer
  It used for displaying directed graphs, e.g., a decision tree.

- GraphVisualizer

It visualizes XML BIF or DOT format graphs, e.g., for Bayesian networks.

- BoundaryVisualizer
  It allows the visualization of classifier decision boundaries in two dimensions.



Tools Menu

It is useful for other applications.

- ArffViewer
  An MDI application for viewing ARFF files in spread- sheet format.

- SqlViewer
  It represents an SQL worksheet, for querying databases via JDBC.

- Bayes net editor
  It is an application for editing, visualizing and learning Bayes nets.

Help Menu

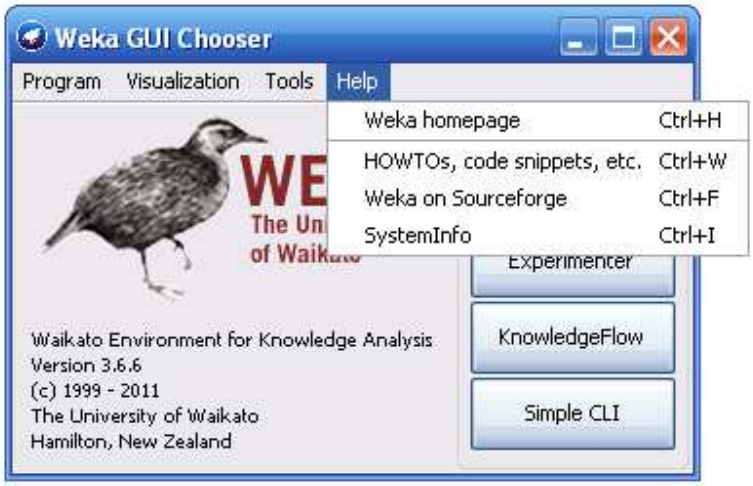
It is used for online resources for WEKA can be found here.

- Weka homepage
  It opens a browser window with WEKA's homepage.

- HOWTOs, code snippets, etc.
  The general WekaWiki [2], containing lots of examples and HOWTOs around the development and use of WEKA.

- Weka on Sourceforge
  WEKA's project homepage on Sourceforge.net.

- SystemInfo
  Lists some internals about the Java/WEKA environment, e.g., the CLASSPATH.


Weka GUI Chooser
Program
Visualization
Tools
Help
Weka homepage Ctrl+H
HOWTOs, code snippets, etc. Ctrl+W
Weka on Sourceforge Ctrl+F
SystemInfo Ctrl+I
Experimenter
KnowledgeFlow
Simple CLI
Waikato Environment for Knowledge Analysis
Version 3.6.6
(c) 1999 - 2011
The University of Waikato
Hamilton, New Zealand

APPLICATIONS

The GUI Chooser consists of four buttons—one for each of the four major Weka applications—and four menus.

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 the Explorer but with a drag-and-drop interface. One advantage is that it supports incremental learning.

- SimpleCLI
  It 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.

EXPLORER

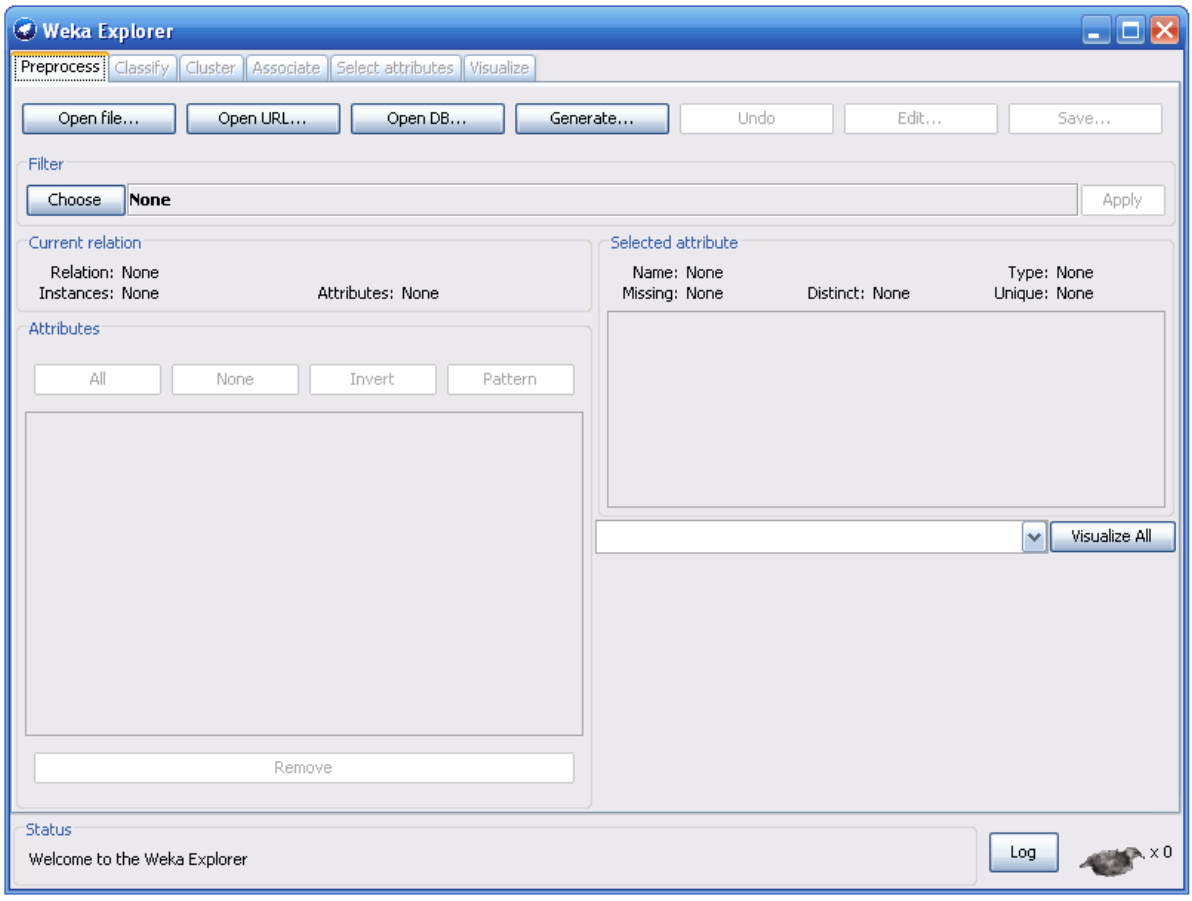
It is a user interface which contains a group of tabs just below the title bar.

The tabs are as follows:

1. Preprocess
2. Classify
3. Cluster
4. Associate
5. Select Attributes
6. Visualize

The bottom of the window contains status box, log and WEKA bird.

PREPROCESSING



LOADING DATA

The first four buttons at the top of the preprocess section enable you to load Data into WEKA:

1. **Open file**:

   It shows a dialog box allowing you to browse for the data file on the local file system.

2. **Open URL**:

   Asks for a Uniform Resource Locator address for where the data is stored.

3. **Open DB**:

   It reads data from a database.

4. **Generate**:

   It is used to generate artificial data from a variety of Data Generators.

Using the Open file button we can read files in a variety of formats like WEKA's ARFF format, CSV format. Typically ARFF files have .arff extension and CSV files .csv extension.

THE CURRENT RELATION

The Current relation box contains the currently loaded data i.e. interpreted as a single relational table in database terminology, which has three entries:

1. **Relation:**
   a. It provides the name of the relation in the file from which it was loaded. Filters are used modify the name of a relation.
2. **Instances:**
   a. The number of instances (data points/records) in the data.
3. **Attributes:**
   a. The number of attributes (features) in the data.

ATTRIBUTES

It is located below the current relation box which contains four buttons. They are:

1. All is used to tick all boxes
2. None is used to clear all boxes
3. Invert is used make ticked boxes unticked.
4. Pattern is used to select attributes by representing an expression.
   E.g. a.* is used to select all the attributes that begins with a.

SELECTED ATTRIBUTE:

It is located beside the current relation box which contains the following:

1. Name

   It specifies the name of the attribute i.e. same as in the attribute list.

2. Type

   It specifies the type of attribute, most commonly Nominal or Numeric.

3. Missing

   It provides a numeric value of instances in the data for which an attribute is missing.

4. Distinct

   It provides the number of different values that the data contains for an attribute.

5. Unique

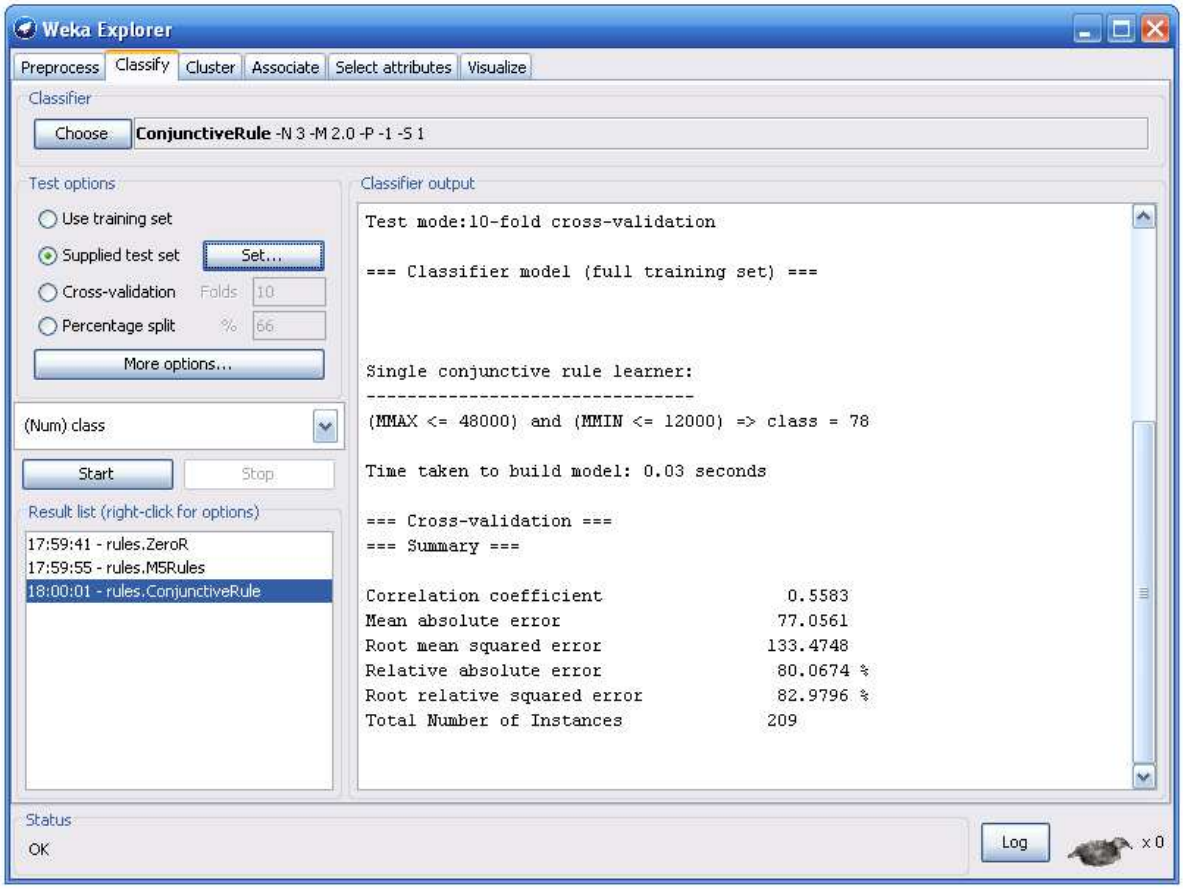
   It provides the number of instances in the data having a value for an attribute that no other instances have.

FILTERS

By clicking the Choose button at the left of the Filter box, it is possible to select one of the filters in WEKA. Once a filter has been selected, its name and options are shown in the field next to the Choose button, by clicking on this box with the left mouse button it shows a GenericObjectEditor dialog box which is used to configure the filter.

CLASSIFICATION

Classification has a text box which gives the name of currently selected classifier, and its options. By clicking it with the left mouse button it shows a GenericObjectEditor dialog box, which is same as for filters i.e. used to configure the current classifier options.



TEST OPTIONS

The result of applying the chosen classifier will be tested according to the options that are set by clicking in the Test options box. There are four test modes:

1. Use training set.
2. Supplied test set.
3. Cross-validation.
4. Percentage split.

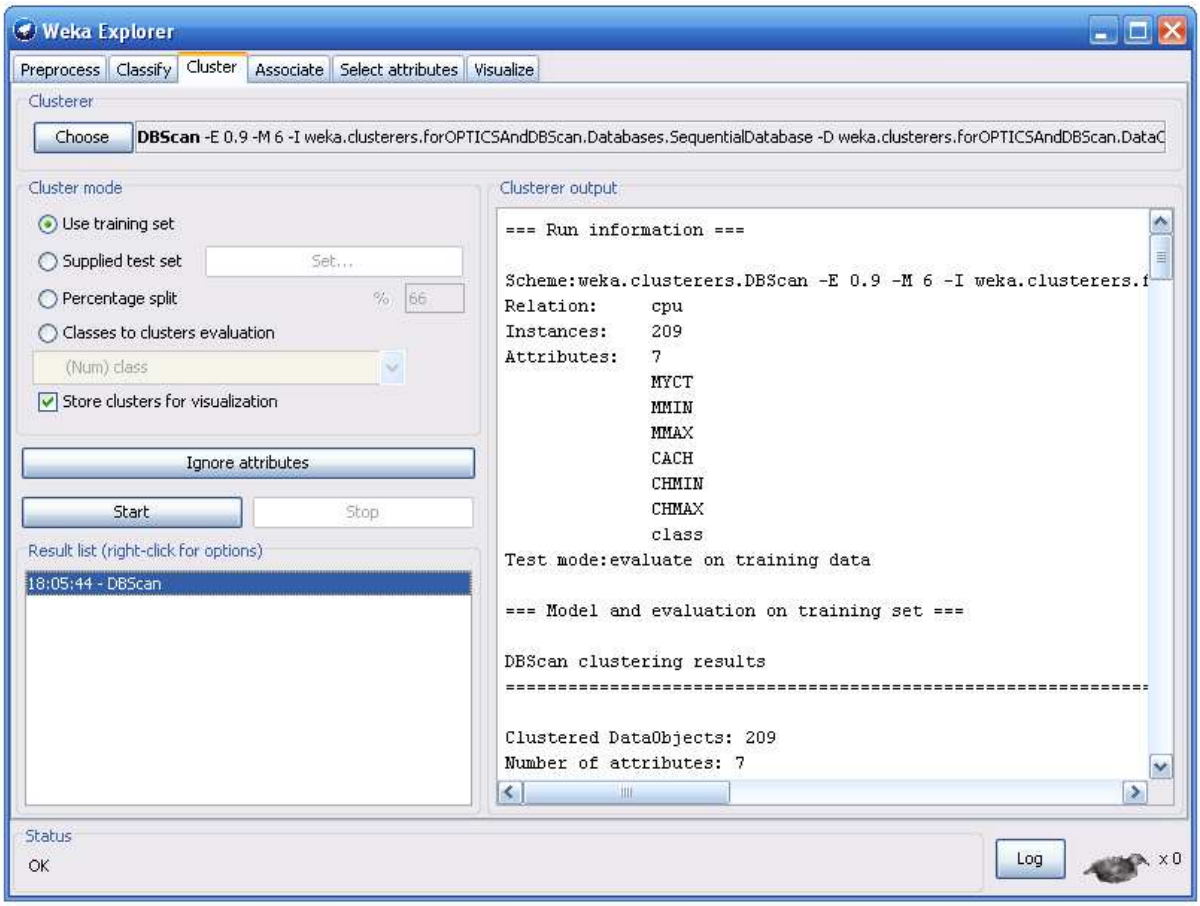
Once the classifier, test options and class have all been set, the learning process is started by clicking on the Start button. We can stop the training process at any time by clicking on the Stop button.

The Classifier output area to the right of the display is filled with text describing the results of training and testing.

After training several classifiers, the Result List will contain several entries using which we can move over various results that have been generated. By pressing Delete we can remove a selected entry from the results.

CLUSTERING

By clicking the text box beside the choose button in the Clusterer box, it shows a dialog box used to choose a new clustering scheme.



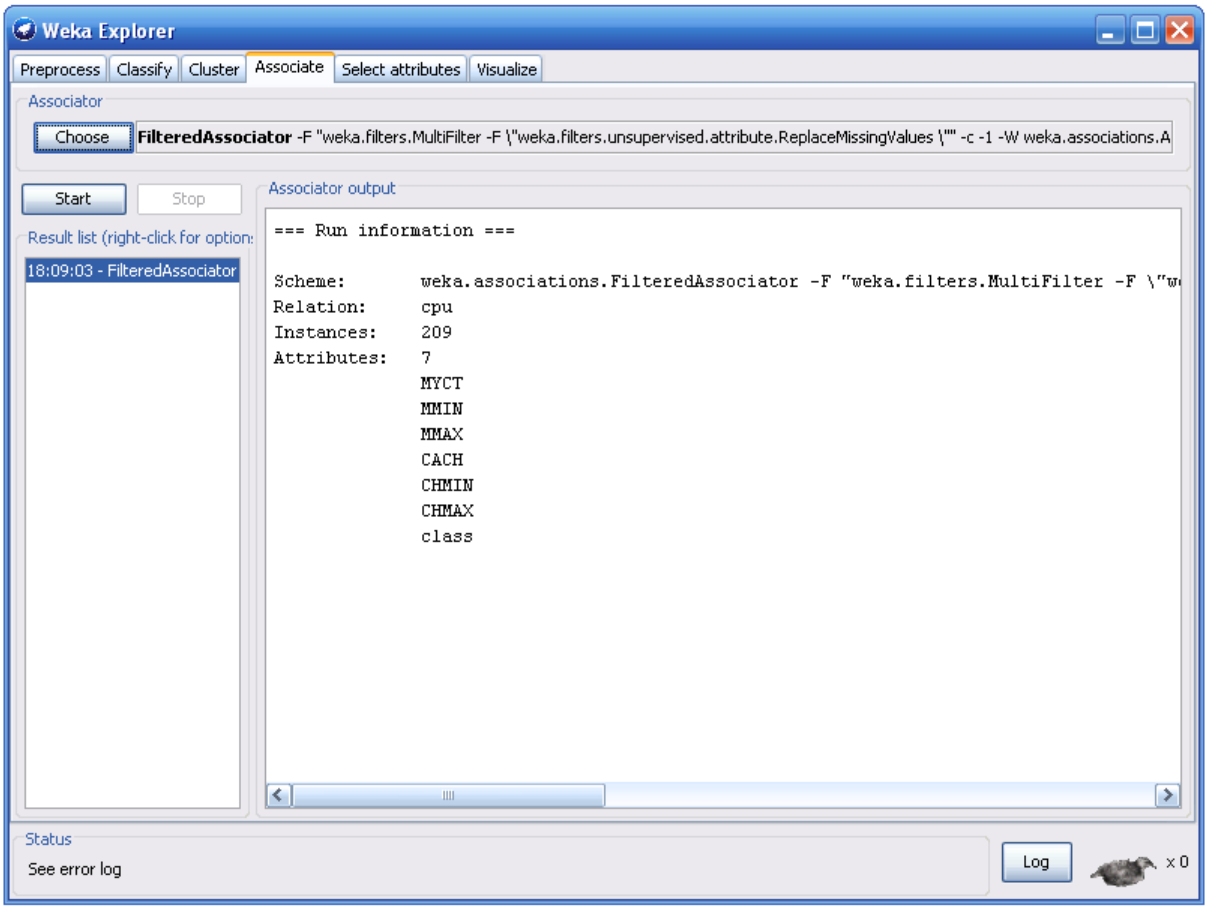
The Cluster mode box is used to choose what to cluster and how to evaluate the results. The first three options in it are same as in classification like Use training set, Supplied test set and Percentage split. The fourth option is classes to clusters evaluation.

