7CCSMDM1 (Data Mining)

Practical 1: Classifying Data using WEKA

1 Installing WEKA

WEKA is an open source, freely downloadable package written in Java which contains many useful Data Mining tools. WEKA can be obtained from here:

`http://www.cs.waikato.ac.nz/ml/weka/`

1. Download the most recent **stable** version of WEKA (currently version 3.8.2). For the workstations in the lab, you will need to select the version that is in the section of the download page marked **Other platforms (Linux, etc...)**. Install this version in your Linux profile, following the instructions on the WEKA download page.

2. On the KEATS page for this module (7CCSMDM1), under **[Week 1]**, find and download the **.zip** archive file containing the data sets which will be used in this practical.

3. Unzip the archive.

4. In the *GUI Chooser*. Here, click on `Explorer`.

5. This will bring up the *Explorer* page, with the *Preprocess* tab highlighted. Here, click on `Open file...` and select `iris.arff`. Use the file manager to locate the data file in the folder where you unzipped the archive (above). The data file will be opened and parsed, as shown in the screenshot in Figure 1. WEKA can load data in a number of formats. Its native format is *ARFF (Attribute-Relation File Format)*, but it can also read CSV files and other formats.

6. In the *attribute pane* in the *Explorer* tab, click on each of the attributes and explore the information displayed.

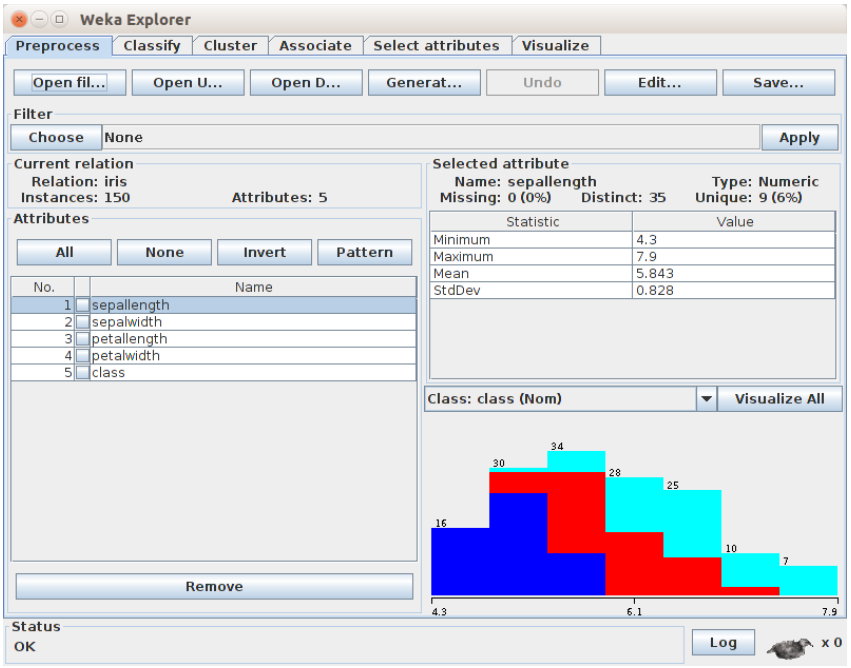


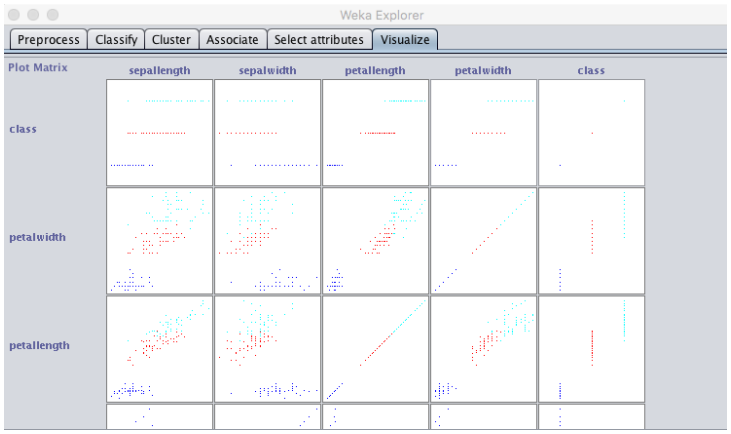
Figure 1: WEKA Explorer, Iris data set

2 Building Decision Trees with WEKA

With this exercise, you will build *Decision Trees* using two different `Classifier` packages in the WEKA Explorer.

