

Lab 1

Introduction to the Weka Explorer

Overview: Our primary tool for hands-on experience with data mining will be the Weka Data Mining Workbench. This powerful software environment will enable us to conveniently experiment with a variety of data mining strategies, algorithms, evaluation and visualization techniques. Though we have not yet begun to explore the details of these algorithms yet, today's activity will allow us to get our feet wet with the software we'll be using.

1. Overview

The Weka Data Mining Workbench was developed at the University of Waikato in New Zealand by the authors of our Data Mining text. It collects in one place a variety of powerful tools for pre-processing data sets and running all of the cutting edge machine learning algorithms on them. It also includes tools for visualizing both the data and the models that the mining algorithms produce. And it makes powerful statistical methods of evaluating the models easily available.

Throughout the course of the semester, we will use Weka to implement and experiment with the key objectives of the class. Specifically, as we study a new learning method, we'll use Weka to apply that learning method to a dataset and analyze the output to gain a greater understanding of the patterns inherent in the data. We will also use the models that Weka learns to predict behavior based on new data. Once multiple algorithms have been learned in class, we will use Weka to apply different algorithms to the same dataset to determine the best one for understanding and prediction.

Since we have not as yet learned any algorithms, today's lab is a straightforward exploration of Weka's main GUI, the *Explorer*. We will use a tutorial provided by the authors themselves. During the activity you will encounter concepts that you haven't read about yet. Have no fear – we will cover them soon enough, and in the meantime ask a lot of questions. This experience will prepare us well for the conceptual material soon to follow.

2. The Weka Explorer Interface

The first thing you will need is your own copy of the datasets provided by Weka. These are located in the folder entitled `Weka datasets` in the CSC-272 folder on Box. Copy the entire folder to your USB drive or to a shared network drive. You will need to be able to access these datasets from any computer that you're working on throughout the semester.

Next, turn to the next page and get started. The tutorial exercises are clearly labeled in bold. Most of them require some sort of answer or observation based on your experiment. Enter your answers in the space on the lab report located on the last page of this handout, and hand in the report when you are done with lab.

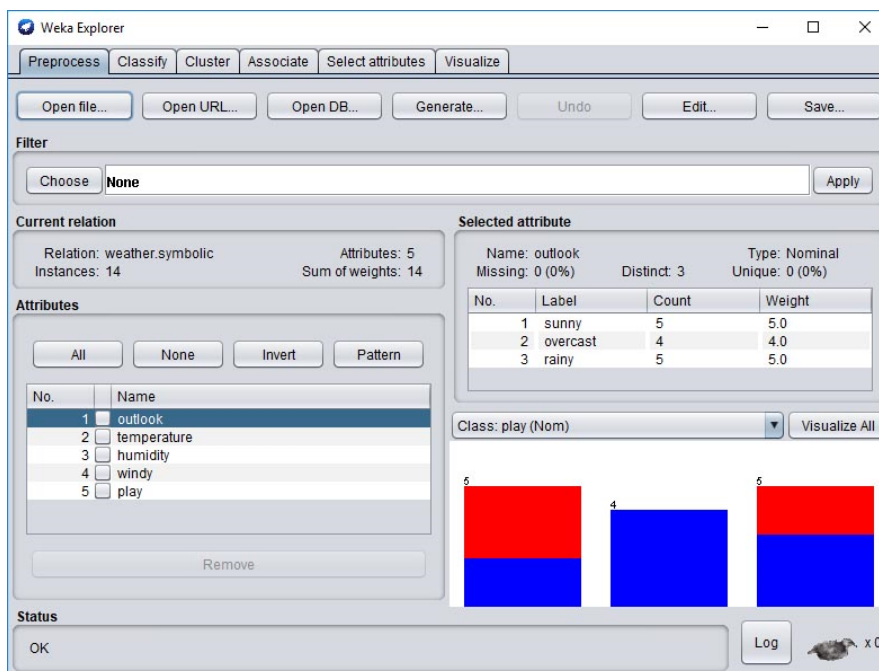
3. Introduction to the Explorer Interface

Click the *Start* menu on the R203 computers and search for *Weka 3.8*. (On the R201 iMacs, click the magnifying glass icon and search for *weka-3-8-1-oracle-jvm.app*.) This starts up the Weka GUI Chooser (shown below). Click the *Explorer* button to enter the Weka Explorer. The Preprocess panel opens up when the Explorer interface is started.