An additional option in the Cluster mode box is the Store clusters for visualization which finds whether or not it will be possible to visualize the clusters once training is complete.

Ignore Attributes: when clustering, some attributes in the data should be ignored. It shows a small window that allows you to select which attributes are ignored.

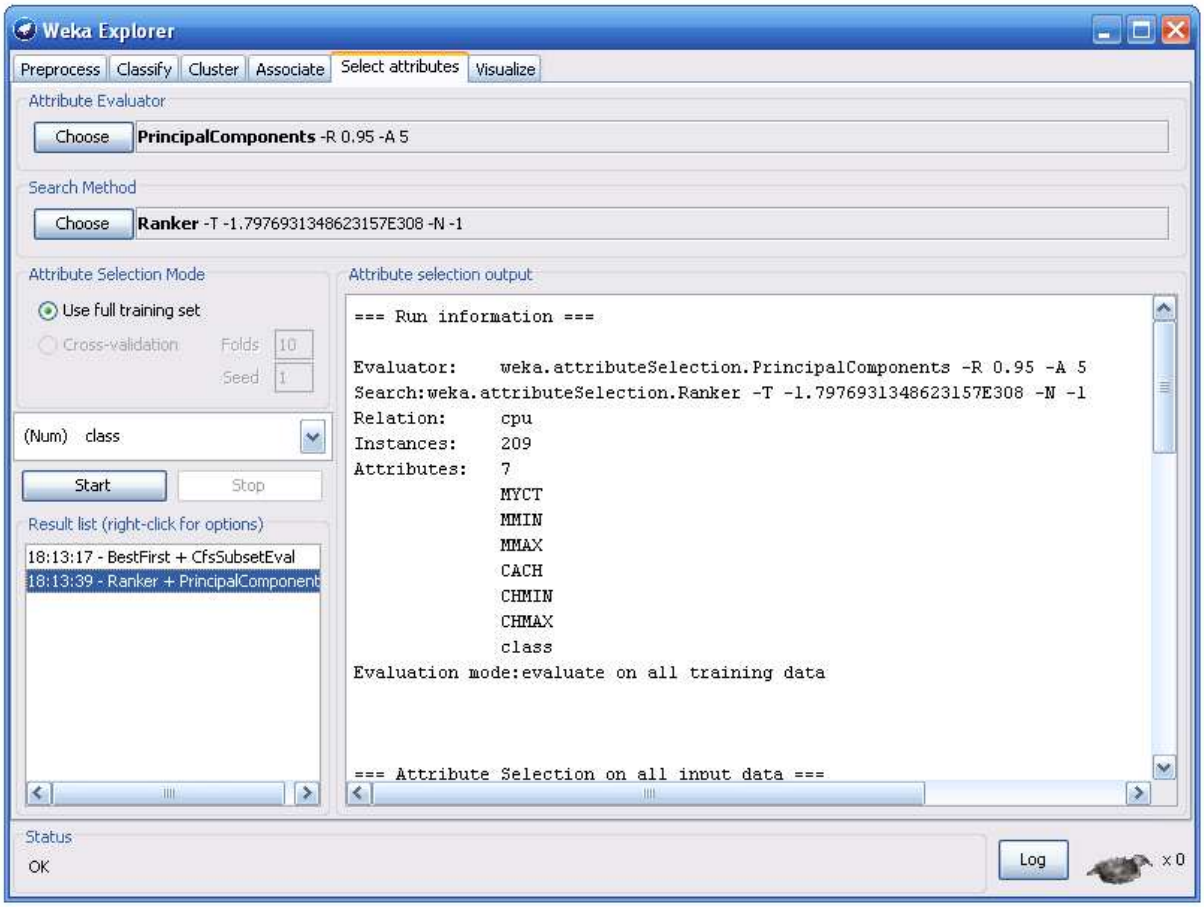
ASSOCIATING

It contains schemes for learning association rules, and the learners are chosen and configured in the same way as the clusters, filters, and classifiers in the other panels.



SELECTING ATTRIBUTES

Attribute selection involves searching through all possible combinations of attributes in the data to find which subset of attributes works best for prediction. To do this, two objects must be set up: an attribute evaluator and a search method. The evaluator determines what method is used to assign a worth to each subset of attributes. The search method determines what style of search is performed.



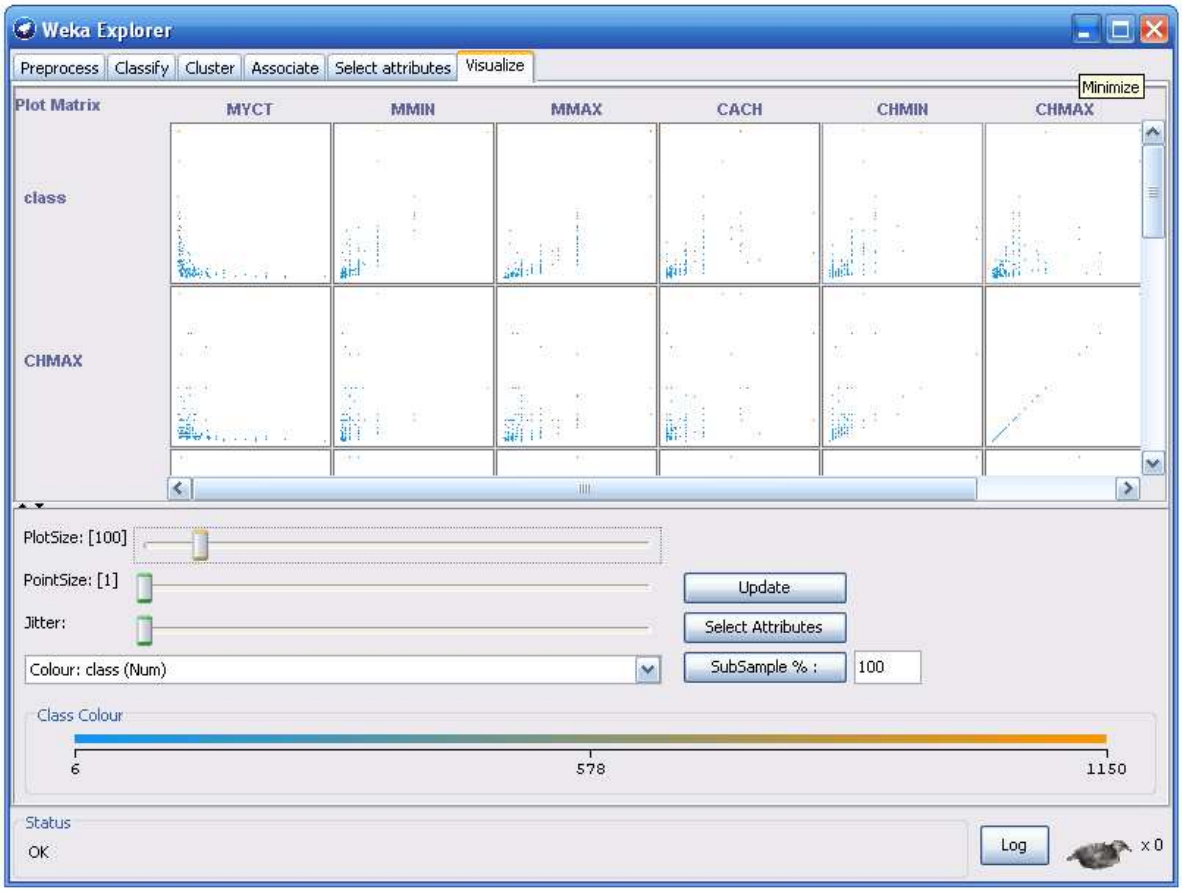
The Attribute Selection Mode box has two options:

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.

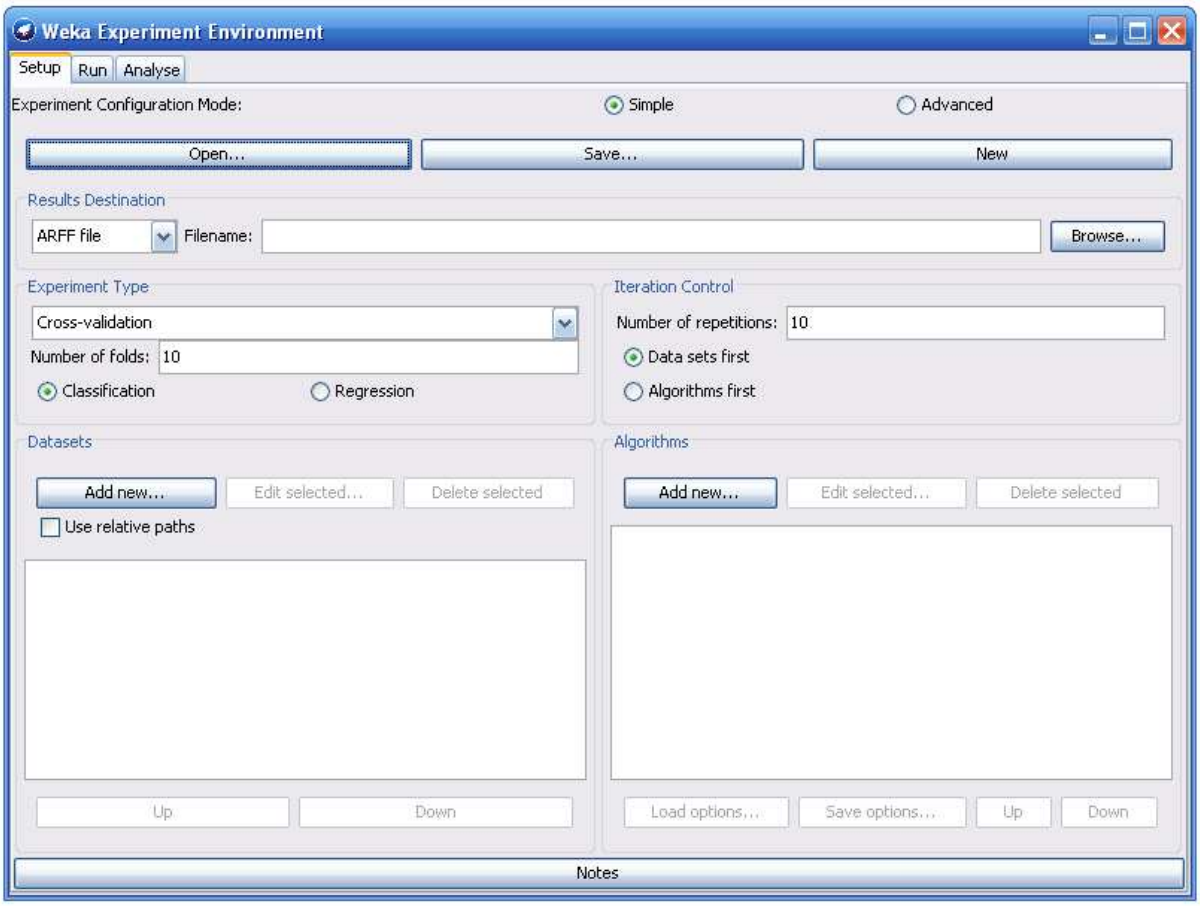
VISUALISING

WEKA's visualization section allows you to visualize 2D plots of the current relation.



EXPERIMETER

The Weka Experiment Environment enables the user to create, run, modify, and analyses experiments in a more convenient manner. It can also be run from the command line using the Simple CLI.



New Experiment:

After clicking on new default parameters for an Experiment are defined.

We can choose the experiment in two different modes:

1. Simple
2. Advanced

Simple Mode:

1. Result Destination:
   By default, an ARFF file is the destination for the results output. But we can also choose CSV file as the destination for output file. The advantage of ARFF or CSV files is that they can be created without any additional classes. The drawback is the lack of ability to resume the interrupted experiment.

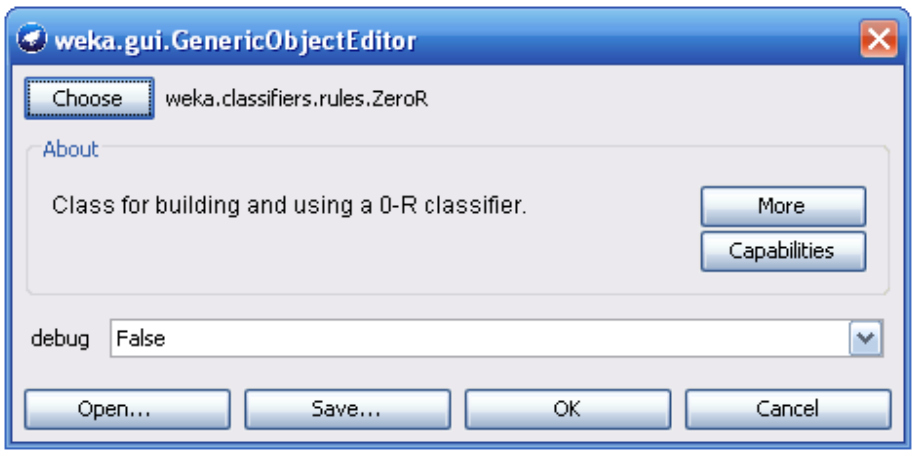
2. Experiment type:
   The user can choose between the following three different types:
   1. Cross-validation:
      It is a default type and it performs stratified cross-validation with the given number of folds.
   2. Train/Test Percentage Split:
      It splits a dataset according to the given percentage into a train and a test file after the order of the data has been randomized and stratified.
   3. Train/Test Percentage Split:
      As it is impossible to specify an explicit train/test files pair, one can abuse this type to un-merge previously merged train and test file into the two original files.

   Additionally, one can choose between Classification and Regression, depending on the datasets and classifiers one uses.

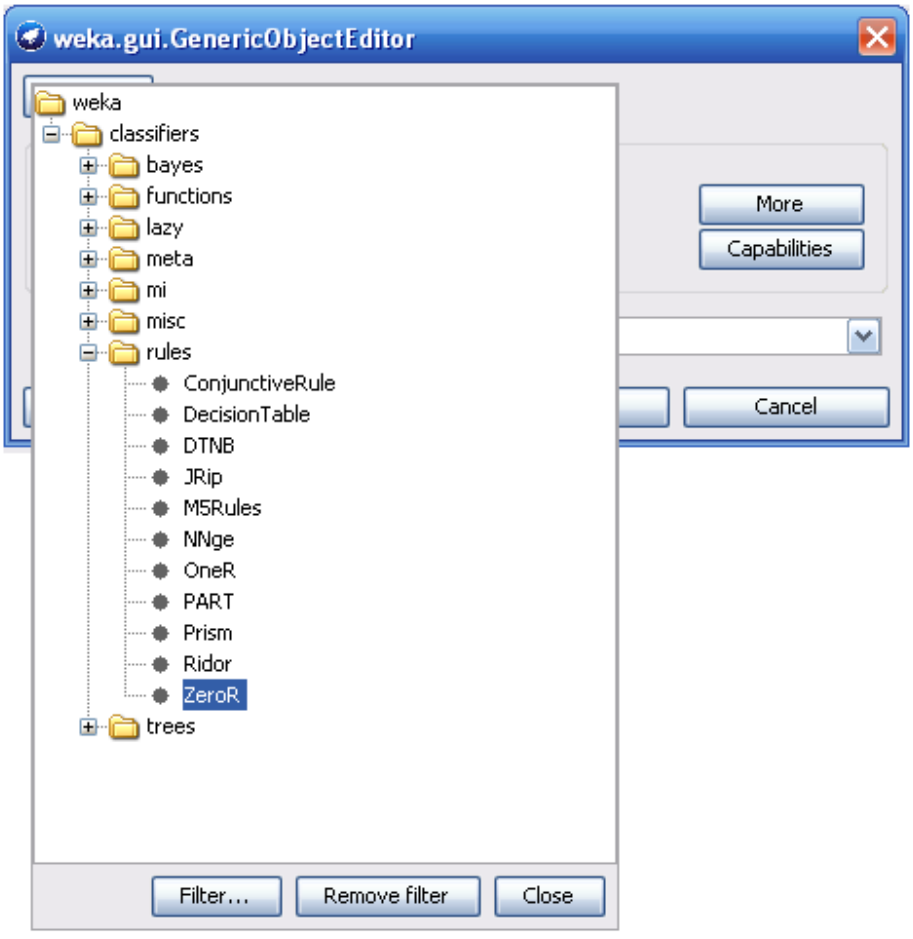
3. Data Sets:
   One can add dataset files either with an absolute path or with a relative path.

4. Iteration control:
   1. Number of repetitions:
      In order to get statistically meaningful results, the default number of iterations is 10.

   2. Data sets first/Algorithms first:
      As soon as one has more than one dataset and algorithm, it can be useful to switch from datasets being iterated over first to algorithms.
   3. Algorithms:
      New algorithms can be added via the “Add New” button. Opening this dialog for the first time, ZeroR is presented.



