**Weka Exercise 1**

**Getting Acquainted With Weka**

Most CE802 students use the Weka package as the basis of their assignments. This is because it not only provides implementations of a wide range of learning procedures but also includes the machinery for running systematic experiments and reporting relevant statistics for the results. In other words, it will do a lot of the work for you.

These exercises serve two purposes:

* They enable you to discover what facilities Weka provides and how to use them.
* They allow you to see some of the learning procedures that we discuss in the lectures in action.

**Obtaining Weka**

Implementations of Weka for a wide variety of machines/operating systems can be downloaded from the Weka website ( <http://www.cs.waikato.ac.nz/ml/weka/index.html> ). Various versions of Weka are on offer; you almost certainly want the stable version which is currently Weka 3.6. Since Weka is written in Java it requires the Java virtual machine. Choose the appropriate download option if you do not already have this on your computer. The code comes as a self-extracting executable file (weka-3-6-8.exe) so installation is very simple indeed.

**Running Weka**

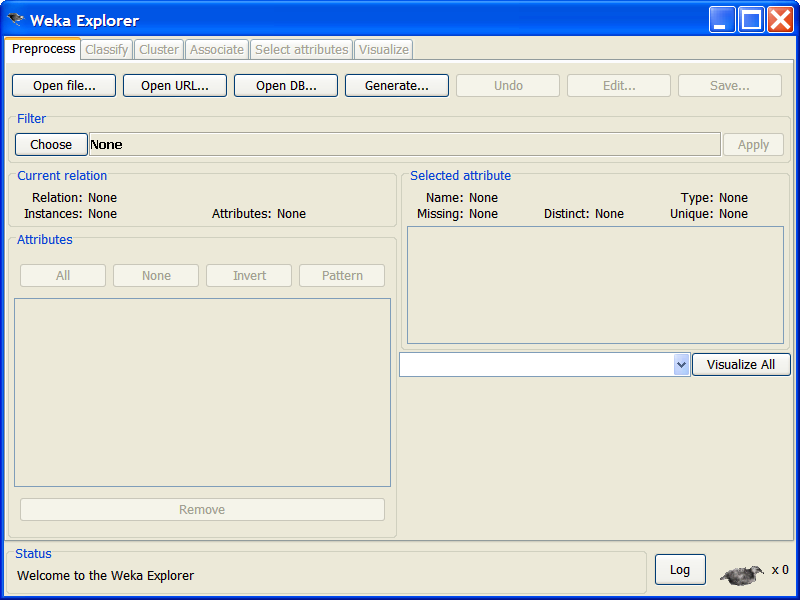
Assuming you do not override the defaults during installation, Weka will be located in a folder called Weka-3.6 in the Program Files folder. The main program can be launched via a short cut or by clicking on a file called either weka.exe or weka.jar (there are minor differences between different versions). Once launched, a small window will appear, usually in the top right of your screen, through which you chose the interface you want to use.



The Explorer is the most useful for most CE802 assignments. Clicking on the button will launch the Explorer interface.

**The Explorer Interface**

This is probably the most confusing part of becoming familiar with Weka because you are presented with quite a complex screen.



Initially “preprocess” will have been selected. This is the tab you select when you want to tell Weka where to find the data set that you want to use.

Weka processes data sets that are in its own ARFF format. Conveniently, the download will have set up a folder within the Weka-3.6 folder called “data”. This contains a selection of data files in ARFF format.

**ARFF format files**

You do not need to know about ARFF format unless you wish to convert data from other formats. However, it is useful to see the information that such files provide to Weka.

The following is an example of an ARFF file for a dataset similar to the one used in the decision tree lecture:

@relation weather.symbolic

@attribute outlook {sunny, overcast, rainy}

@attribute temperature {hot, mild, cool}

@attribute humidity {high, normal}

@attribute windy {TRUE, FALSE}

@attribute play {yes, no}

@data

sunny,hot,high,FALSE,no

sunny,hot,high,TRUE,no

overcast,hot,high,FALSE,yes

rainy,mild,high,FALSE,yes

rainy,cool,normal,FALSE,yes

rainy,cool,normal,TRUE,no

overcast,cool,normal,TRUE,yes

sunny,mild,high,FALSE,no

sunny,cool,normal,FALSE,yes

rainy,mild,normal,FALSE,yes

sunny,mild,normal,TRUE,yes

overcast,mild,high,TRUE,yes

overcast,hot,normal,FALSE,yes

rainy,mild,high,TRUE,no

It consists of three parts. The @relation line gives the dataset a name for use within Weka. The @attribute lines declare the attributes of the examples in the data set (Note that this will include the classification attribute). Each line specifies an attribute’s name and the values it may take. In this example the attributes have nominal values so these are listed explicitly. In other cases attributes might take numbers as values and in such cases this would be indicated as in the following example:

