Practical 8

**Further Practice on Classification Methods in WEKA**

# What are we doing?

In previous practicals (Practical 04 and 05), you were demonstrated how to use WEKA for utilising basic classification methods (including Decision Tree methods and others). This practical extends your practice on classification methods in WEKA.

The first two sections of this practical document briefly summarises algorithms implemented in WEKA categorised by a number of groups of classification methods, and also introduces the Experimenter module provided WEKA which facilitates for comparing the performance of multiple classification solutions.

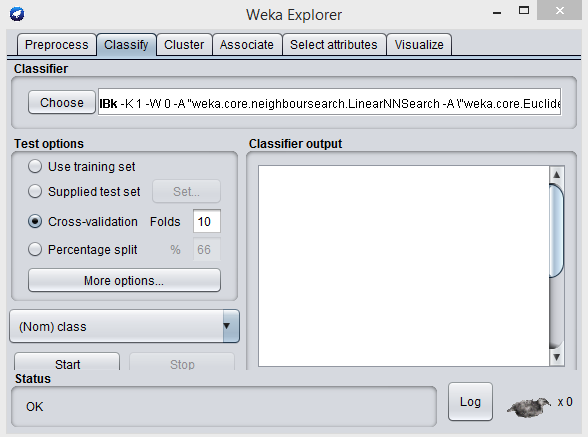
After going through the section, you will be given some laboratory tasks for your further practice on WEKA.

**Submission:**

You are required to submit one document containing screen-captured images for each task (as instructed) via the weekly-practical submission box (available on CP1407 LearnJCU).

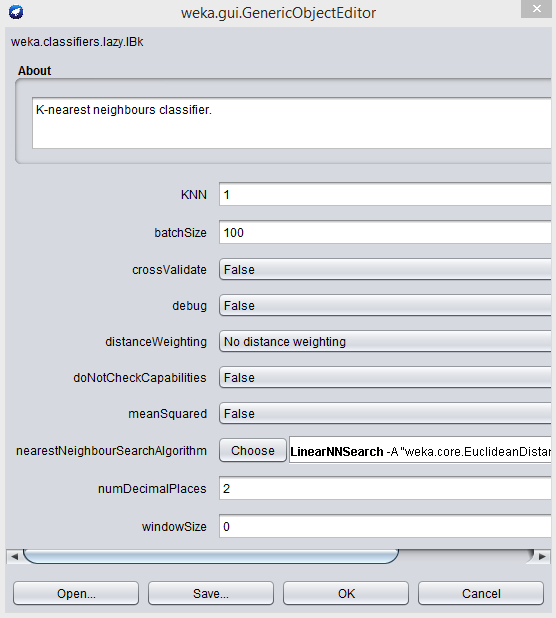
# Classification Methods in WEKA

WEKA is well equipped with various classification algorithms. Pressing the ‘Choose’ button on the Classify tab reveals a list of folders. The nearest neighbour, rule-based, Bayesian and neural network methods are listed, respectively, in the lazy, rules, bayes and functions folders.

Some of those methods are here demonstrated with the data sets in iris.arff and weather.nominal.arff files.

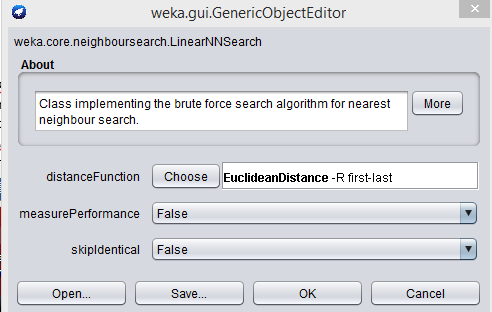
1. **Lazy methods**

In the Lazy category folder, Weka provides several lazy learning algorithms. One of them is IBk which is implemented for k-nearest neighbour approach. The IBk method has a number of input parameters. When you select IBK as a classifier, the default settings of its parameters is listed next to the ‘Choose’ button. You can open the parameter setting window for this method by clicking over the command section.