By clicking on the Choose button one can choose another classifier which is as shown in the below diagram:

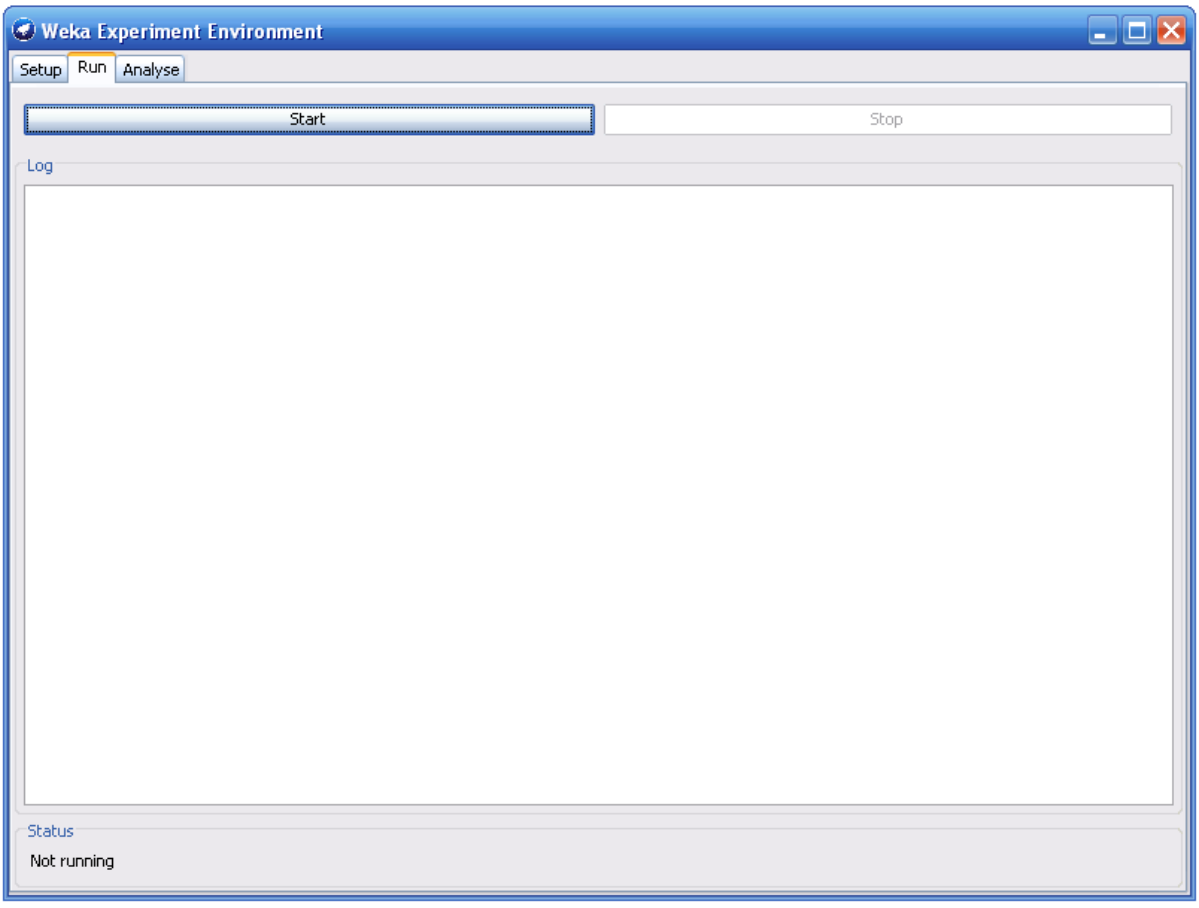


The “Filter...” button enables us to highlight classifiers that can handle certain attributes and class types. With “Remove Filter” button one can clear the classifiers that are highlighted earlier.

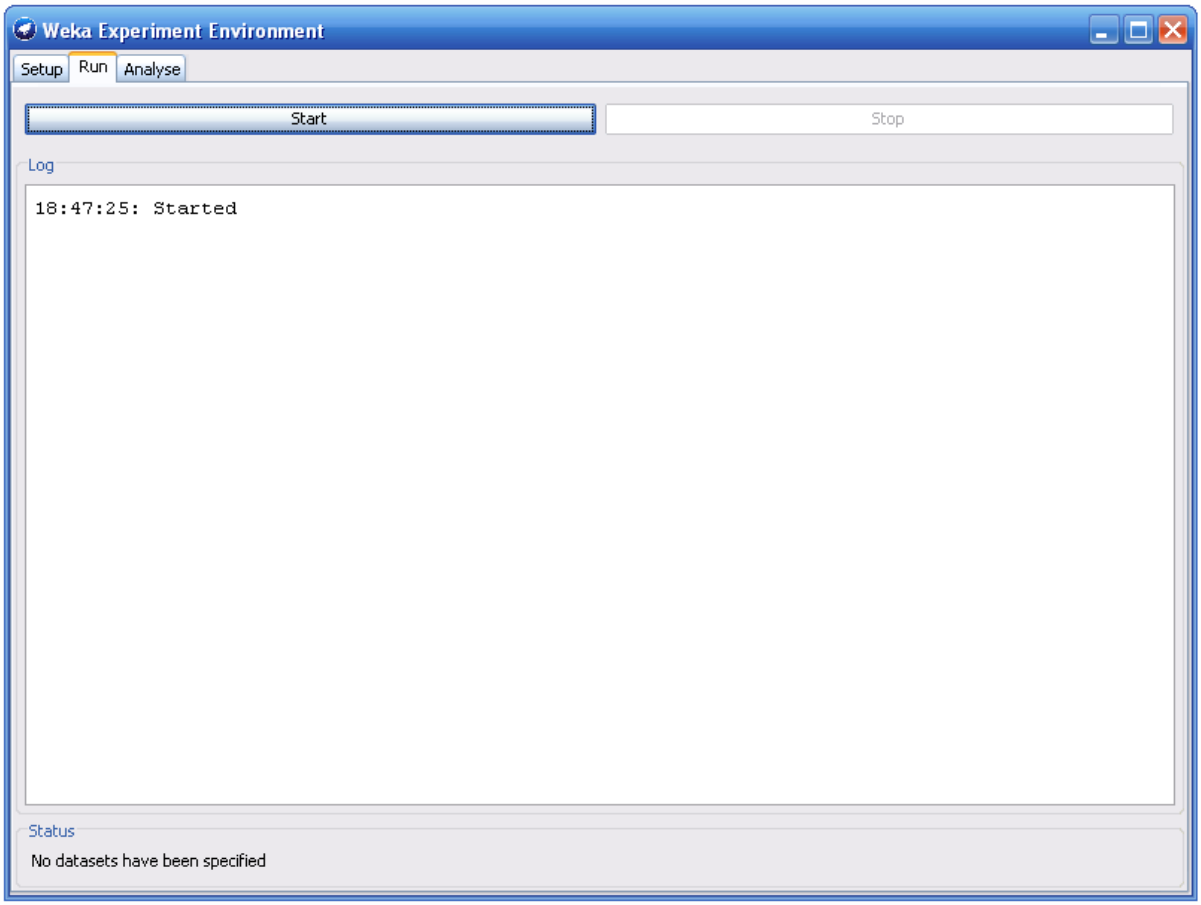
With the Load options... and Save options... buttons one can load and save the setup of a selected classifier from and to XML.

Running an Experiment:

To run the current experiment, click the Run tab at the top of the Experiment Environment window. The current experiment performs 10 runs of 10-fold stratified cross-validation.



After clicking the Run tab, it shows a window with start button and stop button, by clicking on start button we can run the experiment and by clicking on stop button we can run the experiment.



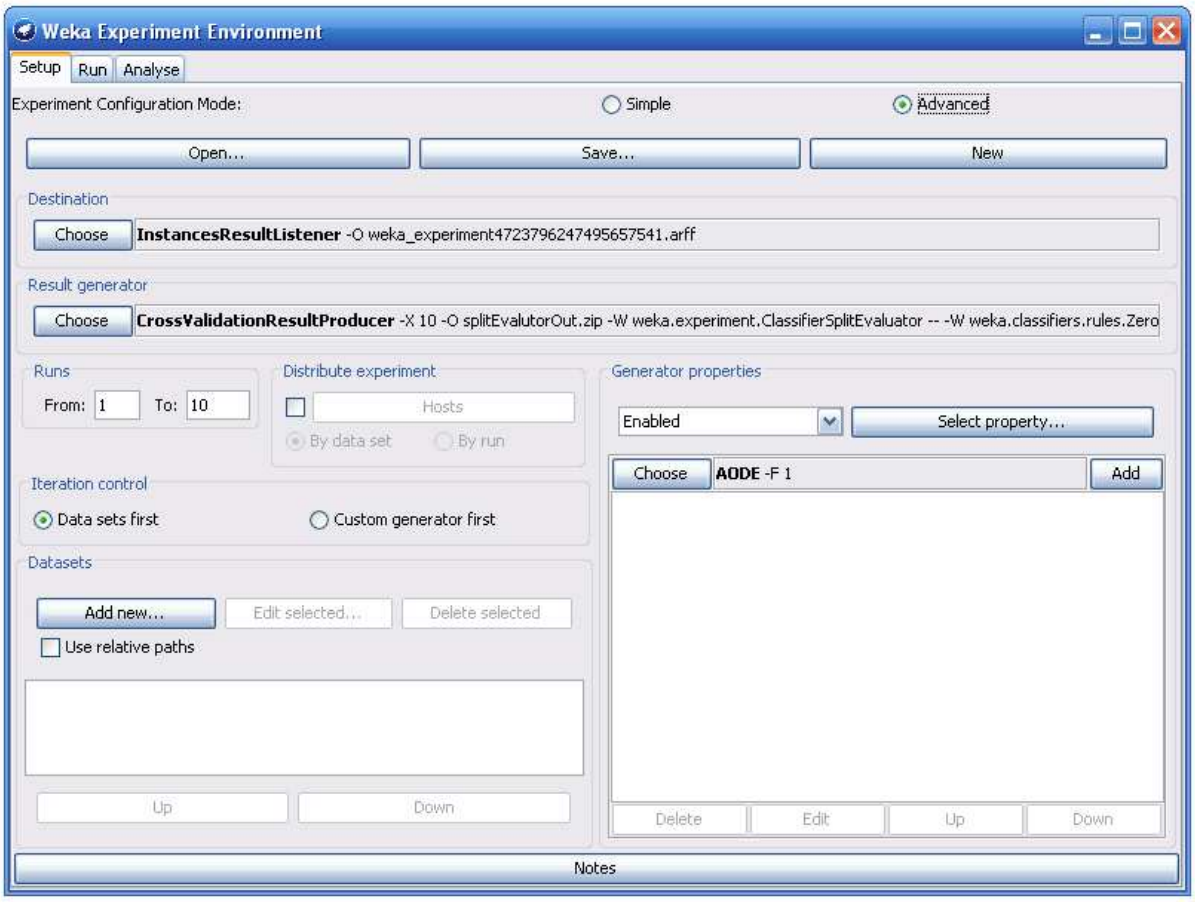
If the experiment was defined correctly, the 3 messages shown above will be displayed in the Log panel.

Advanced Mode:

Defining an experiment:

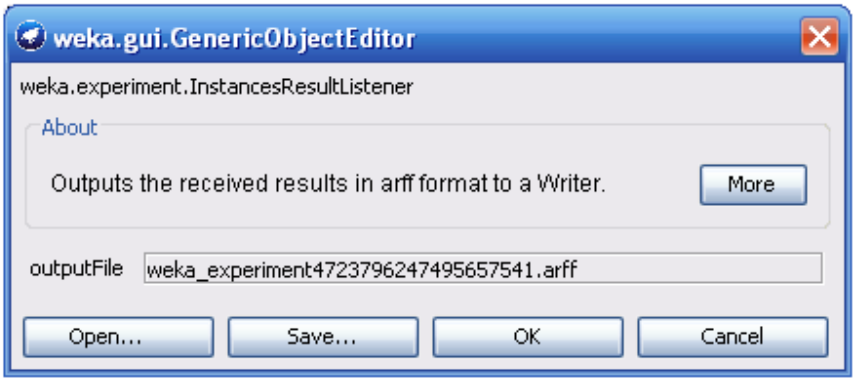
When the Experimenter is started in Advanced mode, the Setup tab is displayed. Now click New to initialize an experiment.

To define the dataset to be processed by a scheme, first select Use relative paths in the Datasets panel of the Setup tab and then click on Add new... button.

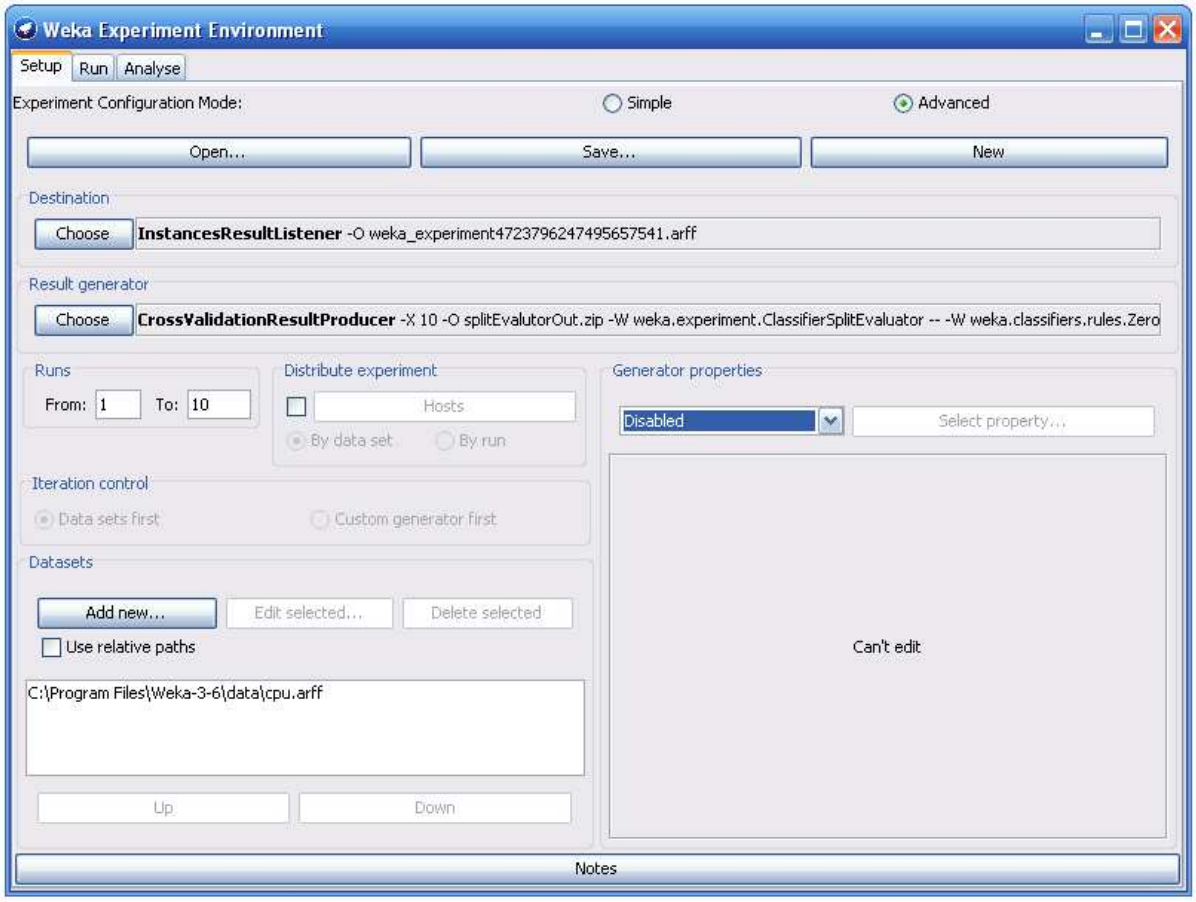


Saving Results of the experiment

To identify a dataset to which the results are to be sent, click on the Instances- ResultListener entry in the Destination panel, which opens a dialog box with a label named as “output file”.



Now give the name of the output file and click on OK button. The dataset name is now displayed in the Datasets panel of the Setup tab. This is as shown in the following figure:



Now we can run the experiment by clicking the Run tab at the top of the experiment environment window. The current experiment performs 10 randomized train and test runs.

To change from random train and test experiments to cross-validation experiments, click on the Result generator entry.

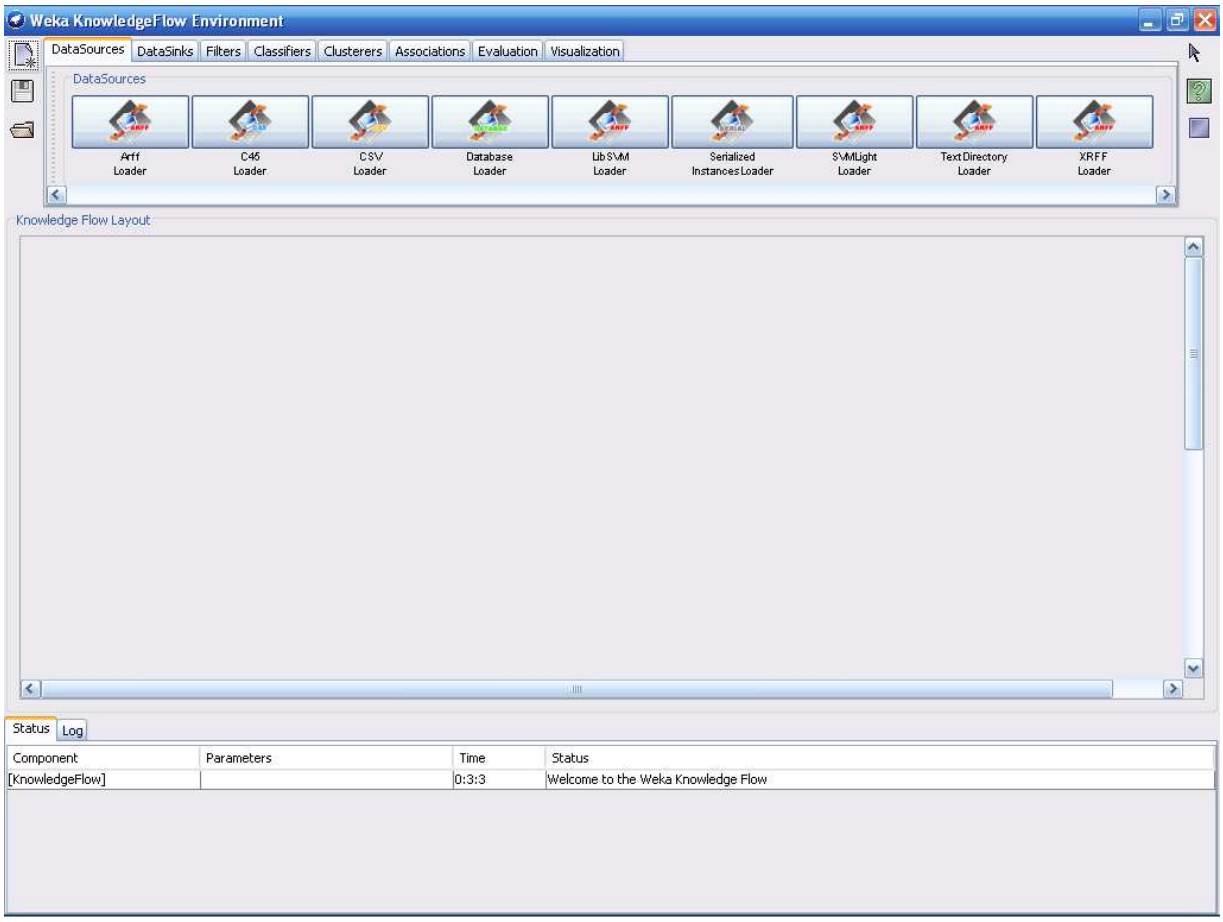
Using analysis tab in experiment environment window one can analyze the results of experiments using experiment analyzer.