Loading a Dataset

Load a dataset by clicking the *Open file* button in the top left corner of the panel. Inside your Weka data folder, which you downloaded from Box, you will find a file named *weather.nominal.arff*. This contains the nominal version of the standard "weather" dataset in Table 1.2 of the *Data Mining* textbook. Open this file. The screen will look like this:



As the result shows, the weather data has 14 instances, and 5 attributes called *outlook*, *temperature*, *humidity*, *windy*, and *play*. Click on the name of an attribute in the left subpanel to see information about the selected attribute on the right, such as its values and how many times an instance in the dataset has a particular value. This information is

also shown in the form of a histogram. All attributes in this dataset are *nominal* - that is, they have a predefined finite set of values. The last attribute, *play*, is the *class attribute*; its value can be *yes* or *no*.

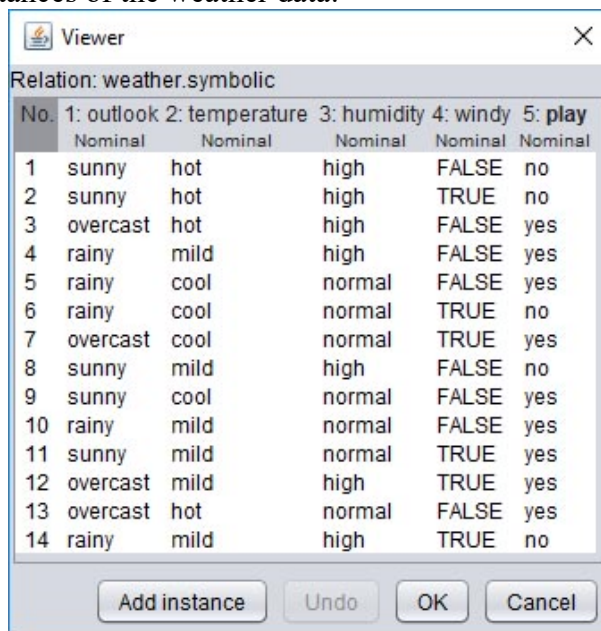
Familiarize yourself with the Preprocess panel by doing the following exercises. Record your results on the lab report on the final page of this handout.

Exercise 1 What are the values that the attribute *temperature* can have?

Exercise 2 Load a new dataset. Click the *Open file* button and select the file *iris.arff*, which corresponds to the iris dataset in Table 1.4 of the textbook. How many instances does this dataset have? How many attributes? What is the range of possible values of the attribute *petallength*?

The Dataset Editor

It is possible to view and edit an entire dataset from within Weka. To do this, load the *weather.nominal.arff* file again. Click the *Edit* button from the row of buttons at the top of the Preprocess panel. This opens a new window called Viewer, which lists all instances of the weather data:



No.	1: outlook	2: temperature	3: humidity	4: windy	5: play
	Nominal	Nominal	Nominal	Nominal	Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Exercise 3 What is the function of the first column (**No.**) in the Viewer window?

Exercise 4 What is the value of the class attribute of instance number 8 in the weather data? (Re-read above for a reminder of what the class attribute is.)

Exercise 5 Load the iris data and open it in the editor. How many numeric and how many nominal attributes does this dataset have? (Again, re-read above if necessary.)

Applying a Filter

Weka *filters* can be used to modify datasets in a systematic fashion – that is, they are data preprocessing tools. Reload the *weather.nominal* dataset, and let's remove an attribute from it. The appropriate filter is called *Remove*. Its full name is `weka.filters.unsupervised.attribute.Remove`

Examine this name carefully. Filters are organized into a hierarchical structure of which the root is *weka*. Those in the *unsupervised* category don't require a class attribute to be set; those in the *supervised* category do. Filters are further divided into those that operate primarily on attributes (the *attribute* category) and ones that operate primarily on instances (the *instance* category).