The parameter kNN allows the user to specify the value of k. When the parameter crossValidate is set to true, the value of k between 1 and the value set for the kNN parameter is determined automatically with leave-one-out cross-validation. When the k nearest neighbours vote for the class of the unseen record, the distances between the examples and the record may also be considered. Therefore, in the distanceWeighting parameter, the user can select ‘No distance weighting’, ‘Weighting by 1/distance’ (division by the distance) or ‘Weighting by 1-distance’ (subtracting the distance).

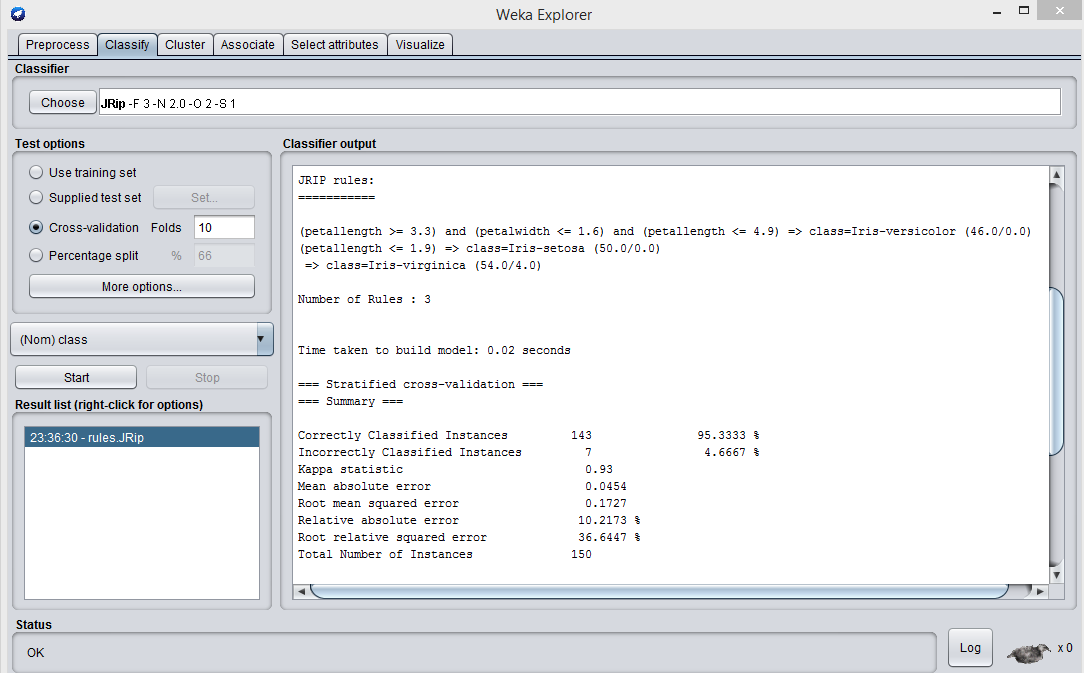
The IBk method also allows the user to specify the eventual number of training examples remaining in the model’s memory space (windowSize).

The most significant feature of the method is that it enables the user to select a nearest neighbour search algorithm (press the Choose button associated with the nearestNeighbourSearchAlgorithm parameter) and a distance function for measuring similarity (click on the command line for nearestNeighbourSEarchAlgorithm and then on the Choose button associated with the distanceFunction parameter).