KNOWLEDGE FLOW

The Knowledge Flow provides an alternative to the Explorer as a graphical front end to WEKA's core algorithms. It is represented as shown in the following figure. The Knowledge Flow presents a data-flow inspired interface to WEKA.

The Knowledge Flow offers the following features:

1. Intuitive data flow style layout.
2. Process data in batches or incrementally.
3. Process multiple batches or streams in parallel (each separate flow executes in its own thread).
4. Chain filters together.
5. View models produced by classifiers for each fold in a cross validation.
6. Visualize performance of incremental classifiers during processing
7. Plug-in facility for allowing easy addition of new components to the Knowledge Flow.



Components

The components are

1. Data Sources: All WEKA loaders are available.
2. Data Sinks: All WEKA savers are available.
3. Filters: All WEKA's filters are available.
4. Classifiers: All WEKA classifiers are available.
5. Clusterers: All WEKA clusterers are available.
6. Evaluation: It contains different kinds of techniques like
    - TrainingSetMaker,
    - TestSetMaker,
    - CrossValidationFoldMaker,
    - TrainTestSplitMaker,
    - ClassAssigner
    - ClassValuePicker,
    - ClassifierPerformanceEvaluator,
    - IncrementalClassifierEvaluator,
    - ClustererPerformanceEvaluator,
    - PredictionAppender.
7. Visualization: It contains different models like
    - DataVisualizer,
    - ScatterPlotMatrix
    - AttributeSummarizer
    - ModelPerformanceChart,
    - TextViewer
    - GraphViewerbased,
    - StripChart.

Plug-in Facility:

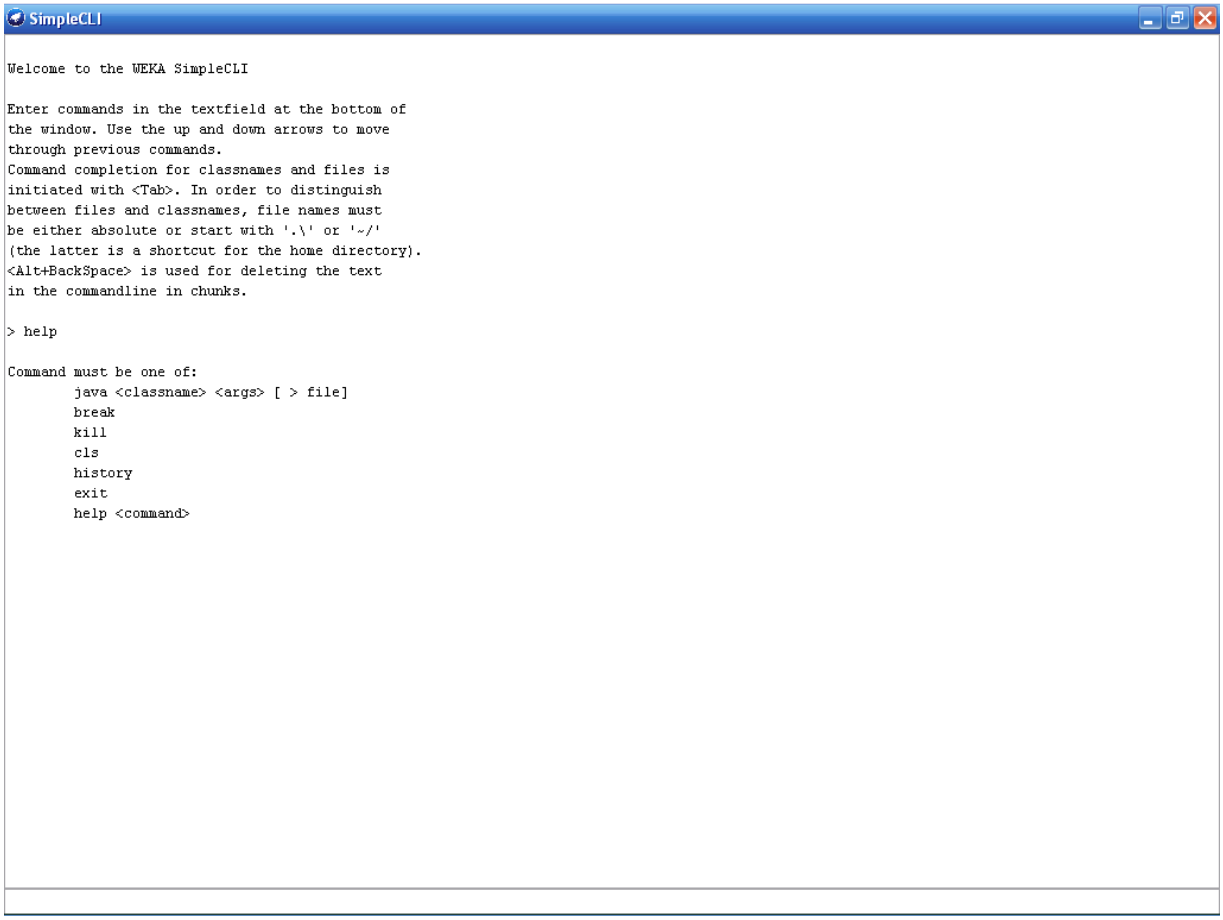
The Knowledge Flow offers the ability to easily add new components via a plug-in mechanism.

SIMPLE CLI

The Simple CLI provides full access to all Weka classes like classifiers, filters, clusterers, etc., but without the hassle of the CLASSPATH.

It offers a simple Weka shell with separated command line and output.

The simple command line interface is represented as shown in the following figure:



The following commands are available in the Simple CLI:

- Java <classname> [<args>]

  It invokes a java class with the given arguments (if any)
- Break

  It stops the current thread, e.g., a running classifier, in a friendly manner
- Kill

  it stops the current thread in an unfriendly fashion
- Cls

It clears the output area

- Exit

  It exits the Simple CLI

- Help [<command>]

  It provides an overview of the available commands if without a command name as argument, otherwise more help on the specified command

In order to invoke a Weka class, only the way is one has to prefix the class with"java". This command tells the Simple CLI to load a class and execute it with any given parameters.

TYPES OF FILES

AREF file

Attribute Relationship File Format (ARFF) is the text format file used by Weka to store data in a database. This kind of file is structured as follows ("weather" relational database).

The ARFF file contains two sections: the header and the data section. The first line of the header tells us the relation name. Then there is the list of the attributes (@attribute...). Each attribute is associated with a unique name and a type.

The @relation Declaration

The relation name is defined as the first line in the ARFF file. The format is:

@relation <relation-name>

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

The @attribute Declarations

The latter describes the kind of data contained in the variable and what values it can have. The variables types are: numeric, nominal, string and date

The format for the @attribute statement is:

@attribute <attribute-name> <datatype>

Attribute declarations take the form of an orderd sequence of @attribute statements. Each attribute in the data set has its own @attribute statement which uniquely defines the name of that attribute and it's data type. The order the attributes are declared indicates the column position in the data section of the file. For example, if an attribute is the third one declared then Weka expects that all that attributes values will be found in the third comma delimited column.

Missing values are represented by a single question mark, as in:

```
@data
4.4,?,1.5,?,Iris-setosa
```

Dates must be specified in the data section using the string representation specified in the attribute declaration. For example:

```
@RELATION Timestamps
```

@ATTRIBUTE timestamp DATE "yyyy-MM-dd HH:mm:ss"

@DATA
"2001-04-03 12:12:12"
"2001-05-03 12:59:55"

The class attribute is by default the last one of the list. In the header section there can also be some comment lines, identified with a '%' at the beginning, which can describe the database content or give the reader information about the author. After that there is the data itself (@data), each line stores the attribute of a single entry separated by a comma.

CSV File:

Comma-separated values (CSV) file stores tabular data (numbers and text) in plain-text form. As a result, such a file is easily human-readable (e.g., in a text editor).

CSV is a simple file format that is widely supported by consumer, business, and scientific applications. Among its most common uses is to move tabular data between programs that naturally operate on a more efficient or complete proprietary format. For example: a CSV file might be used to transfer information from a database program to a spreadsheet.

EXPERIMENT 1: Create an ARFF file with the following data.

No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	**play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes

PROCEDURE:

1. Select Start button → All Programs →Accessories → Note Pad.

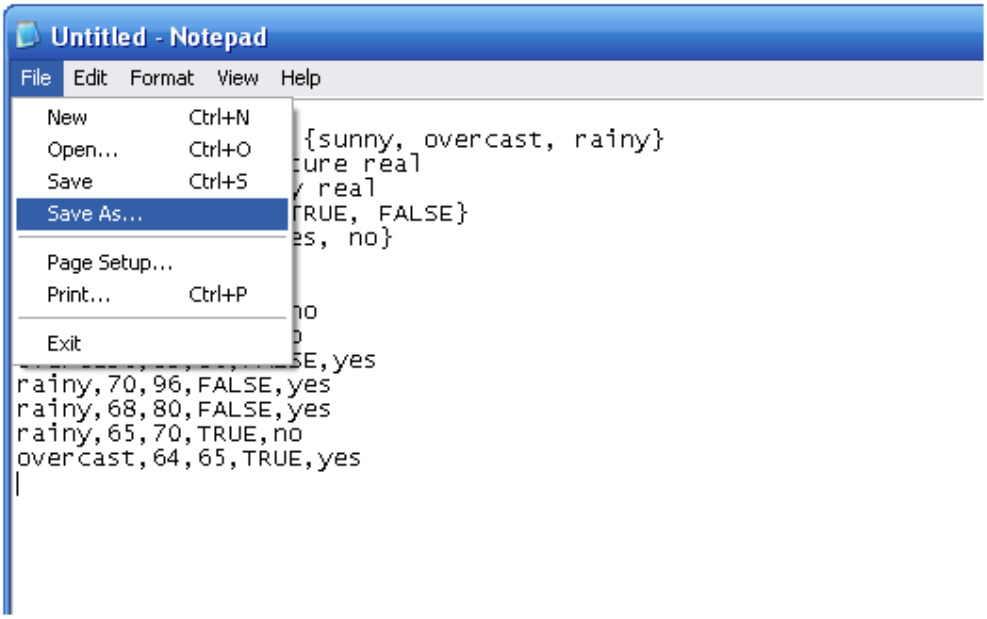
2. Enter the below code in Notepad editor.

@relation weather

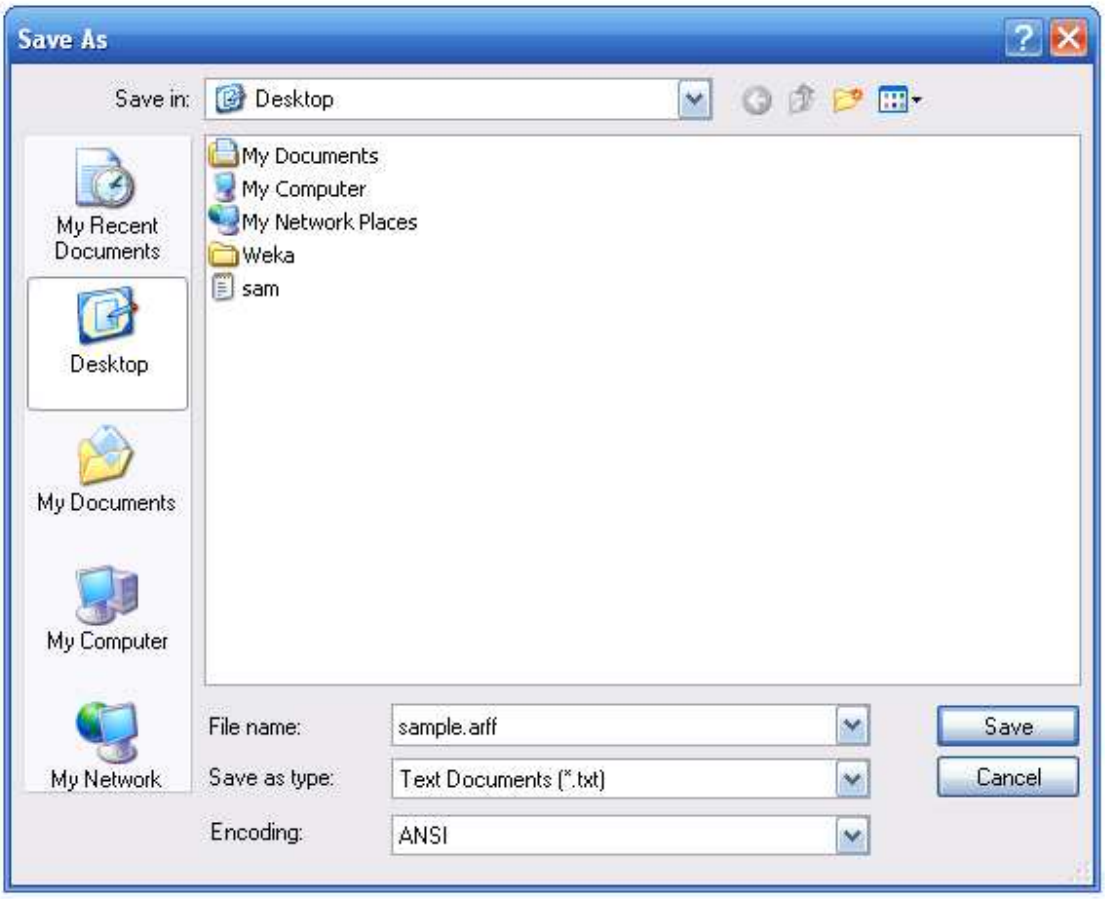
@attribute outlook {sunny, overcast, rainy}
@attribute temperature real
@attribute humidity real
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
rainy,68,80,FALSE,yes
rainy,65,70,TRUE,no
overcast,64,65,TRUE,yes

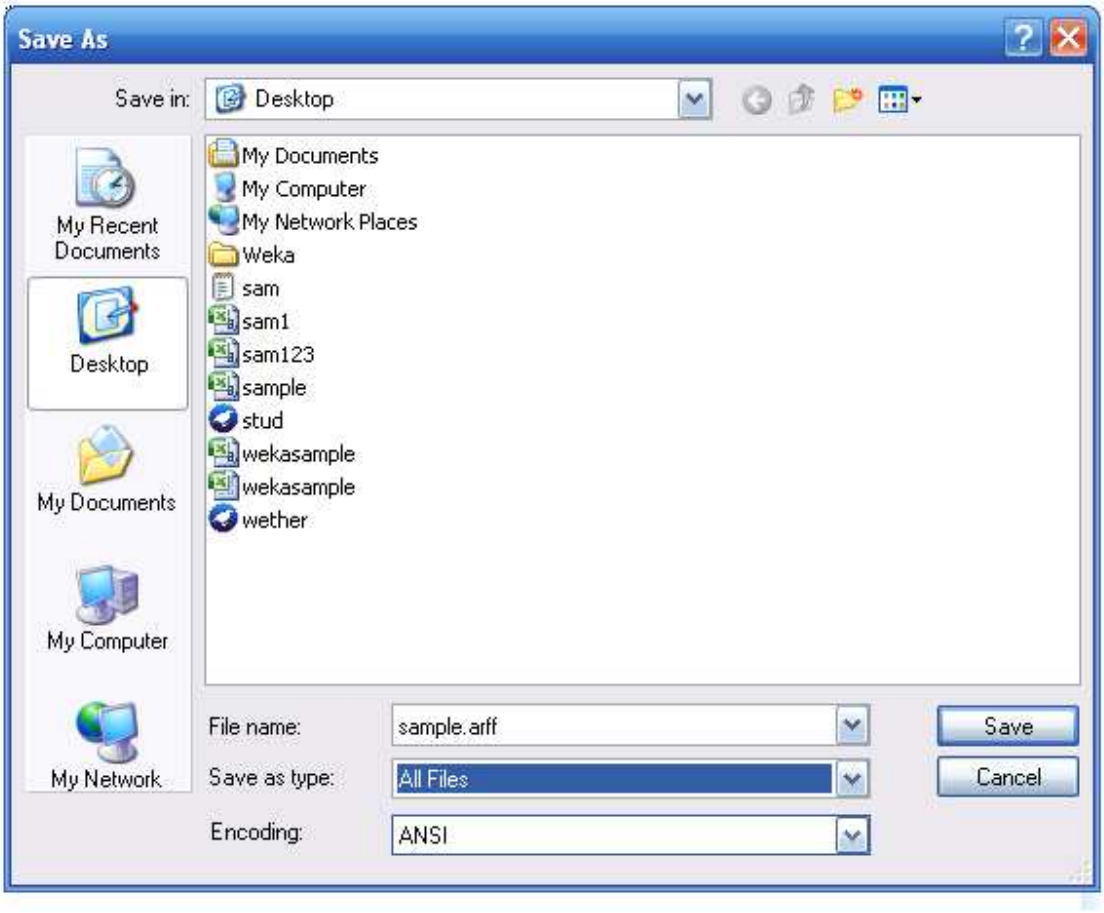
3. Select File Menu → Save As option.