FIRST: Start with the iris data set entered, as above. In this iris data the class represents the type of flower (Iris setosa, iris versicolor, iris virginica) to which the sepal and petal width and length belong to.

1. In the *Explorer*, click on the *Visualize* tab. This shows a matrix of scatter plots, illustrating how the different attributes relate to each other (or not).



2. In the *Explorer*, click on the `Classify` tab and then click on the `Choose` button. Scroll down to the set of `trees` classifiers, and select `J48`.

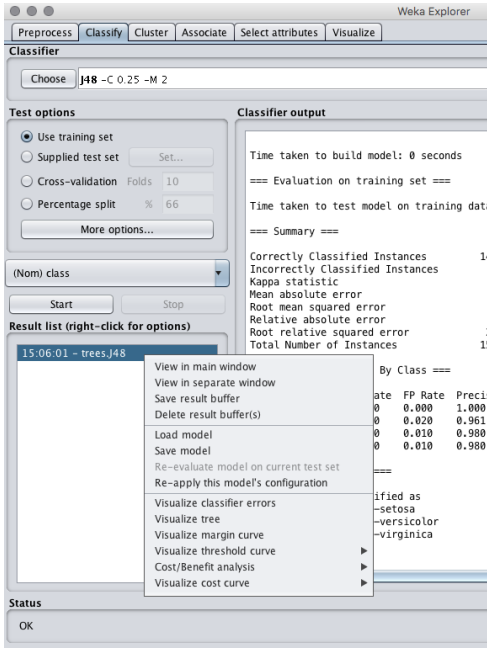
3. For `Test options`, click on `Use training set`. Then click on `Start`.

4. The results will look something like the example shown below. Note that the percentage of **Correctly Classified Instances** is 98%.

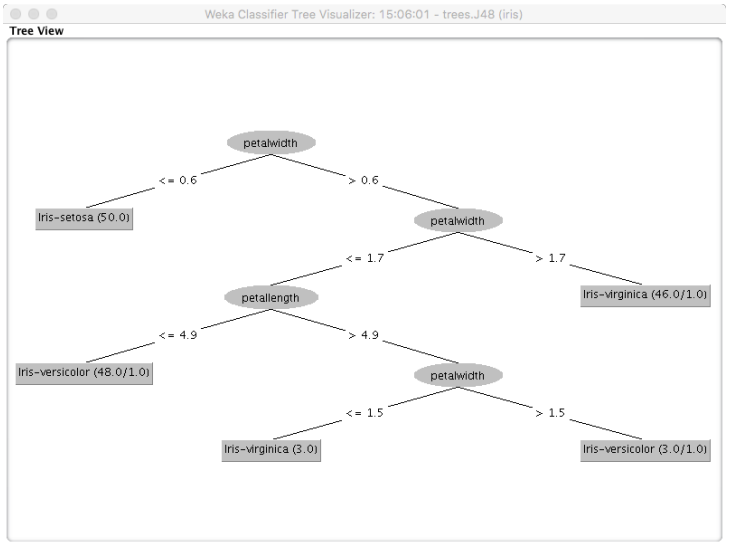
```
=== Run information ===
Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     iris
Instances:    150
Attributes:   5
              sepallength
              sepalwidth
              petallength
              petalwidth
              class
Test mode:    evaluate on training data
=== Classifier model (full training set) ===
J48 pruned tree
------------------
petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth > 0.6
|   petalwidth <= 1.7
|   |   petallength <= 4.9: Iris-versicolor (48.0/1.0)
|   |   petallength > 4.9
|   |   |   petalwidth <= 1.5: Iris-virginica (3.0)
|   |   |   petalwidth > 1.5: Iris-versicolor (3.0/1.0)
|   petalwidth > 1.7: Iris-virginica (46.0/1.0)

Number of Leaves  :  5
Size of the tree :  9
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         147               98      %
Incorrectly Classified Instances         3                2      %
Kappa statistic                          0.97
Mean absolute error                      0.0233
Root mean squared error                  0.108
Relative absolute error                  5.2482 %
Root relative squared error             22.9089 %
Total Number of Instances              150
=== Detailed Accuracy By Class ===
                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area...
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000...
                 0.980    0.020    0.961      0.980    0.970      0.955    0.990...
                 0.960    0.010    0.980      0.960    0.970      0.955    0.990...
Weighted Avg.    0.980    0.010    0.980      0.980    0.980      0.970    0.993...
=== Confusion Matrix ===
  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 49  1 |  b = Iris-versicolor
  0  2 48 |  c = Iris-virginica
```