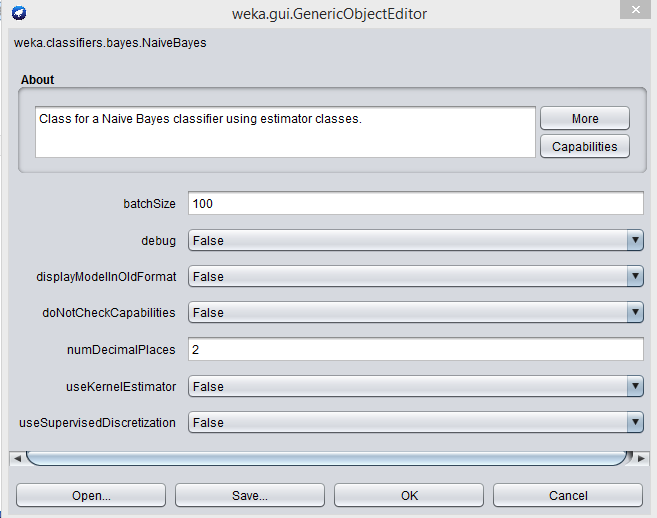
1. **Rule-based Methods**

In the rules folder, Weka provides a number of learning methods of different levels of sophistication.

JRip is a Java implementation of the Ripper algorithm. It is an incremental rule induction algorithm with pruning that works for both categorical and numeric attributes. A test run of the algorithm using the iris data set produces two ordered rules plus a default rule, as shown in the Classifier output window in the figure as below. After each rule, the number of examples of the class and the number of examples of other classes covered by the rule are listed. The classification model has an overall accuracy rate of 95%.



1. **Bayesian Classification Methods**

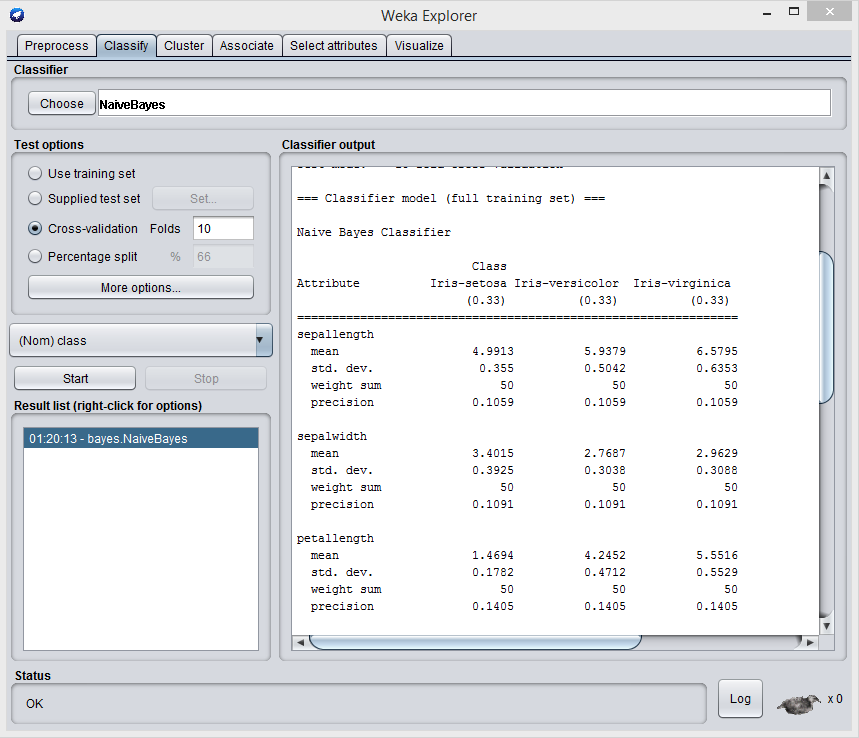
In the Bayes category folder, Weka offers a variety of naïve Bayes solutions. A typical naïve Bayes classifier is NaiveBayes. Once you choose NaiveBayes as a classifier, you can open the parameter selection window for this classifier (by clicking on the command line for NaiveBayes).

Among various parameter options, the parameter useKernelEstimator allows the user to specify how numeric attributes are modelled.

If “False” is selected for this option, simple naïve Bayes algorithm is employed in which numeric attributes are modelled by normal distribution.

If “True” is selected, the NaiveBayes method uses a kernel estimator to estimate the distribution of the numeric attributes rather than assuming a normal distribution. The estimator locates and utilizes the normal kernels to represent the data distribution more accurately.