Click the *Choose* button in the Preprocess panel to open a hierarchical menu from which you select a filter by following the path corresponding to its full name. Use the path given in the full name above to select the *Remove* filter. The word "Remove" will appear in the field next to the *Choose* button.

Click on the field containing "Remove". The Generic Object Editor window, which is used throughout Weka to set parameter values for all of the tools, opens. In this case it contains a short explanation of the Remove filter. Click *More* to get a fuller description. Enter 3 into the *attribute-Indices* field and click the *OK* button. The window with the filter options closes. Now click the *Apply* button on the right, which runs the data through the filter. The filter removes the attribute with index 3 from the dataset, and you can see that this has happened. This change does not affect the dataset in the file. It only applies to the data held in memory. The changed dataset can be saved to a new ARFF file by pressing the *Save* button and entering a file name. The action of the filter can be undone by pressing the *Undo* button. Again, this applies to the version of the data held in memory.

What we have described illustrates how filters are applied to data. However, in the particular case of *Remove*, there is a simpler way of achieving the same effect. Instead of invoking a filter, attributes can be selected using the small boxes in the Attributes subpanel and removed using the *Remove* button that appears at the bottom, below the list of attributes.

Exercise 6 Load the *weather.nominal* dataset. Use the filter *weka.unsupervised.instance.RemoveWithValues* to remove all instances in which the humidity attribute has the value *high*. To do this, first make the field next to the *Choose* button show the text *RemoveWithValues*. Then click on it to get the Generic Object Editor window, and figure out how to change the filter settings appropriately. Play with it. It's not immediately obvious. (Hint: If the index of the "humidity" attribute is 3, what is the index of the "high" value?) Click *Edit* to confirm the change.

Exercise 7 Undo the change to the dataset that you just performed, and verify that the data has reverted to its original state.

The Visualize Panel

Now take a look at Weka's data visualization facilities. These work best with numeric data, so we use the iris data. Load *iris.arff*, which contains the iris dataset of Table 1.4 containing 150 examples of three types of Iris: *Iris setosa*, *Iris versicolor*, and *Iris virginica*.

Click the *Visualize* tab to bring up the Visualize panel. Click the first plot in the second row to open up a window showing an enlarged plot using the selected axes. Instances are shown as little crosses, the color of which depends on the instance's class. The *x-axis* shows the *sepalength* attribute, and the *y-axis* shows *petalwidth*.

Clicking on one of the crosses opens up an Instance Info window, which lists the values of all attributes for the selected instance. Close the Instance Info window again.

The selection fields at the top of the window containing the scatter plot determine which attributes are used for the *x-* and *y-*axes. Change the *x-axis* to *petalwidth* and the *y-axis* to *petallength*. The field showing *Color: class (Nom)* can be used to change the color coding.

Each of the barlike plots to the right of the scatter plot window represents a single attribute. In each bar, instances are placed at the appropriate horizontal position and scattered randomly in the vertical direction. Clicking a bar uses that attribute for the *x-axis* of the scatter plot. Right-clicking a bar does the same for the *y-axis*. Use these bars to change the *x-* and *y-*axes back to *sepalength* and *petalwidth*.

The *Jitter* slider displaces the cross for each instance randomly from its true position, and can reveal situations where instances lie on top of one another. Experiment a little by moving the slider.

The *Select Instance* button and the *Reset*, *Clear*, and *Save* buttons let you modify the dataset. Certain instances can be selected and the others removed. Try the *Rectangle* option: Select an area by left-clicking and dragging the mouse. The *Reset* button changes into a *Submit* button. Click it, and all instances outside the rectangle are deleted. You could use *Save* to save the modified dataset to a file. *Reset* restores the original dataset.

The Classify Panel

Now we apply a classifier to the weather data. Load the weather data again. Go to the Preprocess panel, click the *Open file* button, and select *weather.nominal.arjf* from the data directory. Then switch to the Classify panel by clicking the *Classify* tab at the top of the window.

