# DMAT – Workbook 03

## Classification via Decision Trees in WEKA

The following guide is based WEKA version 3.4.1 Additional resources on WEKA, including sample data sets can be found from the official WEKA Web site.

This example illustrates the use of C4.5 (J48) classifier in WEKA. The sample data set used for this example, unless otherwise indicated, is the bank data available in comma-separated format (bank-data.csv). This document assumes that appropriate data preprocessing has been perfromed. In this case ID field has been removed. Since C4.5 algorithm can handle numeric attributes, there is no need to discretize any of the attributes. For the purposes of this example, however, the "Children" attribute has been converted into a categorical attribute with values "YES" or "NO".

WEKA has implementations of numerous classification and prediction algorithms. The basic ideas behind using all of these are similar. In this example we will use the **modified**version of the bank data to classify new instances using the C4.5 algorithm (note that the C4.5 is implemented in WEKA by the classifier class: `weka.classifiers.trees.J48`). The modified (and smaller) version of the bank data can be found in the file "bank.aff" and the new unclassified instances are in the file "bank-new.aff".

NB The datasets provided differ slightly from those in the diagrams.

- bank.arff – 500 instances
- bank-new.arff has 100 instances

As usual, we begin by loading the data ("bank.arff") into WEKA, as seen in Figure 20:

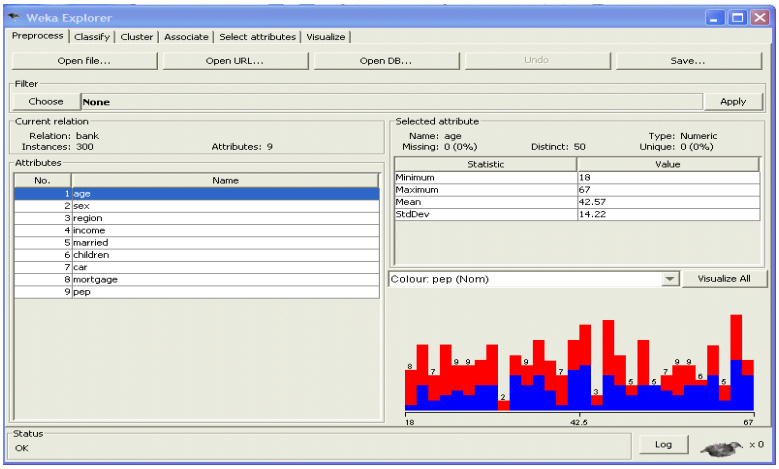


Figure 20

Next, we select the "Classify" tab and click the "Choose" button to select the J48 classifier, as depicted in Figures 21-a and 21-b. Note that J48 (implementation of C4.5 algorithm) does not require discretization of numeric attributes, in contrast to the ID3 algorithm from which C4.5 has evolved.