The figure below shows part of the result of a test run using the NaiveBayes method (with the kernel estimator turned off (selecting “False” option)) over the iris.arff data set. It shows the means and standard deviations of the descriptive attributes for each class, based on which the likelihood of each class is calculated. The overall accuracy rate is above 95%, not bad for such a naïve system.



Try another test run with the kernel estimator turned on (selecting “True”). You will find that the overall accuracy rate is increased to nearly 97%, higher than the result produced by the simple Naïve Bayes method.

**\*\*\*\*\*\*\*\*\*\*\*\*\* Task for submission \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

At this stage, screen capture both classification results (one with kernel estimator turned on and the other with the option turned off) and include the captured image(s) in the document to submit

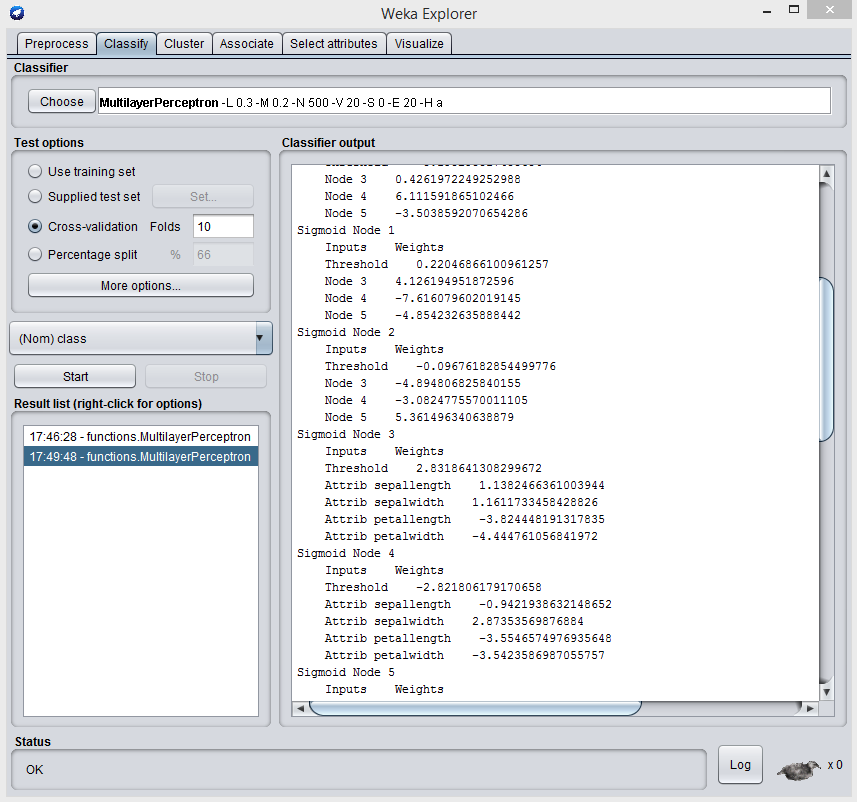
**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

1. **Artificial Neural Network Methods**

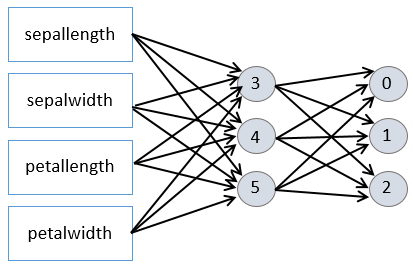
Weka offers an artificial neural network solution in the functions folder. Known as MutiLayerPerceptron, the method takes a set of input parameters and produces a trained network of neuron node solutions using back-propagation.

The parameter include settings for the number of hidden layers and the nodes on each hidden layer, transformation of nominal attributes to binary, normalisation of attribute values, the training time in terms of the number of epochs (the number of times the entire training set is swept through), the learning rate and any possible learning decay, etc.

The topology of the network, the weights attached to links, and the performance measures are described in a textual format and displayed in the Classifier output window as shown in the figure below. The input nodes and the weights attached to the link from the input nodes are listed for each node.



The figure below is the graphical representation of the topology of the network.