@attribute temperature numeric

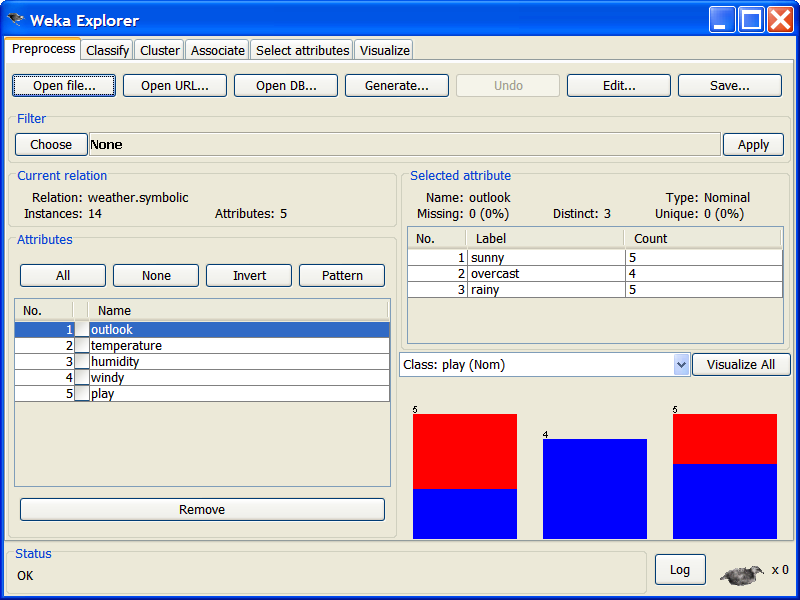
The remainder of the file lists the actual examples, in comma separated format; the attribute values appear in the order in which they are declared above.

**Opening a data set.**

In the Explorer window, click on “Open file” and then use the browser to navigate to the ‘data’ folder within the Weka-3.6 folder. Select the file called weather.nominal.arff. (This is in fact the file listed above).

This is a ‘toy’ data set, like the ones used in class for demonstration purposes. In this case, the normal usage is to learn to predict the ‘play’ attribute from four others providing information about the weather.

The Explorer window should now look like this:



Most of the information it displays is self-explanatory: it is a data set containing 14 examples (instances) each of which has 5 attributes. The ‘play’ attribute has been suggested as the class attribute (i.e. the one that will be predicted from the others).

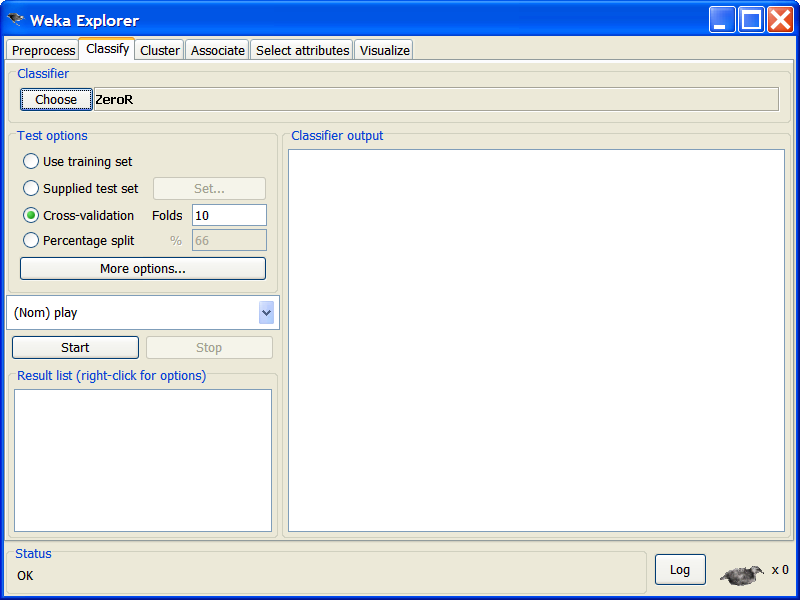
Most of the right hand of the window gives you information about the attributes. Initially, it will give you information about the first attribute (‘outlook’). This shows that it has 3 possible values tells you how many there are of each value. The bar chart in the lower right shows how the values of the suggested class variable are distributed across the possible values of the ‘outlook’.