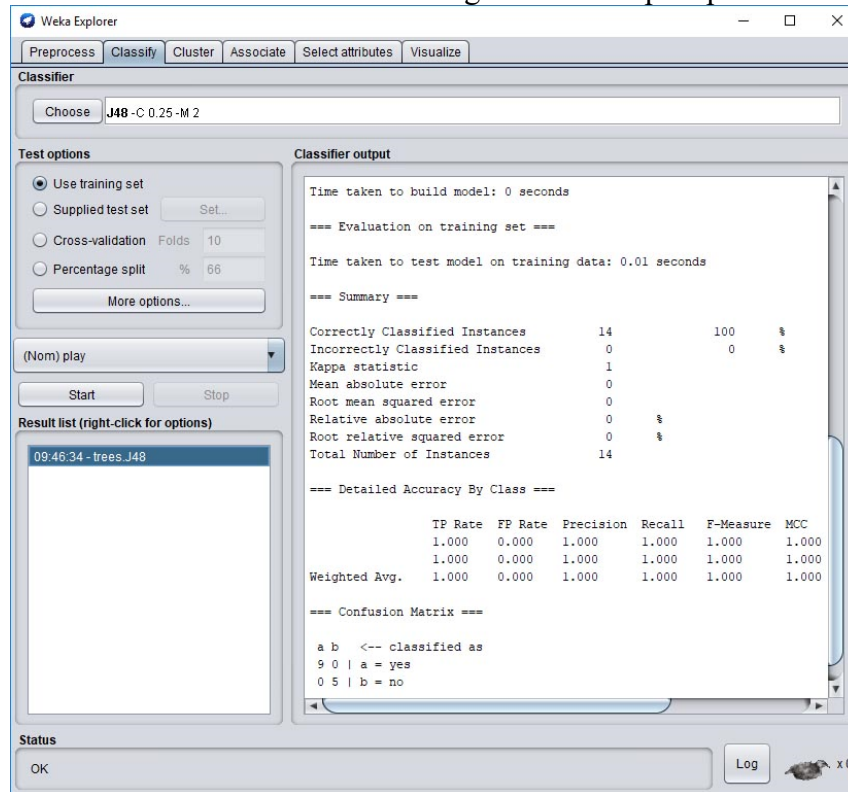
Using the C4.5 Classifier

The C4.5 algorithm for building decision trees is implemented in Weka as a classifier called *J48*. Select it by clicking the *Choose* button near the top of the *Classify* tab. A dialog window appears showing various types of classifier. Click the *trees* entry to reveal its subentries, and click *J48* to choose that classifier. Classifiers, like filters are organized in a hierarchy: *J48* has the full name *weka.classifiers.trees.J48*.

The classifier is shown in the text box next to the *Choose* button: It now reads *J48 -C 0.25 -M 2*. This text gives the default parameter settings for this classifier, which in this case rarely require changing to obtain good performance.

For illustrative purposes we evaluate the performance using the training data, which has been loaded in the Preprocess panel. (As we will see, this is not generally a good idea

because it leads to unrealistically optimistic performance estimates.) Choose *Use training set* from the *Test options* part of the Classify panel. Once the test strategy has been set, the classifier is built and evaluated by clicking the *Start* button. This processes the training set using the currently selected learning algorithm, C4.5 in this case. Then it classifies all the instances in the training data and outputs performance statistics:



Interpreting the Output

The outcome of training and testing appears in the Classifier Output box on the right. Scroll through the text and examine it. First, look at the part that describes the decision tree:

```

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)

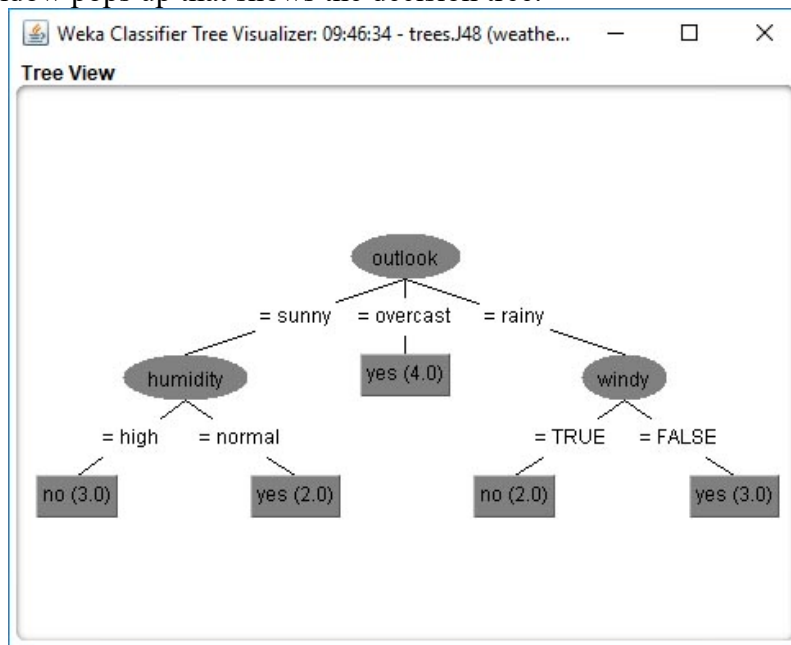
Number of Leaves   :    5

Size of the tree    :    8

```

This represents the decision tree that was built, including the number of instances that fall under each leaf. The textual representation is clumsy to interpret but Weka can generate an equivalent graphical version.

Here's how to get the graphical tree. Each time the *Start* button is pressed and new classifier is built and evaluated, a new entry appears in the Result List panel in the lower left corner of the Explorer window. To see the tree, right-click (option-click on a Mac) on the entry *trees.J48* that has just been added to the result list and choose *Visualize tree*. A window pops up that shows the decision tree:



Right-click (option-click) a blank spot in this window to bring up a new menu enabling you to auto-scale the view. You can pan around by dragging the mouse.

Now look at the rest of the information in the Classifier Output area. The next two parts of the output report on the quality of the classification model based on the chosen test option.

This text states how many and what proportion of the test instances have been correctly classified:

Correctly Classified Instances 14 100%

This is accuracy of the model on the data used for testing. In this case it is completely accurate (100%), which is often the case when the training set is used for testing.

At the bottom of the output is the *confusion matrix*:

```

===Confusion Matrix===
a b    <- classified as
9 0 | a = yes
0 5 | b = no
  
```

Each element in the matrix is a count of instances. Rows represent the true classes, and columns represent the predicted classes. As you can see, all 9 *yes* instances have been predicted as *yes*, and all 5 *no* instances as *no*.

Exercise 8. How would this instance be classified using the decision tree?

```
outlook= sunny, temperature = cool, humidity = high, windy == TRUE
```

Setting the Test Method

When the *Start* button is pressed, the selected learning algorithm is run and the dataset that was loaded in the Preprocess panel is used with the selected test protocol. For example, in the case of *tenfold cross-validation* this involves running the learning algorithm 10 times to build and evaluate 10 classifiers. A model built from the full training set is then displayed in the Classifier Output area: This may involve running the learning algorithm one final time. The remainder of the output depends on the test protocol that was chosen using test options (which we'll discuss later).

Exercise 9 Load the iris data using the Preprocess panel. Evaluate C4.5 on this data using (a) the training set and (b) tenfold cross-validation. What is the estimated percentage of the correct classifications for (a) and (b)? Which estimate is more realistic (more trustworthy, based on what you just read)?

Visualizing Classification Errors

Right-click (option-click) the most recent *trees.J48* entry in the result list and choose *Visualize classifier errors*. A scatter plot window pops up. Instances that have been classified correctly are marked by the little crosses; ones that are incorrect are marked by little squares.

Exercise 10 Use the *Visualize classifier errors* function to find the wrongly classified test instances for the cross-validation performed in Exercise 9. What can you say about the location of the errors? (This is a good place to use the Jitter slider to get a better view of what is going on.) That is, where do the errors lie in relation to the accurate predictions?

Lab 1 Report: Introduction to the Weka Explorer

Name: _____

Exercise 1:

Exercise 2:

Exercise 3:

Exercise 4:

Exercise 5:

Exercise 6: (How did you change the filter settings? What did you have to figure out?)

Exercise 7: (Just put a check mark to indicate that you did this.)

Exercise 8:

Exercise 9: (Explain your answer.)

Exercise 10: (Explain your answer.)