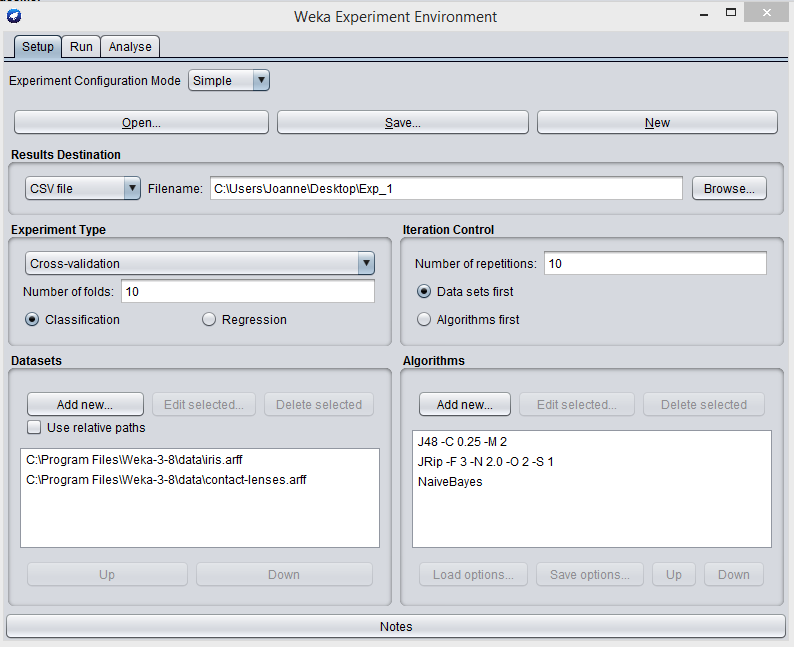


Weka offers a graphical user interface for the neural network solution, which the user can access by setting the GUI parameter to True. Through the interface, the user can add and delete neuron nodes and so modify the topology of the network. The user can also set some input parameters and start the training process.

# Comparing the power of classification techniques

Weka provides a system module called **Experimenter** for comparing the performance of classification solutions. The subsystem allows the user to set up experiments with different techniques and different parameter settings over a number of data sets. The performance data are then collected and comparisons are made of the statistical significance in differences of the accuracy rates of the classification models produces by the techniques.

The experiments are performed in a batch-processing fashion: the experiment details are set up, the computational task is configured and the machine (or machines) is left to complete the experiment. The interface of the subsystem has three tabs: Setup, Run and Analysis.



In the Setup tab, the user creates a new experiment by pressing the New button. The Open and Save button allow the user to save an experiment and then open and edit it later. This also means that an experiment can be repeated if necessary.