4. In Save as Dialog box enter the filename in File Name text box with extension .arff.



5. Click Save as type list box and select All Files from drop down list box.



6. Click on Save Button

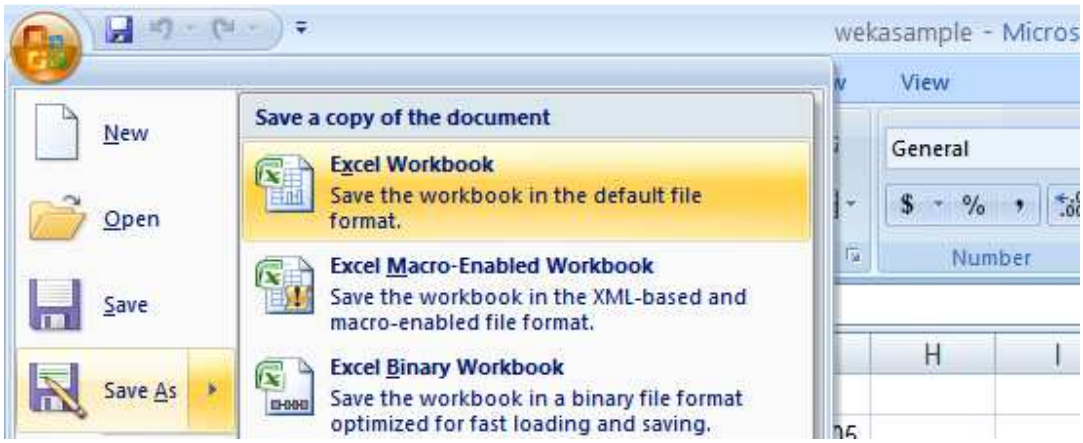
EXPERIMENT 2:

Create a CSV file with the following data from MS-Excel.

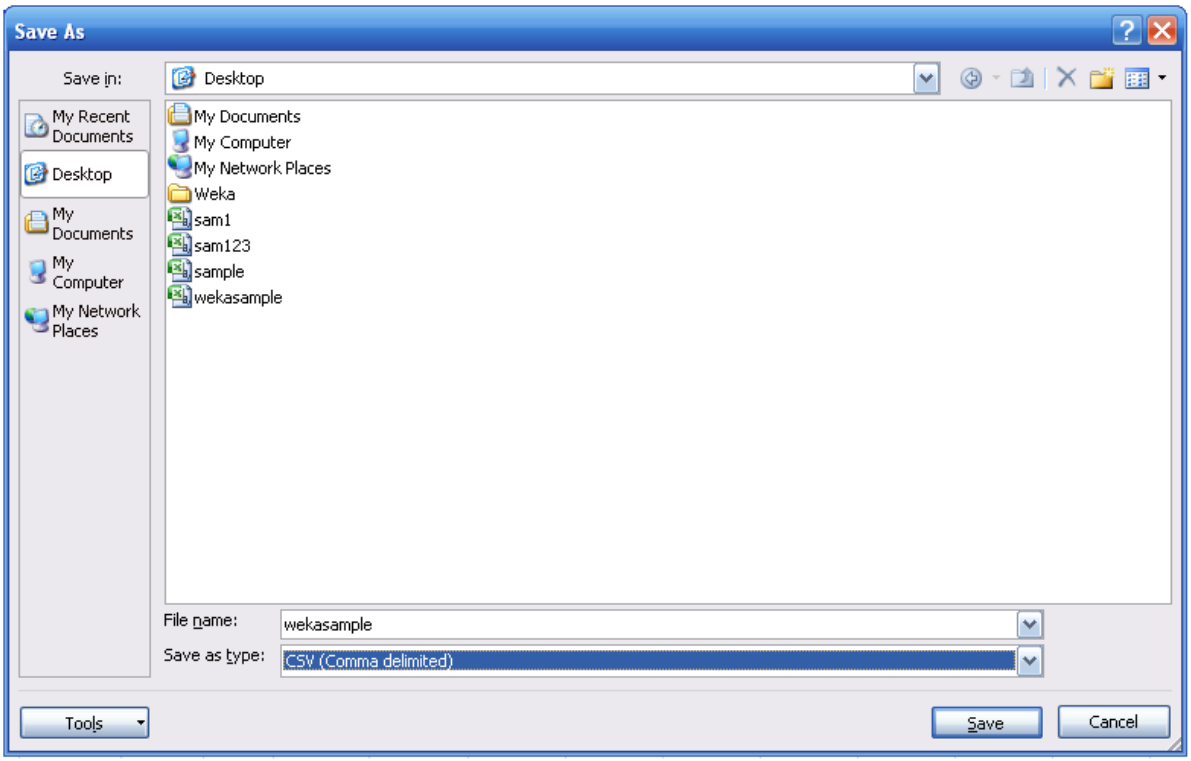
PROCEDURE:

1. Select start button → All Programs → Microsoft Office → Microsoft Office Excel 2007.

2. Enter the data into Excel Spread sheet.
3. You can convert an Excel worksheet to a text file by using the Save As command or click the Microsoft Office Button , and then click Save As.
4.



5. The Save As dialog box appears.
6. Enter the filename in File name box
7. In the Save as type box, choose the text file format for the worksheet.
   For example, click Text (Tab delimited) or CSV (Comma delimited).



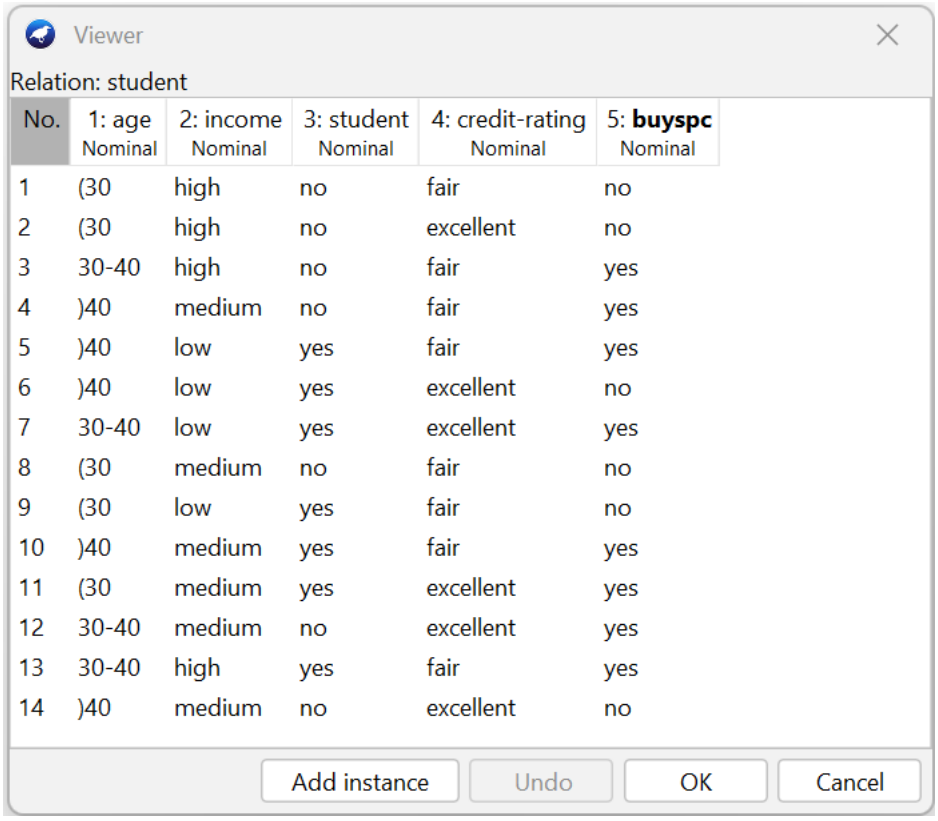
8. Click on Save Button.

Experiment 1:
Aim: Demonstration of pre-processing on dataset student.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


Viewer
Relation: student
No.
1: age
Nominal
2: income
Nominal
3: student
Nominal
4: credit-rating
Nominal
5: buyspc
Nominal
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
Add instance
Undo
OK
Cancel

Experiment 2:

Aim: Implementation of pre-processing on dataset labor.arff.

@relation labor
@attribute 'duration' real
@attribute 'wage-increase-first-year' real
@attribute 'wage-increase-second-year' real
@attribute 'wage-increase-third-year' real
@attribute 'cost-of-living-adjustment' {'none','tcf','tc'}
@attribute 'working-hours' real
@attribute 'pension' {'none','ret_allw','empl_contr'}
@attribute 'standby-pay' real
@attribute 'shift-differential' real
@attribute 'education-allowance' {'yes','no'}
@attribute 'statutory-holidays' real
@attribute 'vacation' {'below_average','average','generous'}
@attribute 'longterm-disability-assistance' {'yes','no'}
@attribute 'contribution-to-dental-plan' {'none','half','full'}
@attribute 'bereavement-assistance' {'yes','no'}
@attribute 'contribution-to-health-plan' {'none','half','full'}
@attribute 'class' {'bad','good'}

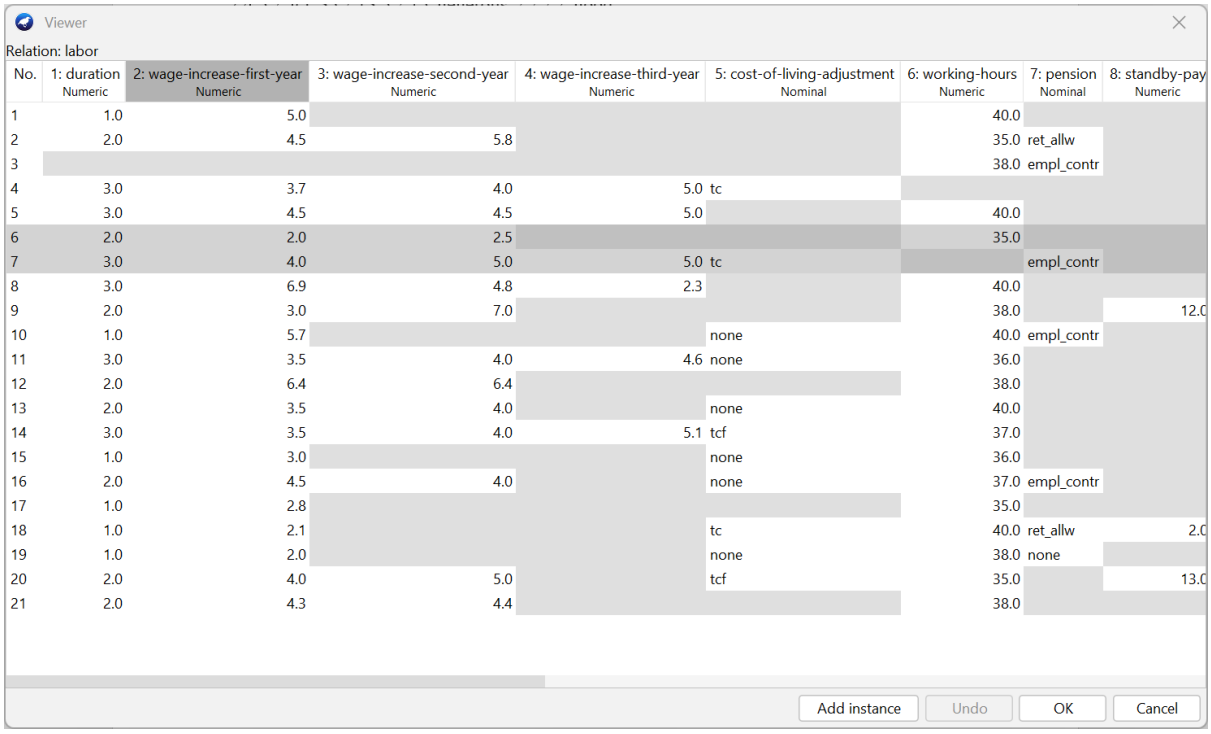
@data
1,5,?,?,?,40,?,?,2,?,11,'average',?,?,'yes',?,'good'
2,4.5,5.8,?,?,35,'ret_allw',?,?,'yes',11,'below_average',?,'full',?,'full','good'
?,?,?,?,?,38,'empl_contr',?,5,?,11,'generous','yes','half','yes','half','good'
3,3.7,4,5,'tc',?,?,?,?,'yes',?,?,?,?,'yes',?,'good'
3,4.5,4.5,5,?,40,?,?,?,?,12,'average',?,'half','yes','half','good'
2,2,2.5,?,?,35,?,?,6,'yes',12,'average',?,?,?,?,'good'
3,4,5,5,'tc',?,'empl_contr',?,?,?,12,'generous','yes','none','yes','half','good'
3,6.9,4.8,2.3,?,40,?,?,3,?,12,'below_average',?,?,?,?,'good'
2,3,7,?,?,38,?,12,25,'yes',11,'below_average','yes','half','yes',?,'good'
1,5.7,?,?,'none',40,'empl_contr',?,4,?,11,'generous','yes','full',?,?,'good'
3,3.5,4,4.6,'none',36,?,?,3,?,13,'generous',?,?,'yes','full','good'
2,6.4,6.4,?,?,38,?,?,4,?,15,?,?,'full',?,?,'good'
2,3.5,4,?,'none',40,?,?,2,'no',10,'below_average','no','half',?,'half','bad'
3,3.5,4,5.1,'tcf',37,?,?,4,?,13,'generous',?,'full','yes','full','good'
1,3,?,?,'none',36,?,?,10,'no',11,'generous',?,?,?,?,'good'
2,4.5,4,?,'none',37,'empl_contr',?,?,?,11,'average',?,'full','yes',?,'good'
1,2.8,?,?,?,35,?,?,2,?,12,'below_average',?,?,?,?,'good'

1,2.1,?,?,'tc',40,'ret_allw',2,3,'no',9,'below_average','yes','half',?,'none','bad'

1,2,?,?,'none',38,'none',?,?,'yes',11,'average','no','none','no','none','bad'

2,4,5,?,'tcf',35,?,13,5,?,15,'generous',?,?,?,?,'good'

2,4.3,4.4,?,?,38,?,?,4,?,12,'generous',?,'full',?,'full','good'



No.	1: duration Numeric	2: wage-increase-first-year Numeric	3: wage-increase-second-year Numeric	4: wage-increase-third-year Numeric	5: cost-of-living-adjustment Nominal	6: working-hours Numeric	7: pension Nominal	8: standby-pay Numeric
1	1.0	5.0				40.0		
2	2.0	4.5	5.8			35.0	ret_allw	
3						38.0	empl_contr	
4	3.0	3.7	4.0	5.0	tc			
5	3.0	4.5	4.5	5.0		40.0		
6	2.0	2.0	2.5			35.0		
7	3.0	4.0	5.0	5.0	tc		empl_contr	
8	3.0	6.9	4.8	2.3		40.0		
9	2.0	3.0	7.0			38.0		12.0
10	1.0	5.7			none	40.0	empl_contr	
11	3.0	3.5	4.0	4.6	none	36.0		
12	2.0	6.4	6.4			38.0		
13	2.0	3.5	4.0		none	40.0		
14	3.0	3.5	4.0	5.1	tcf	37.0		
15	1.0	3.0			none	36.0		
16	2.0	4.5	4.0		none	37.0	empl_contr	
17	1.0	2.8				35.0		
18	1.0	2.1			tc	40.0	ret_allw	2.0
19	1.0	2.0			none	38.0	none	
20	2.0	4.0	5.0		tcf	35.0		13.0
21	2.0	4.3	4.4			38.0		

Experiment 3:

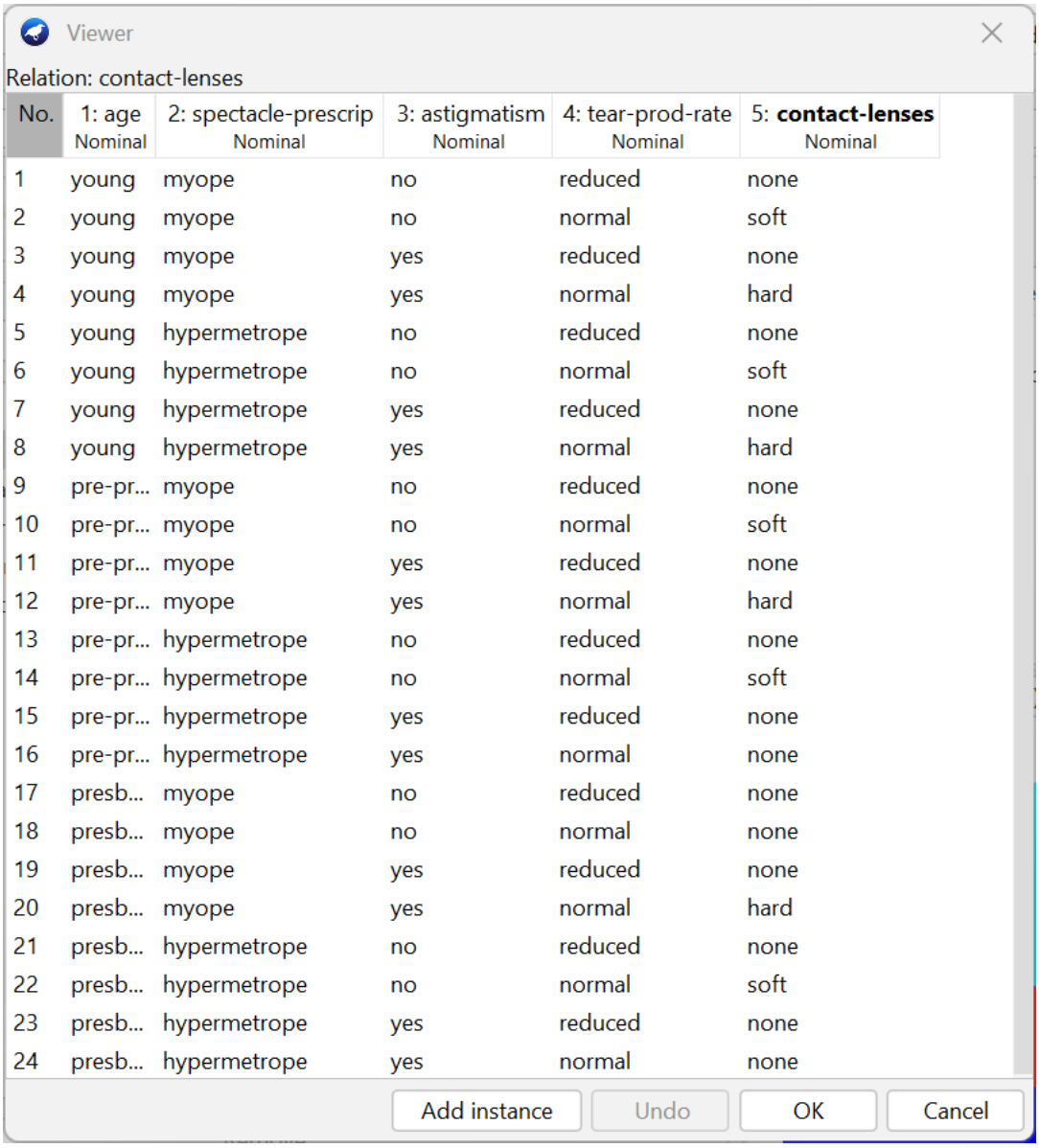
Aim: Demonstration of Association rule process on dataset contactlenses.arff using apriori Algorithm.