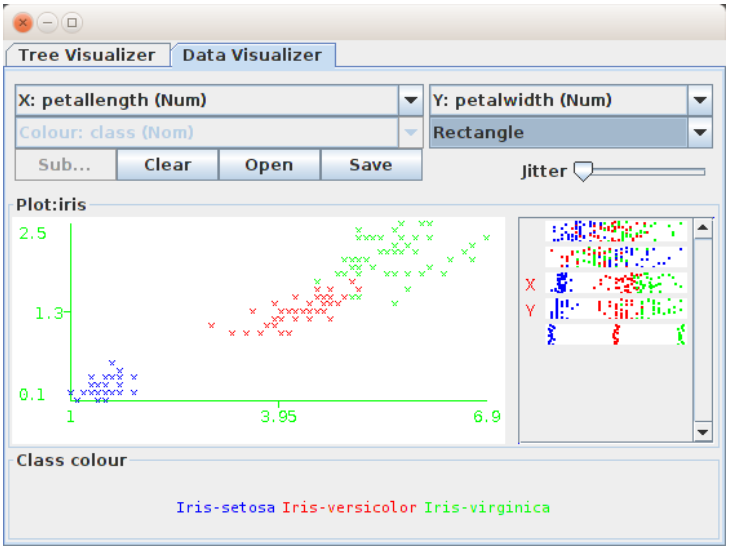
5. In the lower right of the display, there will be a box marked `Result list (right-click for options)`. Select the result set from the list and right-click on it. This will show a menu of options, as below.



6. Select `Visualize tree` from the list of options:

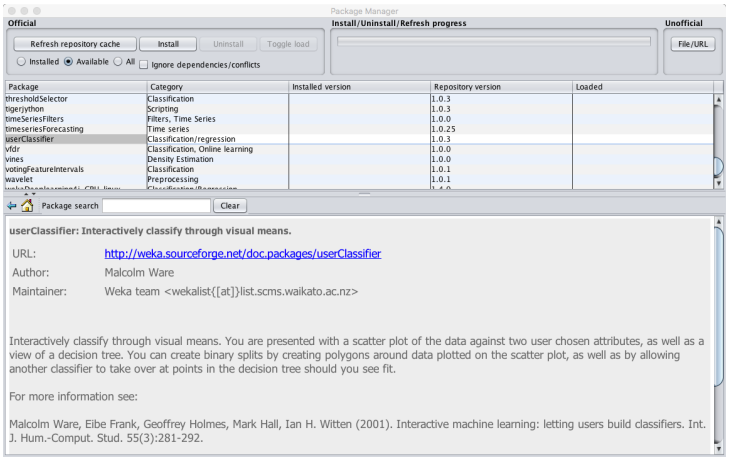


7. Select `Visualize classifier errors` from the list of options: You will see a graph displaying the data. You can use the pulldown menus to select which attribute you want to use for `X` and which attribute for `Y`—as plotted in the data visualizer (2D graph) within that window. The class of each point is illustrated by a different colour, see the key at the bottom of the window. The example below uses *petal length* for the `X` value and *petal width* for the `Y` value:

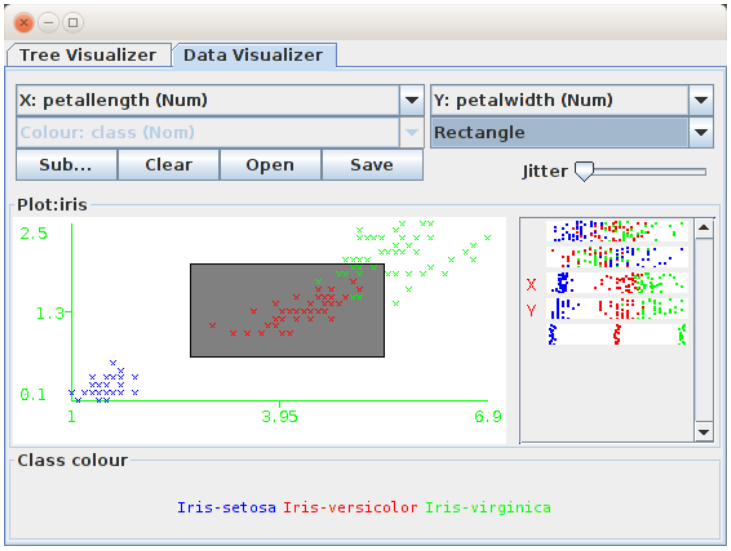


SECOND: Now let's try defining your own classification rules. You need to install the **User classifier** package.

8. Close your WEKA *Explorer* window and return to the *GUI Chooser*
9. Click on `Tools` and select `Package manager`.
10. Find the **userClassifier** package on the list of packages. Select it and click on `Install`. Close the package manager once finished.

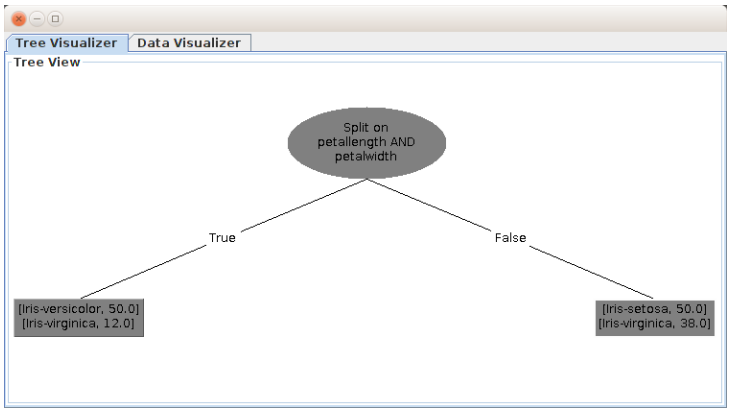


11. Go back to the *Explorer*.
12. Load the `iris` data set again.
13. In the *Explorer*, click on the `Classify` tab and then click on the `Choose` button. Scroll down to the set of `trees` classifiers, and select `User Classifier`.
14. For `Test options`, click on `Use training set`. Then click on `Start`.
    A new window will pop up, which contains the *Tree Visualiser* and the *Data Visualiser*.
15. In the *Data Visualizer*, click on the `Select Instance` pulldown menu and select `Rectangle`. Then in the 2D graph, you can click-and-drag to draw a rectangular region, as below:



Make sure that your region contains at least one point from the data set (i.e., **x**'s displayed in the graph). Then click on `Submit`.
Note that the `Submit` *button is disabled until you have selected a region which contains at least one example.*

16. Then click on the *Tree Visualiser* and you should see that there are nodes which have been added to the tree.
Note that you may have to increase the size of the window in order to see all the nodes in the tree.



17. You can continue adding nodes to the tree by creating more splits in the data set, in this way.

18. When you are done, close the window (click on $\times$ in the upper left corner). This will run your decision tree.

19. In the *Classifier output* section of the *Explorer* window, you'll see the results of your efforts—i.e., the output of evaluating the data set using your decision tree as a classifier.
Is the percentage of **Correctly Classified Instances** higher (better) or lower (worse) than when you used the **J48** classifier earlier?

THIRD:

20. Load the *weather.arff* data into WEKA. Use this tool to create a Decision Tree for the *weather.arff* data. Try different approaches and see if you can improve your evaluation result. Strive to achieve higher accuracy than 66%!

FOURTH:

21. Load the *diabetes.arff* data into WEKA. Use this tool to create a Decision Tree for the *diabetes.arff* data. Try different approaches, and display different `X` and `Y` attributes in order to create your splits.