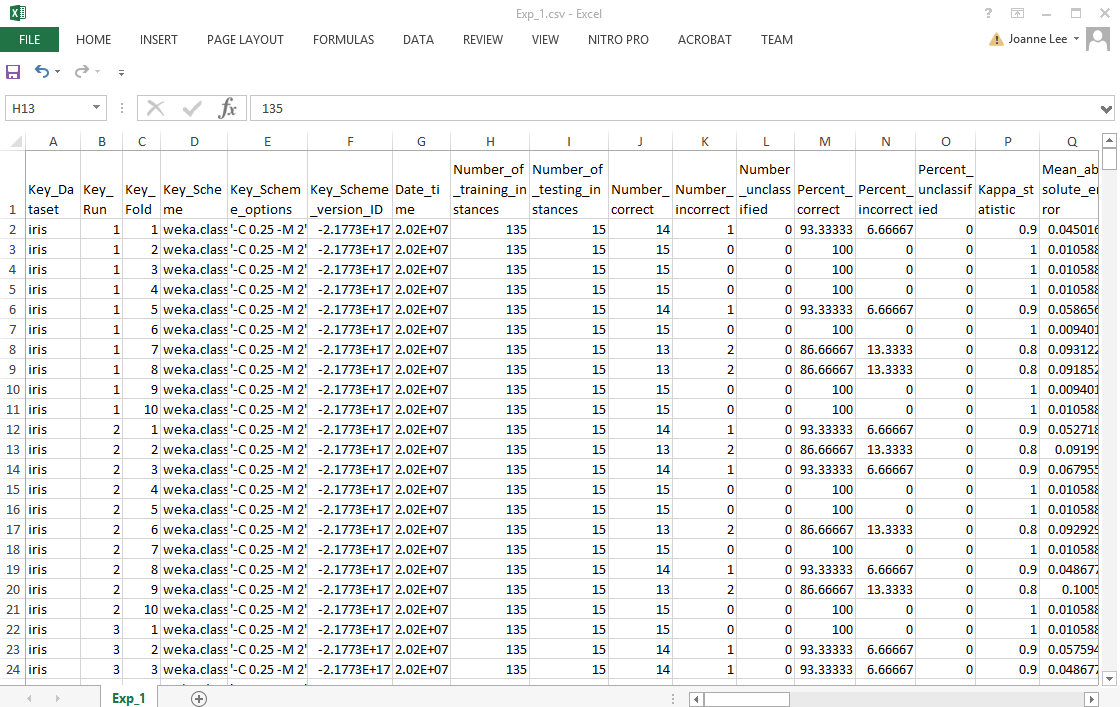
In the Experiment Type block, the user can specify the nature of the experiment (classification or regression) and choose a test option (e.g. cross-validation). In the Iteration Control Block, the user can also specify the number of times that one algorithm is repeated on one data set, and whether the experiment results are grouped by algorithm or by data set.

In the Datasets area, the user can add and remove data sets. The Edit selected button allows the user to open the data viewer and browse the selected data set.

In the Algorithms area, the user can add algorithms, edit the parameters of a selected algorithms, or remove an algorithm.

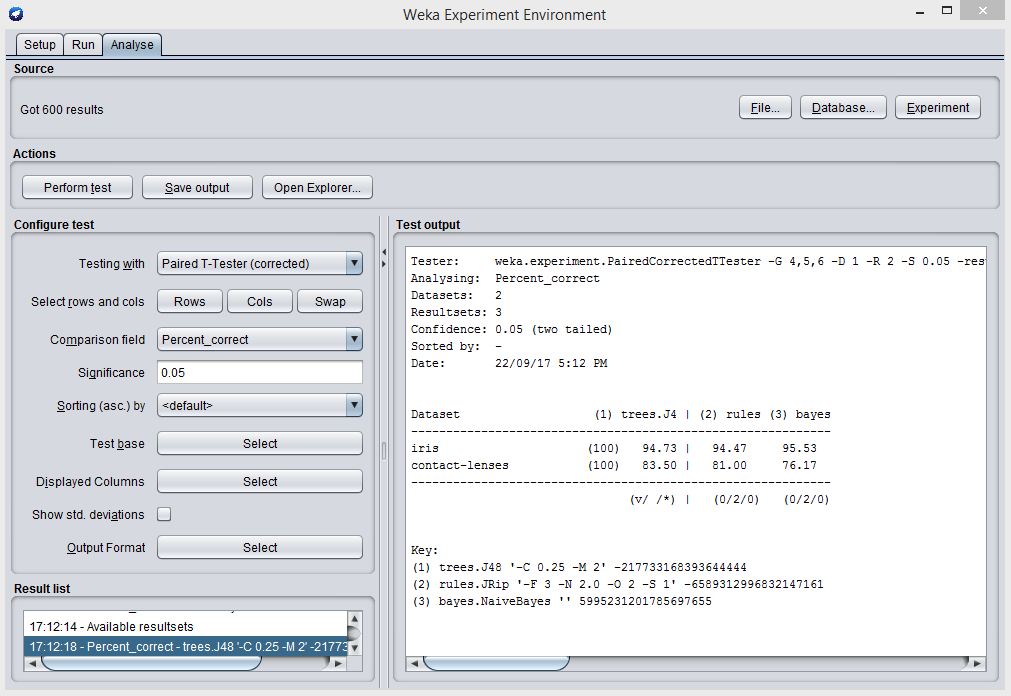
As an example, the above figure shows the setting up of a trial experiment with two data sets and three algorithms with their default parameter settings. The experiment type is set as 10-fold cross-validation and each classification is repeated 10 times.