@relation contact-lenses

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

@data

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



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:

tear-prod-rate=reduced 12 ==> contact-lenses=none 12 <conf:(1)> lift:(1.6) lev:(0.19) [4] conv:(4.5)

spectacle-prescrip=myope tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)

spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)

astigmatism=no tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)

astigmatism=yes tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)

contact-lenses=soft 5 ==> astigmatism=no 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)

contact-lenses=soft 5 ==> tear-prod-rate=normal 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)

tear-prod-rate=normal contact-lenses=soft 5 ==> astigmatism=no 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)

astigmatism=no contact-lenses=soft 5 ==> tear-prod-rate=normal 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5)

contact-lenses=soft 5 ==> astigmatism=no tear-prod-rate=normal 5 <conf:(1)> lift:(4) lev:(0.16) [3] conv:(3.75)

Output:

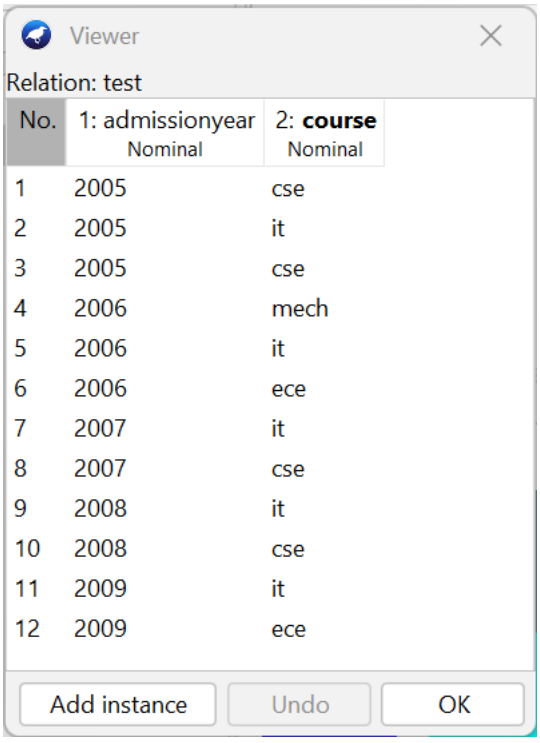
```
12:24:22 - Apriori
=== Run information ===

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
Number of cycles performed: 16

Generated sets of large itemsets:

Size of set of large itemsets L(1): 11

Size of set of large itemsets L(2): 21

Size of set of large itemsets L(3): 6

Best rules found:

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

Experiment 4:

Aim: Implement Association rule process on dataset test.arff using apriori algorithm.

@relation test
@attribute admissionyear {2005,2006,2007,2008,2009,2010}
@attribute course {cse,mech,it,ece}
@data

2005, cse
2005, it
2005, cse
2006, mech
2006, it
2006, ece
2007, it
2007, cse
2008, it
2008, cse
2009, it
2009, ece



=== 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: test
Instances: 12
Attributes: 2
admissionyear
course
=== Associator model (full training set) ===

Apriori
=======

Minimum support: 0.1 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18

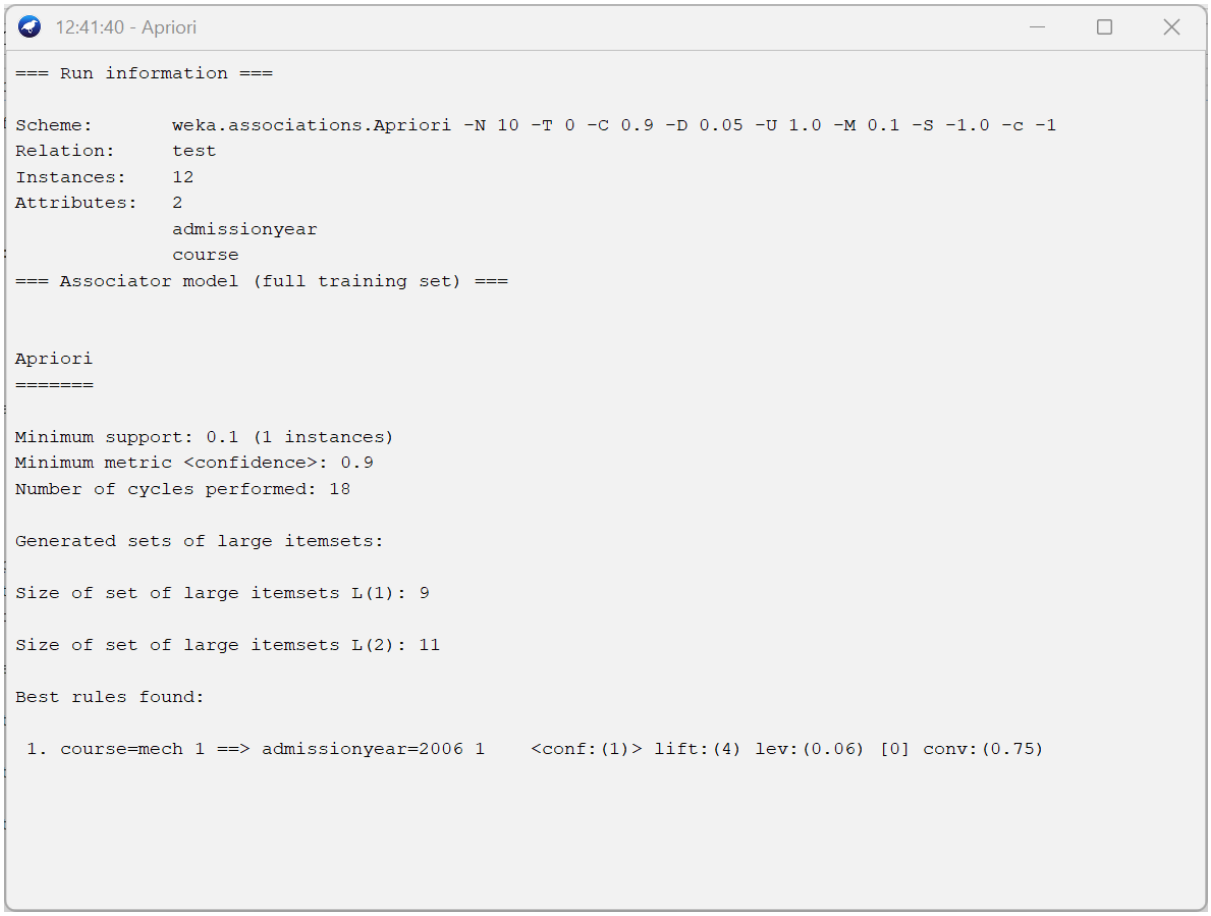
Generated sets of large itemsets:

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

Size of set of large itemsets L(2): 11

Best rules found:

1. course=mech 1 ==> admissionyear=2006 1 <conf:(1)> lift:(4) lev:(0.06) [0] conv:(0.75)


12:41:40 - Apriori
=== 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: test
Instances: 12
Attributes: 2
admissionyear
course
=== Associator model (full training set) ===
Apriori
=======
Minimum support: 0.1 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18
Generated sets of large itemsets:
Size of set of large itemsets L(1): 9
Size of set of large itemsets L(2): 11
Best rules found:
1. course=mech 1 ==> admissionyear=2006 1 <conf:(1)> lift:(4) lev:(0.06) [0] conv:(0.75)

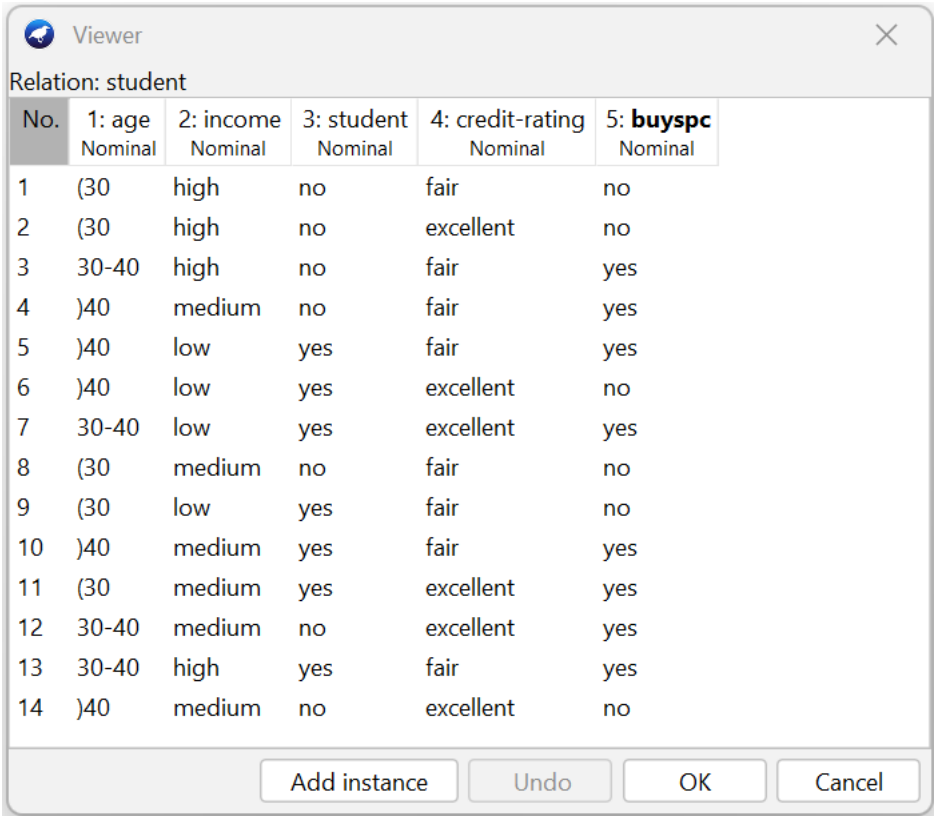
Experiment 5:

Aim: Apply classification rule process on dataset student.arff using j48 algorithm

@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



=== 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 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 ===

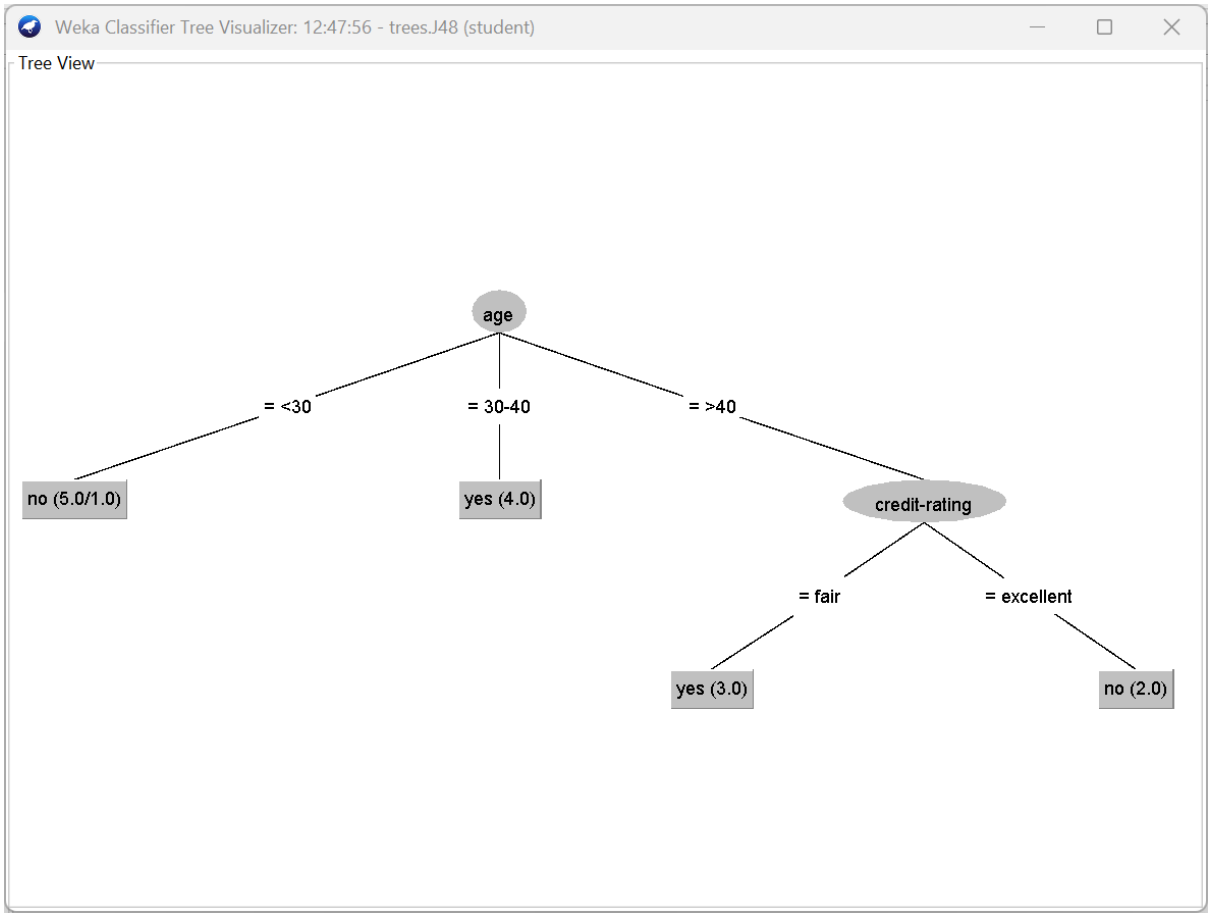
TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.875	0.333	0.778	0.875	0.824	0.559	0.854	0.919	yes
0.667	0.125	0.800	0.667	0.727	0.559	0.854	0.727	no
Weighted Avg.								
0.786	0.244	0.787	0.786	0.782	0.559	0.854	0.837	

=== Confusion Matrix ===

a	b	<-- classified as
7	1	a = yes
2	4	b = no

12:47:56 - trees.J48