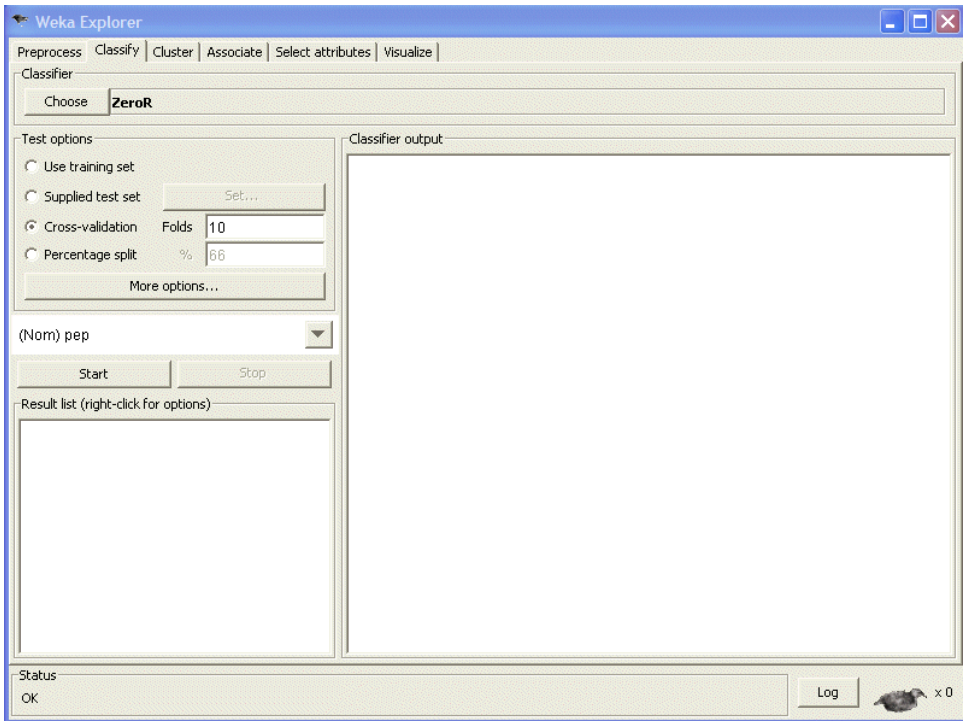


Figure 21-a

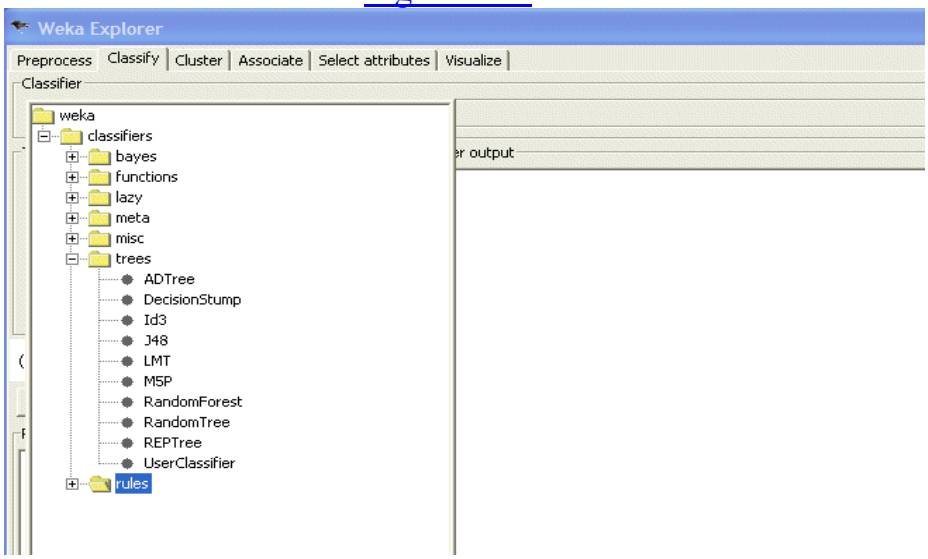


Figure 21-b

Now, we can specify the various parameters. These can be specified by clicking in the text box to the right of the "Choose" button, as depicted in Figure 22. In this example we accept the default values. The default version does perform some pruning (using the subtree raising approach), but does not perform error pruning. The selected parameters are depicted in Figure 22.