In the Results Destination block, the user specifies the location and name of a file to store the results of the experiment. The file stores detailed result data, including the experiment configuration data (data set name, run number, the fold number in cross-validation, the classification algorithm and the algorithm parameter settings) and the performance data (the numbers of correctly and incorrectly classified examples, the percentages of correctly and incorrectly classified examples, etc.). The figure below presents part of the experiment result file Exp\_1.csv. From this result file, significance analysis is conducted and reported.



The Run tab provides the control buttons for starting (Start) and stopping (Stop) an experiment. The Log panel displays log messages about the running status of the experiment, indicating if the experiment completes successfully or is aborted. The Status panel displays run-time text messages about the progress of the experiment.

The Analyse tab consists of two main panels: Configure test allows the user to configure the analysis settings and Test output displays the results. By default, the results of the current experiment are analysed when the user presses the Experiment button. The analysis result is displayed in the Test output panel when the user presses the Perform test button.



The figure above shows the analysis result for the experiment run as an example in this document so far, by taking all default settings.

Pair-wise t-tests are conducted between the baseline algorithm (J48) and either JRip or NaiveBayes with the percentage of correct classification as the comparison field when significance is set to 0.05. The result generally states that the average accuracy rates by the classification techniques for each of the two data sets are similar and the differences in the accuracy rates are statistically insignificant.

In this figure, in the analysis result table, ‘(100)’ represents the number of tests for each algorithm on each data set. An algorithm is repeated 10 times; for each run, 10 randomly divided folds are created and used to test accuracy, resulting in 100 tests. The percentage of correct classifications listed is in fact the average. The user can choose to list the standard deviation with the average by selecting the “Show std. deviations’ checkbox.

At the bottom of analysis table are listed triple counters in (x/y/z) format, showing the number of times the algorithm is statistically better than, the same as and worse than the baseline algorithm. For instance, the triple (0/2/0) below the ‘rules’ column indicates the JRip produces the same levels of accuracy as J48 for the two given data sets. If the triple were (0/1/1), it would mean the JRip produces the same level of accuracy as J48 for one data set, but a statistically worse level of accuracy than J48 for another data set.

The symbol ‘\*’ is placed after the worse accuracy measure, and the symbol ‘v’ is used after the better. The triple (v/ /\*) below the baseline algorithm serves as a reminder of the two symbols.

Weka provides other configuration facilities for tests. For instance, the user can choose a different ‘Comparison field’, for example ‘Percent\_incorrect’ to compare error rates instead of accuracy rates. The user may also specify another algorithm as the baseline algorithm by pressing the Select button for the ‘Test base’ parameter.

The Experimenter can be very useful in practice. It allows the user to compare a number of classification techniques and draw conclusions on which classification technique delivers a better model on the basis of statistical significance.

# Laboratory Tasks

**[Task 1]**

In Weka, load the data set ‘soybean.arff’. Perform classifications using the following methods;

1. NaiveBayes (simple Naïve Bayes algorithm)
2. IBk (with different k values)
3. JRip;
4. MultilayerPerceptron

**\*\*\*\*\*\*\*\*\*\*\*\*\* Task for submission \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Screen capture each classification result in Weka and include in your document to submit.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**[Task 2]**

In Weka, perform an experiment that compares the performance (in terms of accuracy level) of the classification algorithms mentioned in [Task 1] against that of J48 over the data set in soybean.arff. Observe and explain the analysis results. The experiment may take some time because the involvement of the MultilayerPerceptron algorithm.

**\*\*\*\*\*\*\*\*\*\*\*\*\* Task for submission \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Screen capture the Weka experiment analysis result to include in the document to submit, in addition to your own explanation about the analysis results.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***