```
------------------

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 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     0.919     yes
               0.667    0.125    0.800      0.667    0.727      0.559    0.854     0.727     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
```



Experiment 6:

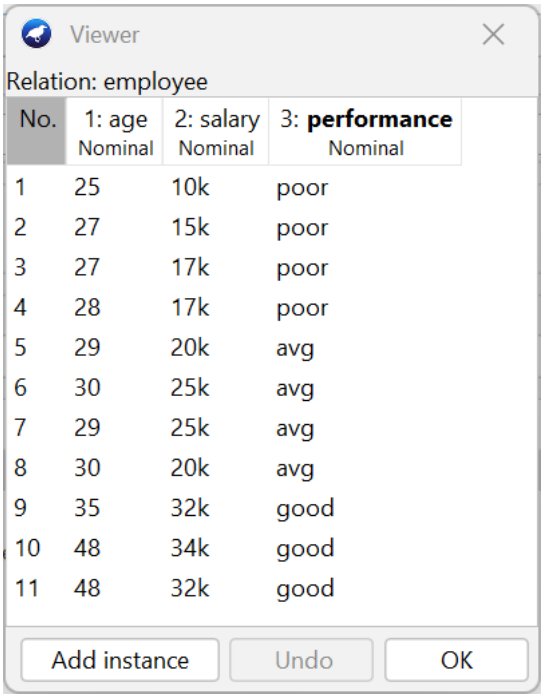
Aim: Perform classification rule process on dataset employee.arff using j48 algorithm.

It is an algorithm to generate a decision tree that is generated by C4.5 (an extension of ID3). It is also known as a statistical classifier. For decision tree classification, we need a database.

@relation employee
@attribute age {25, 27, 28, 29, 30, 35, 48}
@attribute salary{10k,15k,17k,20k,25k,30k,34k,32k}
@attribute performance {good, avg, poor}

@data

25, 10k, poor
27, 15k, poor
27, 17k, poor
28, 17k, poor
29, 20k, avg
30, 25k, avg
29, 25k, avg
30, 20k, avg
35, 32k, good
48, 34k, good
48, 32k, good



=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: employee
Instances: 11
Attributes: 3
age
salary
performance
Test mode: 10-fold cross-validation

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

J48 pruned tree
---

age = 25: poor (1.0)
age = 27: poor (2.0)
age = 28: poor (1.0)
age = 29: avg (2.0)
age = 30: avg (2.0)
age = 35: good (1.0)
age = 48: good (2.0)

Number of Leaves : 7

Size of the tree : 8

Time taken to build model: 0 seconds

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

Correctly Classified Instances	6	54.5455 %
Incorrectly Classified Instances	5	45.4545 %

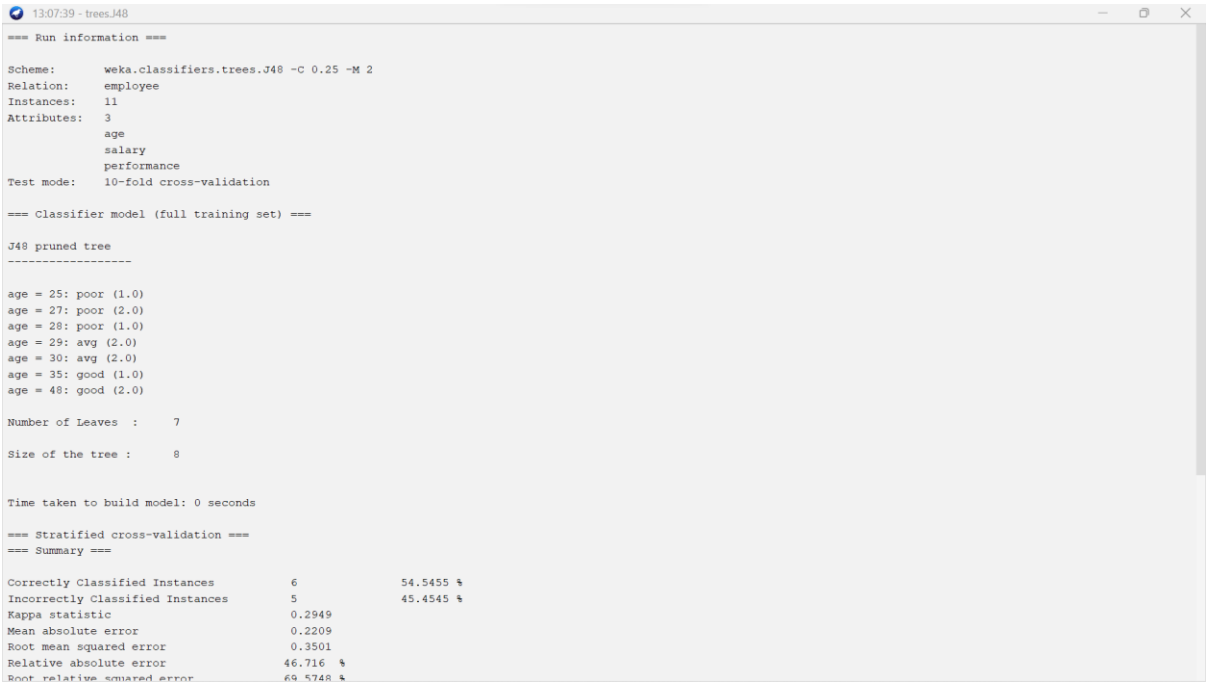
Kappa statistic	0.2949
Mean absolute error	0.2209
Root mean squared error	0.3501
Relative absolute error	46.716 %
Root relative squared error	69.5748 %
Total Number of Instances	11

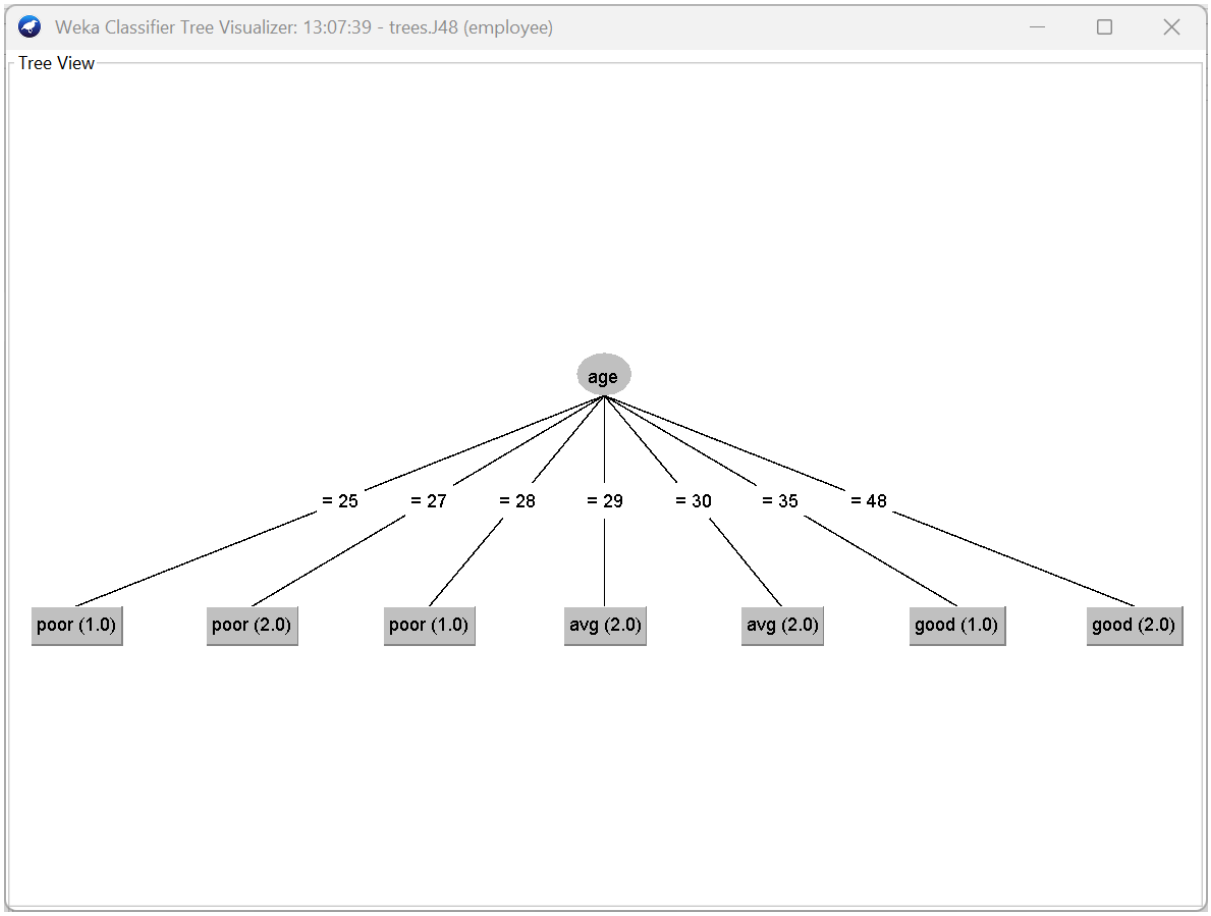
=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.333	0.000	1.000	0.333	0.500	0.516	0.771	0.633	good
	1.000	0.714	0.444	1.000	0.615	0.356	1.000	1.000	avg
	0.250	0.000	1.000	0.250	0.400	0.418	0.804	0.708	poor
Weighted Avg.	0.545	0.260	0.798	0.545	0.506	0.423	0.866	0.794	

=== Confusion Matrix ===

```
 a b c   <-- classified as
 1 2 0 | a = good
 0 4 0 | b = avg
 0 3 1 | c = poor
```




Weka Classifier Tree Visualizer: 13:07:39 - trees.J48 (employee)
Tree View
age
= 25
= 27
= 28
= 29
= 30
= 35
= 48
poor (1.0)
poor (2.0)
poor (1.0)
avg (2.0)
avg (2.0)
good (1.0)
good (2.0)

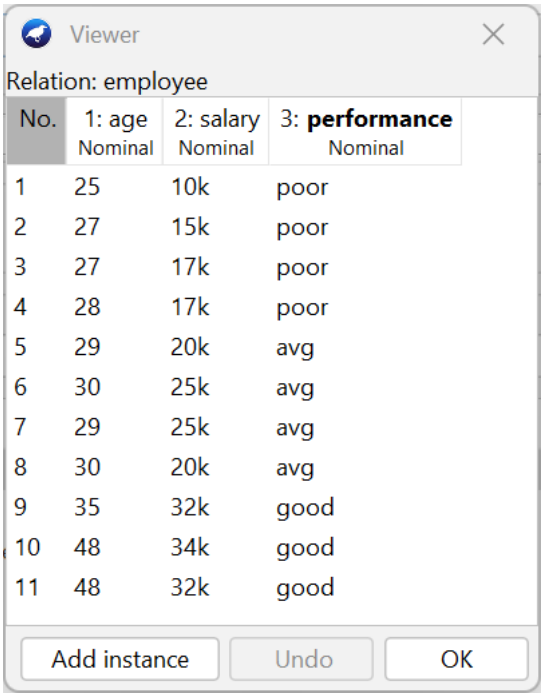
Experiment 7:

Aim: Use classification rule process on dataset employee.arff using id3 algorithm.

@relation employee
@attribute age {25, 27, 28, 29, 30, 35, 48}
@attribute salary{10k,15k,17k,20k,25k,30k,34k,32k}
@attribute performance {good, avg, poor}

@data

25, 10k, poor
27, 15k, poor
27, 17k, poor
28, 17k, poor
29, 20k, avg
30, 25k, avg
29, 25k, avg
30, 20k, avg
35, 32k, good
48, 34k, good
48, 32k, good



== Run information ===

Scheme: weka.classifiers.trees.Id3
Relation: employee
Instances: 11
Attributes: 3
age
salary
performance
Test mode: split 66.0% train, remainder test

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

Id3
age = 25: poor
age = 27: poor
age = 28: poor
age = 29: avg
age = 30: avg
age = 35: good
age = 48: good

Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.01 seconds

=== Summary ===

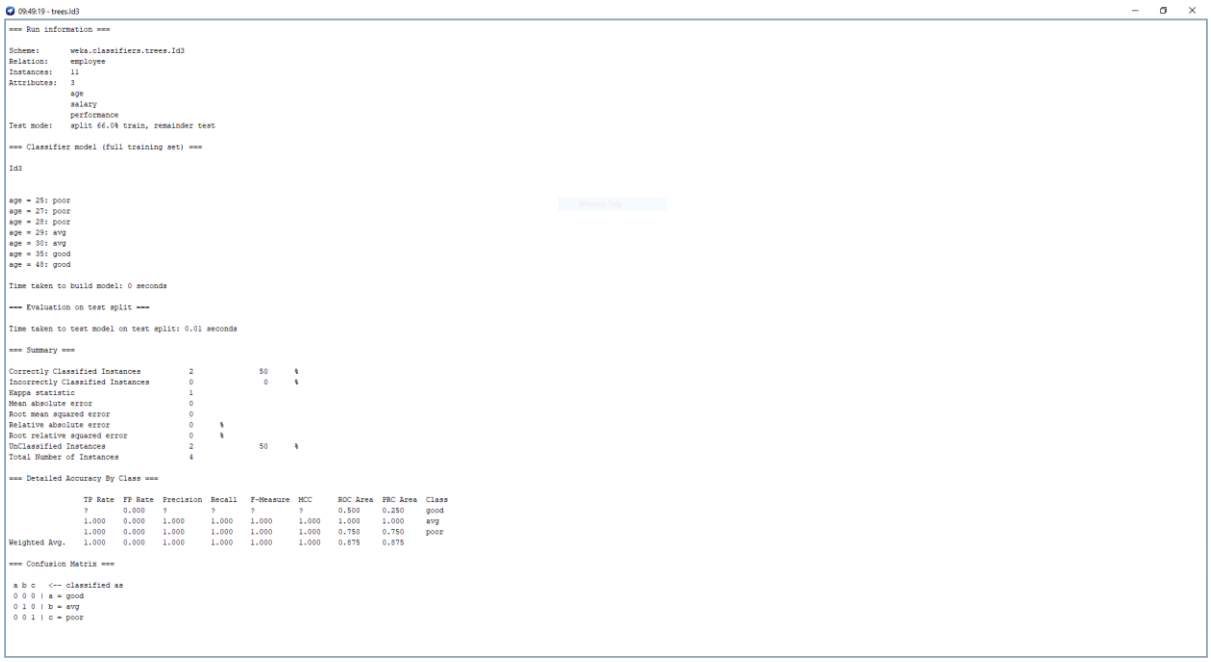
Correctly Classified Instances	2	50	%
Incorrectly Classified Instances	0	0	%
Kappa statistic	1		
Mean absolute error	0		
Root mean squared error	0		
Relative absolute error	0	%	
Root relative squared error	0	%	
UnClassified Instances	2	50	%
Total Number of Instances	4		

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
?	0.000	?	?	?	?	0.500	0.250	good
1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	avg
1.000	0.000	1.000	1.000	1.000	1.000	0.750	0.750	poor
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	0.875	0.875

=== Confusion Matrix ===

a b c <-- classified as
0 0 0 | a = good
0 1 0 | b = avg
0 0 1 | c = poor



```
09:49:19 - trees.Id3
=== Run information ===

Scheme:       weka.classifiers.trees.Id3
Relation:     employee
Instances:    11
Attributes:   3
              age
              salary
              performance
Test mode:    split 66.0% train, remainder test

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

Id3


age = 25: poor
age = 27: poor
age = 28: poor
age = 29: avg
age = 30: avg
age = 35: good
age = 45: good

Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.01 seconds

=== Summary ===

Correctly Classified Instances           2               50      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0      %
Root relative squared error              0      %
UnClassified Instances                   2               50      %
Total Number of Instances                4

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               ?        0.000    ?          ?        ?          ?        0.500     0.250     good
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     avg
               1.000    0.000    1.000      1.000    1.000      1.000    0.750     0.750     poor
Weighted Avg.  1.000    0.000    1.000      1.000    1.000      1.000    0.875     0.875

=== Confusion Matrix ===

 a b c   <-- classified as
 0 0 0 | a = good
 0 1 0 | b = avg
 0 0 1 | c = poor
```

Experiment 8:

Aim: Deploy classification rule process on dataset employee.arff using naïve bayes Algorithm.

@relation employee

@attribute age {25, 27, 28, 29, 30, 35, 48}

@attribute salary{10k,15k,17k,20k,25k,30k,34k,32k}

@attribute performance {good, avg, poor}

@data

25, 10k, poor

27, 15k, poor

27, 17k, poor

28, 17k, poor

29, 20k, avg

30, 25k, avg

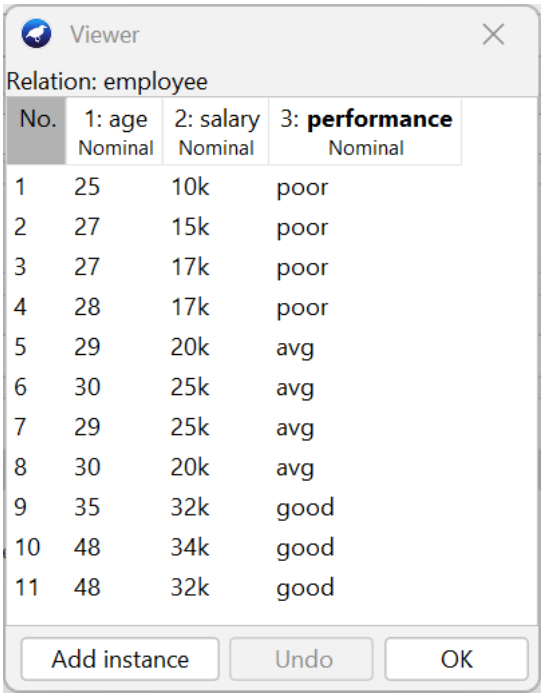
29, 25k, avg

30, 20k, avg

35, 32k, good

48, 34k, good

48, 32k, good



=== Run information ===

Scheme: weka.classifiers.bayes.NaiveBayes
Relation: employee
Instances: 11
Attributes: 3
age
salary
performance
Test mode: evaluate on training data

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

Naive Bayes Classifier

Attribute	Class good (0.29)	avg (0.36)	poor (0.36)
age			
25	1.0	1.0	2.0
27	1.0	1.0	3.0
28	1.0	1.0	2.0
29	1.0	3.0	1.0
30	1.0	3.0	1.0
35	2.0	1.0	1.0
48	3.0	1.0	1.0
[total]	10.0	11.0	11.0
salary			
10k	1.0	1.0	2.0
15k	1.0	1.0	2.0
17k	1.0	1.0	3.0
20k	1.0	3.0	1.0
25k	1.0	3.0	1.0
30k	1.0	1.0	1.0
34k	2.0	1.0	1.0
32k	3.0	1.0	1.0
[total]	11.0	12.0	12.0

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

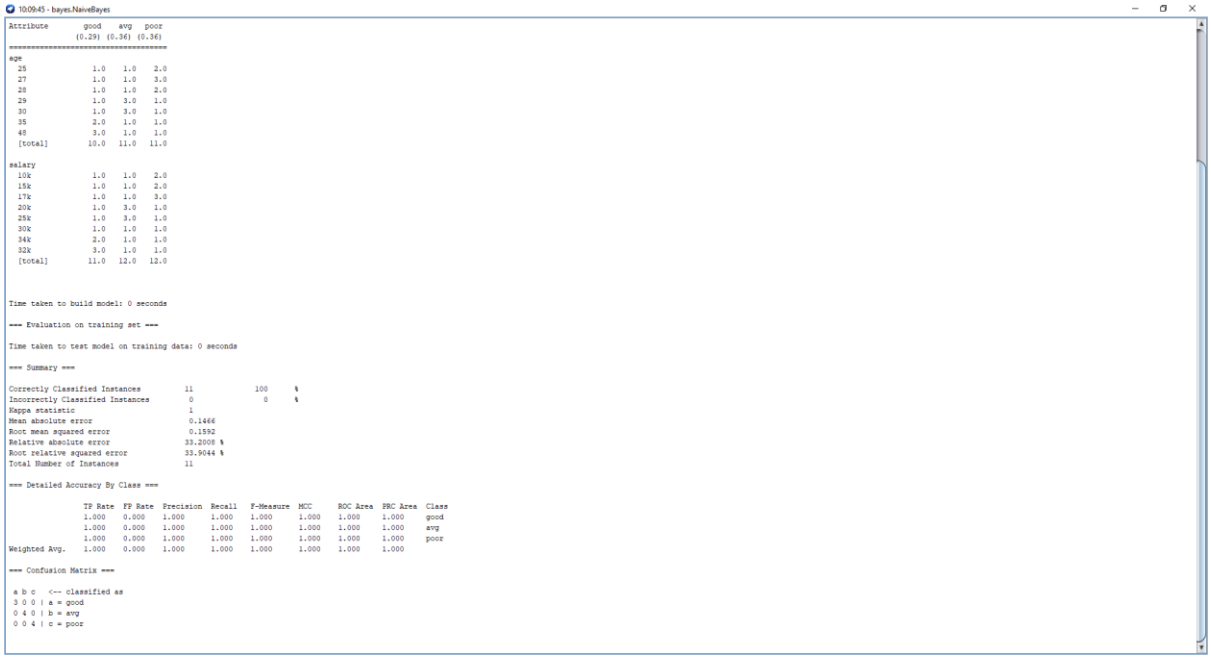
Correctly Classified Instances	11	100	%
Incorrectly Classified Instances	0	0	%
Kappa statistic	1		
Mean absolute error	0.1466		
Root mean squared error	0.1592		
Relative absolute error	33.2008 %		
Root relative squared error	33.9044 %		
Total Number of Instances	11		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	good
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	avg
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	poor
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

=== Confusion Matrix ===