3 Linear Regression with WEKA

With this exercise, you will apply *Linear Regression* in the WEKA Explorer.

1. Open the WEKA *Explorer*, with the *Preprocess* tab highlighted. Here, click on `Open file...` and select `london-borough-profiles-2016.csv`. Note that you'll have to change **Files of Type:** from the default `Arff data files (*.arff)` to `CSV data file (*.csv)` in order to be able to select this data file.

2. The `london-borough-profiles-2016.csv` is a bigger data set than the Iris data and it is an example of a real-world data set. There are 85 attributes and 40 instances in this data set. Most of the attributes are not **numeric**. Since this exercise is to use **Linear Regression**, we will start by *preprocessing* the data set and removing all the non-numeric attributes.

3. In the *Attributes* panel of the *Preprocess* window, scroll down the list and click on every **Numeric** attribute. You can determine what the data type of an attribute is by looking in the right-hand panel that is labelled **Selected attribute** and finding the **Type:** field in the upper right corner.
   Hint: there are 15 Numeric attributes in the data set.

4. Then click on `Invert` in the *Attributes* panel. This will toggle all the boxes you have ticked, so that all the non-Numeric attributes are now selected.

5. Then click on `Remove` in the *Attributes* panel. This will remove all the non-Numeric attributes. *(Don't worry—this is not changing the data file stored on your computer. This is only changing which attributes are loaded into WEKA's memory.)*

6. You can explore the pairwise relationships amongst the numeric attributes by clicking on the *Visualize* tab in the *Explorer*. If you click on any of the scatter plots shown in the *Visualise* window, that pop up another window with the plot you selected in a larger format.
   You can also use the `X` and `Y` drop-down menus on the pop-up window to change which pair of attributes are plotted.

7. For this exercise, we'll look at using **Linear Regression** to try and **predict** two of the values in the data set: **Happiness score** and **Anxiety score**.

8. Click on the `Classify` tab in the *Explorer*.

9. Then click on `Choose`, and in the drop-down list under `functions`, select `LinearRegression`.

10. In the *Test options* panel, click on `cross-validation`.

11. Then, below the *Test options* panel, you will find another drop-down list which contains all the attributes. Select `Anxiety score` (since it is the last one on the list, it is probably already selected).

12. Then click on `Start`.

13. The results should look something like this:

```
Linear Regression Model
Anxiety score 2011-14 (out of 10) =
     -0.1187 * Proportion of population aged 0-15, 2016 +
     -0.0234 * Proportion of population of working-age, 2016 +
     -0.0836 * Proportion of population aged 65 and over, 2016 +
     -0.0047 * New migrant (NINo) rates, (2014/15) +
     -0.0154 * % of employment that is in public sector (2014) +
      0.0097 * Jobs Density, 2014 +
     -0.0182 * % children living in out-of-work households (2014) +
     -1.2341 * Life satisfaction score 2011-14 (out of 10) +
      1.0131 * Worthwhileness score 2011-14 (out of 10) +
     -0.6921 * Happiness score 2011-14 (out of 10) +
     14.6217
Time taken to build model: 0 seconds
=== Cross-validation ===
=== Summary ===
Correlation coefficient                  0.7033
Mean absolute error                      0.2111
Root mean squared error                  0.3202
Relative absolute error                 79.0429 %
Root relative squared error             70.1314 %
Total Number of Instances               39
Ignored Class Unknown Instances                 1
```

The **Regression Equation** is shown at the top of the window above. We will discuss what this means in the upcoming lecture on **Predicting and Classifying Numeric Data**.

14. Repeat the process by going back to step 11, but this time, select `Happiness score` (instead of Anxiety score).

15. After you have run the regression, compare the results for `Root mean squared error` or $rmse$. My answers are below:

attribute	$rmse$
`Anxiety score`	0.3202
`Happiness score`	0.1649

16. The goal is to have a lower $rmse$ value—again, something we will discuss in the upcoming lecture on **Predicting and Classifying Numeric Data**, with respect to evaluating models.

17. Experiment with predicting other numeric attributes in the data set. For which attribute do you get the best prediction? How does that change if you remove more attributes (i.e., predict with fewer)?