If you click on ‘temperature’ in the panel on the left, the information about the ‘outlook’ attribute will be replaced by the corresponding information about the temperature attribute.

**Choosing a classifier**

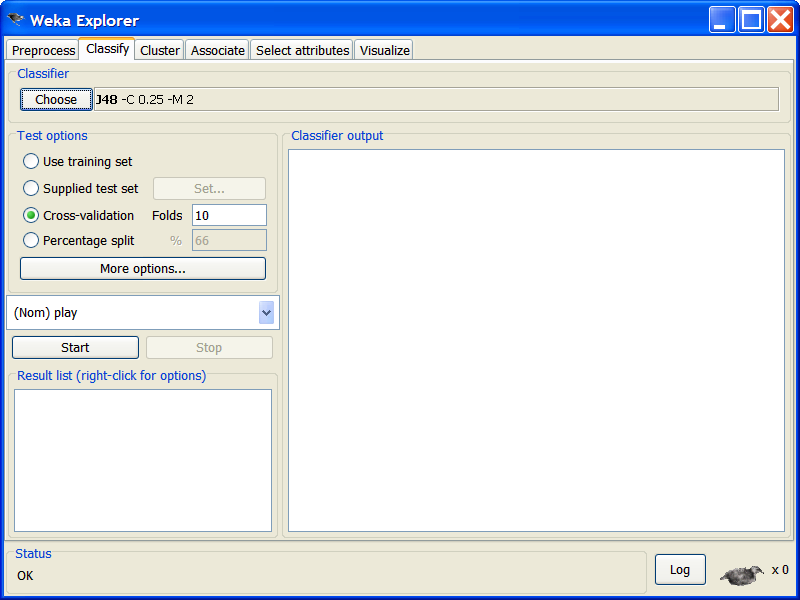
Next we must select a machine learning procedure to apply to this data. The task is classification so click on the ‘classify’ tab near the top of the Explorer window.

The window should now look like this:



By default, a classifier called ZeroR has been selected. We want a different classifier so click on the Choose button. A hierarchical pop up menu appears. Click to expand ‘Trees’, which appears at the end of this menu, then select J48 which is the decision tree program we want.

The Explorer window now looks like this indicating that J48 has been chosen.



The other information alongside J48 indicates the parameters that have been chosen for the program. For this exercise we will ignore these.

**Choosing the experimental procedures**

The panel headed ‘Test options’ allows the user to choose the experimental procedure. We shall have more to say about this later in the course. For the present exercise click on ‘Use training set’. (This will simply build a tree using all the examples in the data set).

The small panel half way down the left hand side indicates which attribute will be used as the classification attribute. It will currently be set to ‘play’. (Note that this is what actually determines the classification attribute – the ‘class’ attribute on the pre-process screen is simply to allow you to see how a variable appears to depend on the values of other attributes).

**Running the decision tree program**

Now, simply click the start button and the program will run. The results will appear in the scrollable panel on the right of the Explorer window. Normally these will be of great interest but for present purposes all we need to notice is that the resulting tree classified all 14 training examples correctly. The tree constructed is presented in indented format, a common method for large trees:

J48 pruned tree

------------------

outlook = sunny

| humidity = high: no (3.0)

| humidity = normal: yes (2.0)

outlook = overcast: yes (4.0)

outlook = rainy

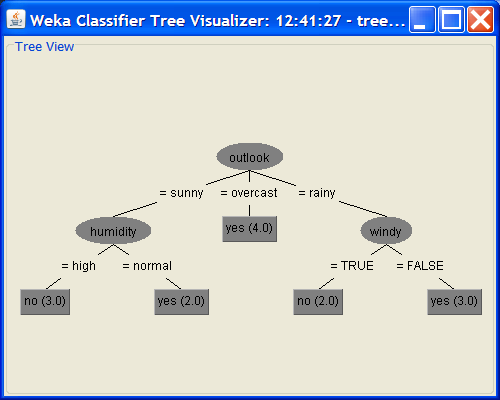
| windy = TRUE: no (2.0)

| windy = FALSE: yes (3.0)

Number of Leaves : 5

Size of the tree : 8

The panel on the lower left headed ‘Result list (right-click for options)’ provides access to more information about the results. Right clicking will produce a menu from which ‘Visualize Tree’ can be selected. This will display the decision tree in a more attractive format:



Note that this form of display is really only suitable for small trees. Comparing the two forms should make it clear how the indented format works.