```
 a b c   <-- classified as
 3 0 0 | a = good
 0 4 0 | b = avg
 0 0 4 | c = poor
```

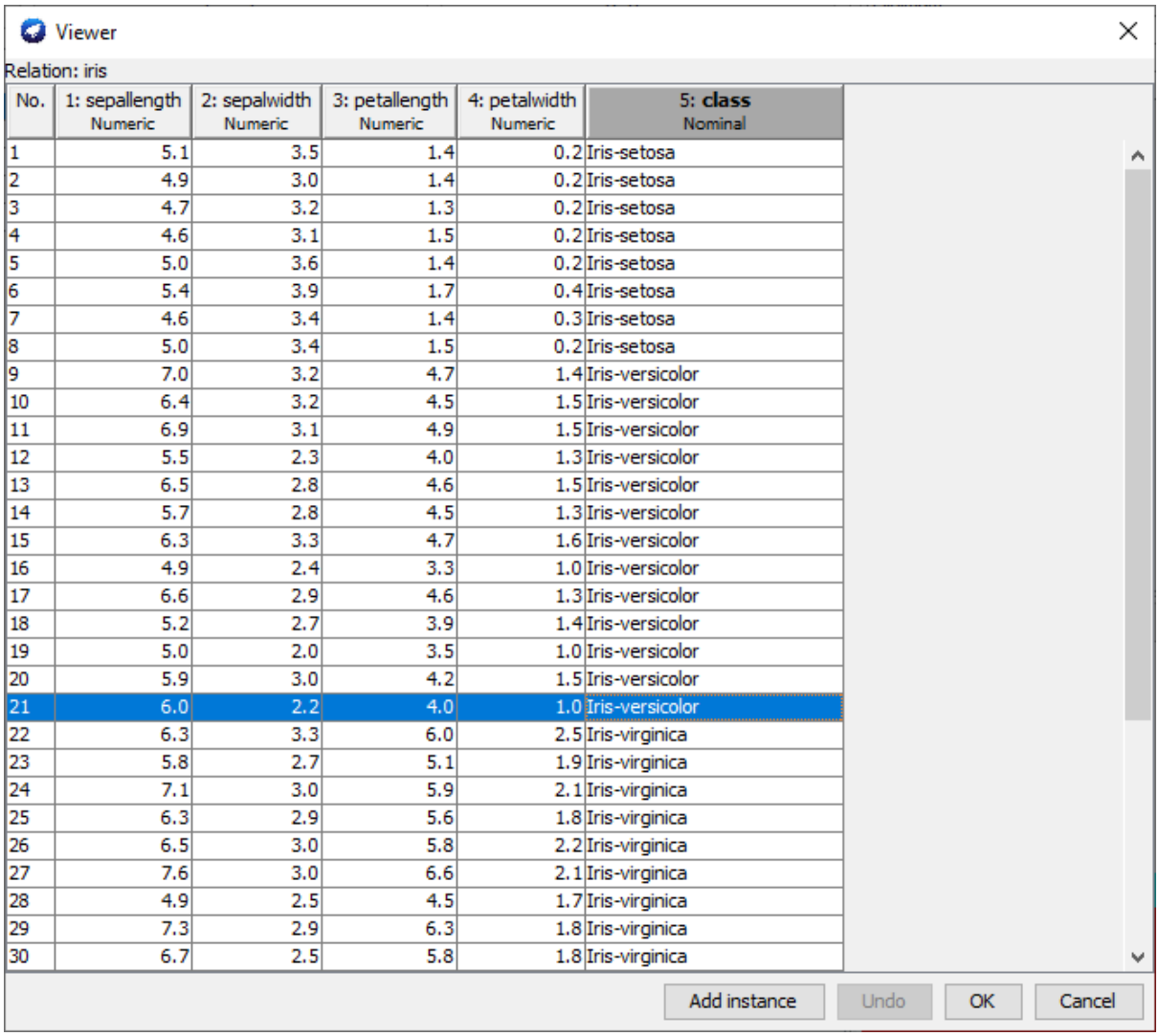


Experiment 9:

Implement clustering rule process on dataset iris.arff using simple k-means.

@RELATION iris
@ATTRIBUTE sepallength REAL
@ATTRIBUTE sepalwidth REAL
@ATTRIBUTE petallength REAL
@ATTRIBUTE petalwidth REAL
@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
5.5,2.3,4.0,1.3,Iris-versicolor
6.5,2.8,4.6,1.5,Iris-versicolor
5.7,2.8,4.5,1.3,Iris-versicolor
6.3,3.3,4.7,1.6,Iris-versicolor
4.9,2.4,3.3,1.0,Iris-versicolor
6.6,2.9,4.6,1.3,Iris-versicolor
5.2,2.7,3.9,1.4,Iris-versicolor
5.0,2.0,3.5,1.0,Iris-versicolor
5.9,3.0,4.2,1.5,Iris-versicolor
6.0,2.2,4.0,1.0,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
6.3,2.9,5.6,1.8,Iris-virginica
6.5,3.0,5.8,2.2,Iris-virginica
7.6,3.0,6.6,2.1,Iris-virginica
4.9,2.5,4.5,1.7,Iris-virginica
7.3,2.9,6.3,1.8,Iris-virginica

6.7,2.5,5.8,1.8,Iris-virginica
7.2,3.6,6.1,2.5,Iris-virginica
6.5,3.2,5.1,2.0,Iris-virginica
6.4,2.7,5.3,1.9,Iris-virginica
6.8,3.0,5.5,2.1,Iris-virginica
5.7,2.5,5.0,2.0,Iris-virginica
5.8,2.8,5.1,2.4,Iris-virginica
6.4,3.2,5.3,2.3,Iris-virginica
6.5,3.0,5.5,1.8,Iris-virginica
7.7,3.8,6.7,2.2,Iris-virginica
7.7,2.6,6.9,2.3,Iris-virginica
6.0,2.2,5.0,1.5,Iris-virginica
6.9,3.2,5.7,2.3,Iris-virginica



=== 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 "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10

Relation: iris

Instances: 42

Attributes: 5

sepallength

sepalwidth

petallength

petalwidth

class

Test mode: evaluate on training data

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

kMeans

======

Number of iterations: 2

Within cluster sum of squared errors: 15.527391849849284

Initial starting points (random):

Cluster 0: 6.4,3.2,4.5,1.5,Iris-versicolor

Cluster 1: 7.1,3,5.9,2.1,Iris-virginica

Missing values globally replaced with mean/mode

Final cluster centroids:

		Cluster#	
Attribute	Full Data	0	1
	(42.0)	(21.0)	(21.0)
sepallength	6.0786	5.581	6.5762
sepalwidth	2.9667	3	2.9333
petallength	4.4238	3.1905	5.6571
petalwidth	1.4857	0.9143	2.0571
class	Iris-virginica	Iris-versicolor	Iris-virginica

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

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

Clustered Instances

0 21 (50%)
1 21 (50%)

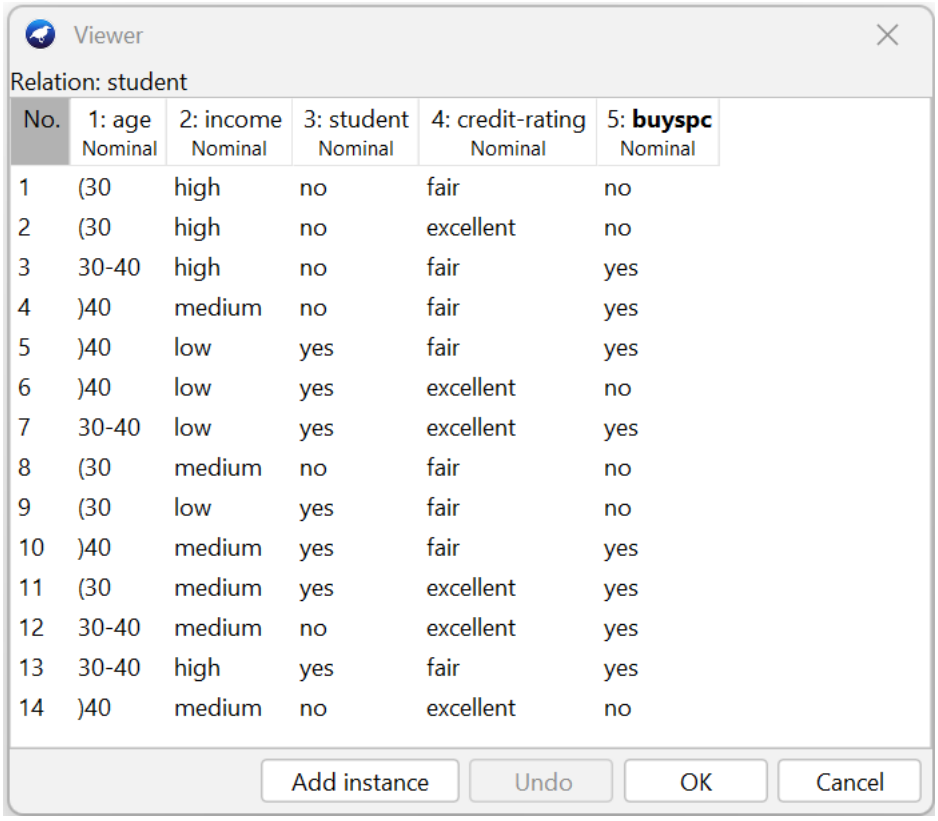
Experiment 10:

Make use of clustering rule process on dataset student.arff using simple k- means.

@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



=== 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 "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1 -S 10

Relation: student

Instances: 14

Attributes: 5

age

income

student

credit-rating

buyspc

Test mode: evaluate on training data

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

kMeans
======

Number of iterations: 5
Within cluster sum of squared errors: 26.0

Initial starting points (random):

Cluster 0: >40,medium,yes,fair,yes
Cluster 1: 30-40,low,yes,excellent,yes

Missing values globally replaced with mean/mode

Final cluster centroids:

	Cluster#		
Attribute	Full Data	0	1
	(14.0)	(9.0)	(5.0)
age	<30	<30	30-40
income	medium	medium	low
student	yes	no	yes
credit-rating	fair	fair	fair
buyspc	yes	yes	yes

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

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

Clustered Instances

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

Experiment 11:

1. Design a decision tree by pruning the nodes on your own. Convert the decision trees into "if- then-else rules". The decision tree must consists of 2-3 levels and convert it into a set of rules.

2. There also exist different classifiers that output the model in the form of rules - one such classifier in Weka is rules. PART, train this model and report the set of rules obtained. Sometimes just one attribute can be good enough in making the decision, yes, just one! Can you predict what attribute that might be in this dataset? OneR classifier uses a single attribute to make decisions (it chooses the attribute based on minimum error). Report the rule obtained by training a one R classifier. Rank the performance of j48, PART and oneR.

@relation student1

@attribute Rid numeric
@attribute Age {youth,middle_aged,senior}
@attribute Income {'high ',medium,'low '}
@attribute Student {no,yes}
@attribute Credit_rating {fair,excellent}
@attribute 'Class: Buys_computer' {no,yes}

@data
1,youth,'high ',no,fair,no
2,youth,'high ',no,excellent,no
3,middle_aged,'high ',no,fair,yes
4,senior,medium,no,fair,yes
5,senior,'low ',yes,fair,yes
6,senior,'low ',yes,excellent,no
7,middle_aged,'low ',yes,excellent,yes
8,youth,medium,no,fair,no
9,youth,'low ',yes,fair,yes
10,senior,medium,yes,fair,yes
11,youth,medium,yes,excellent,yes
12,middle_aged,medium,no,excellent,yes
13,middle_aged,'high ',yes,fair,yes
14,senior,medium,no,excellent,no

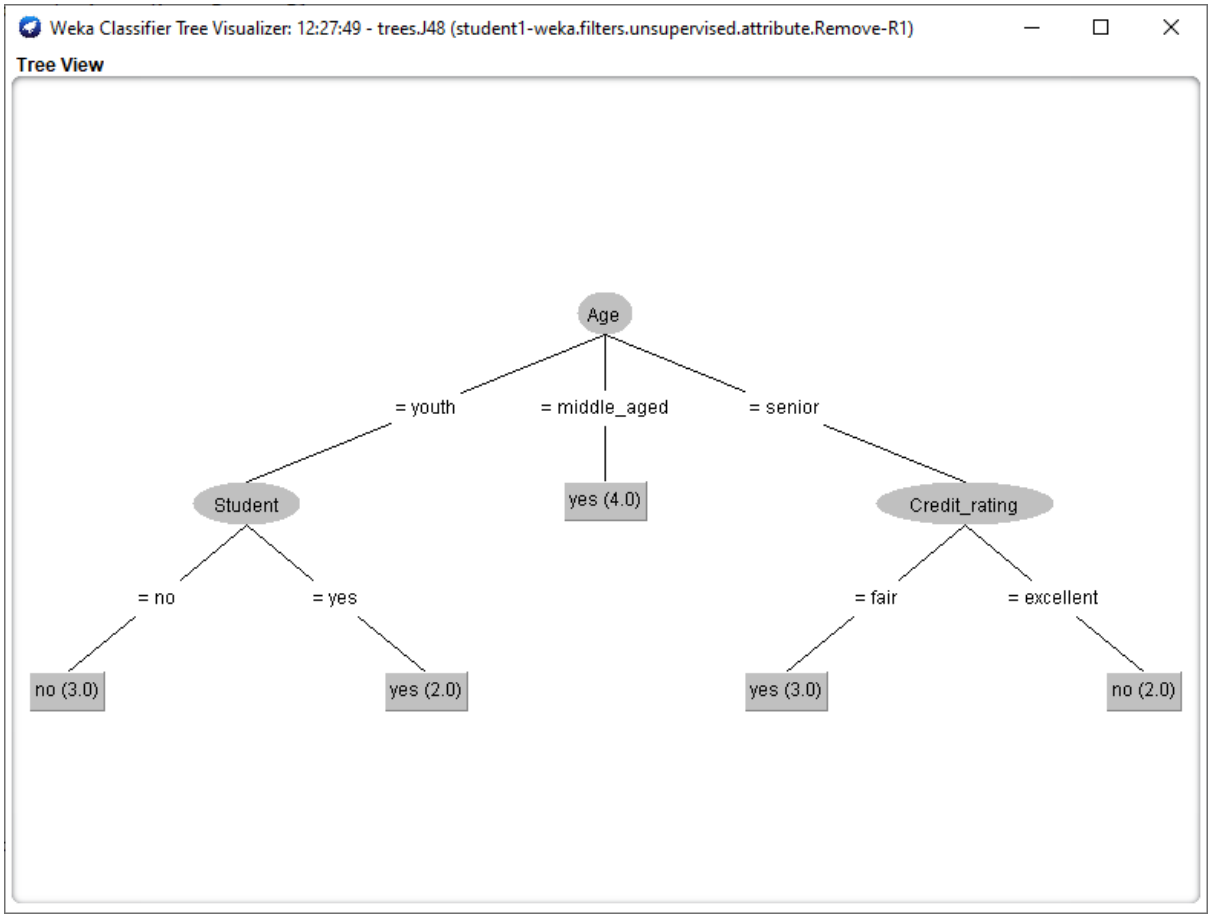
Viewer

Relation: student1-weka.filters.unsupervised.attribute.Remove-R1

No.	1: Age Nominal	2: Income Nominal	3: Student Nominal	4: Credit_rating Nominal	5: **Class: Buys_computer** Nominal
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Add instance | Undo | OK | Cancel

Sample Decision Tree is shown below figure, for student dataset, with 2-3 levels.



Now converting above Decision tree into a set of rules is as follows:

Rule1: If age = youth AND student=yes THEN buys_computer=yes
Rule2: If age = youth AND student=no THEN buys_computer=no
Rule3: If age = middle_aged THEN buys_computer=yes
Rule4: If age = senior AND credit_rating=excellent THEN buys_computer=yes
Rule5: If age = senior AND credit_rating=fair THEN buys_computer=no

Experiment 12:

Generate Association rules for the following transactional database using Apriori algorithm.

TID	List of Items
T100	I1,I2,I5
T200	I2,I4
T300	I2,I3
T400	I1,I2,I4
T500	I1,I3
T600	I2,I3
T700	I1,I3
T800	I1,I2,I3,I5

@relation Trans

@attribute Transaction_id {T100,T200,T300,T400,T500,T600,T700,T800}

@attribute 'Item List' {'I1, I2, I5','I2,I4','I2, I3','I1, I2, I4','I1, I3','I1, I2, I3, I5'}

@data

T100,'I1, I2, I5'

T200,'I2,I4'

T300,'I2, I3'

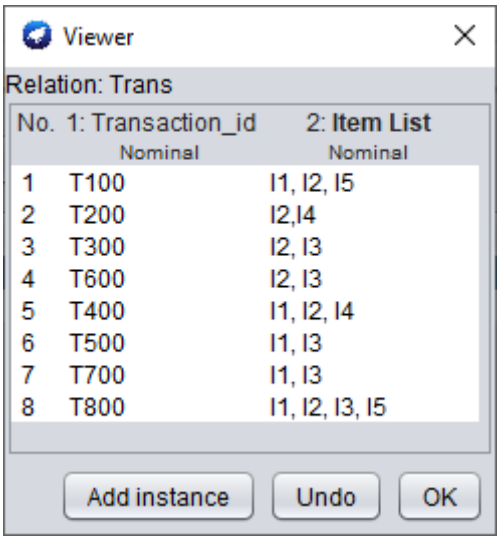
T400,'I1, I2, I4'

T500,'I1, I3'

T600,'I2, I3'

T700,'I1, I3'

T800,'I1, I2, I3, I5'



== 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: Trans

Instances: 8
Attributes: 2
Transaction_id
Item List
=== Associator model (full training set) ===

Apriori
=======

Minimum support: 0.17 (1 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 14

Size of set of large itemsets L(2): 8

Best rules found:

Item List=I1, I2, I5 1 ==> Transaction_id=T100 1 <conf:(1)> lift:(8) lev:(0.11) [0] conv:(0.88)

Transaction_id=T100 1 ==> Item List=I1, I2, I5 1 <conf:(1)> lift:(8) lev:(0.11) [0] conv:(0.88)

Item List=I2,I4 1 ==> Transaction_id=T200 1 <conf:(1)> lift:(8) lev:(0.11) [0] conv:(0.88)

Transaction_id=T200 1 ==> Item List=I2,I4 1 <conf:(1)> lift:(8) lev:(0.11) [0] conv:(0.88)

Transaction_id=T300 1 ==> Item List=I2, I3 1 <conf:(1)> lift:(4) lev:(0.09) [0] conv:(0.75)

Item List=I1, I2, I4 1 ==> Transaction_id=T400 1 <conf:(1)> lift:(8) lev:(0.11) [0] conv:(0.88)

Transaction_id=T400 1 ==> Item List=I1, I2, I4 1 <conf:(1)> lift:(8) lev:(0.11) [0] conv:(0.88)

Transaction_id=T500 1 ==> Item List=I1, I3 1 <conf:(1)> lift:(4) lev:(0.09) [0] conv:(0.75)

Transaction_id=T600 1 ==> Item List=I2, I3 1 <conf:(1)> lift:(4) lev:(0.09) [0] conv:(0.75)

Transaction_id=T700 1 ==> Item List=I1, I3 1 <conf:(1)> lift:(4) lev:(0.09) [0] conv:(0.75)