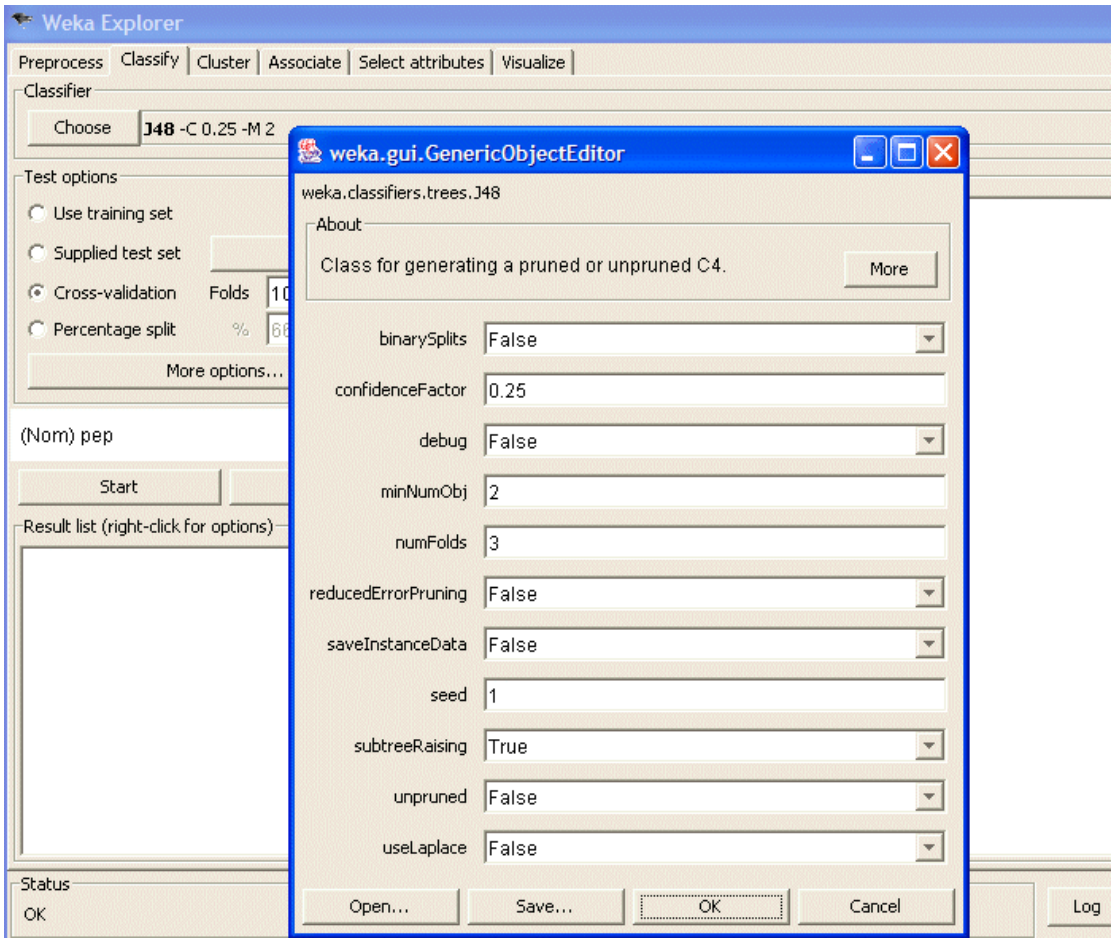


Figure 22

Under the "Test options" in the main panel we select 10-fold cross-validation as our evaluation approach. Since we do not have separate evaluation data set, this is necessary to get a reasonable idea of accuracy of the generated model. We now click "Start" to generate the model. The ASCII version of the tree as well as evaluation statistics will appear in the eight panel when the model construction is completed (see Figure 23).

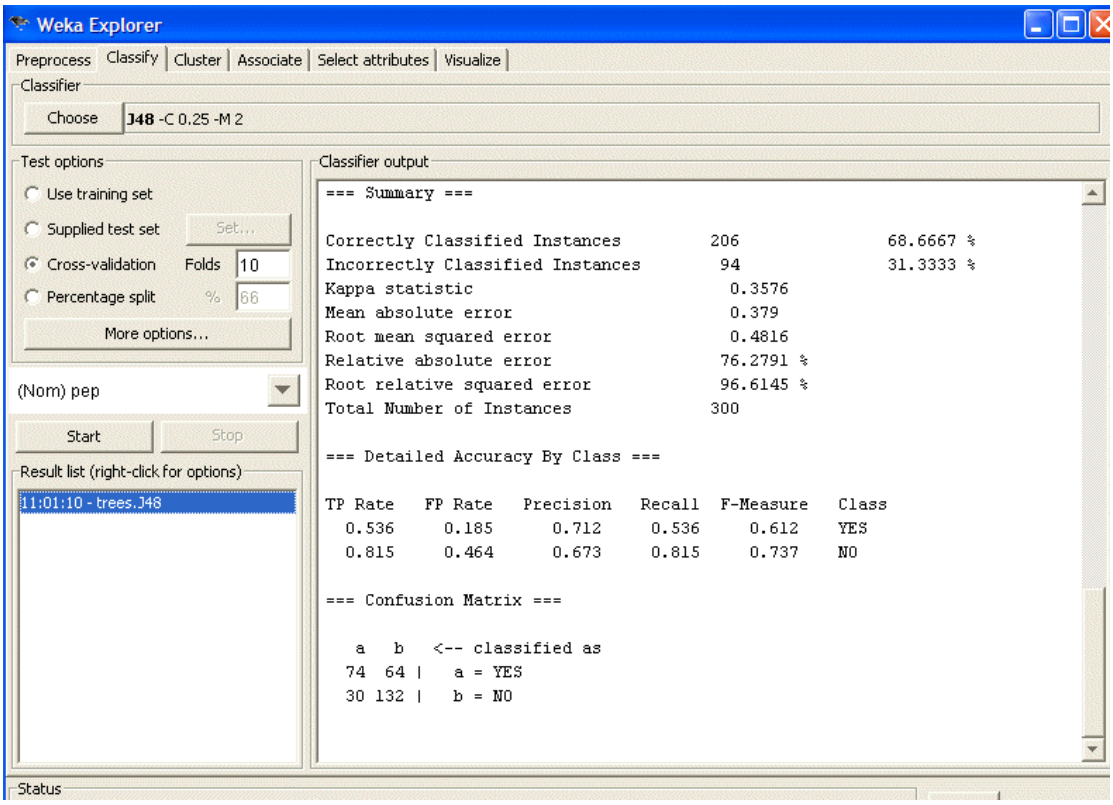


Figure 23

We can view this information in a separate window by right clicking the last result set (inside the "Result list" panel on the left) and selecting "View in separate window" from the pop-up menu. These steps and the resulting window containing the classification results are depicted in Figures 24-a and 24-b.

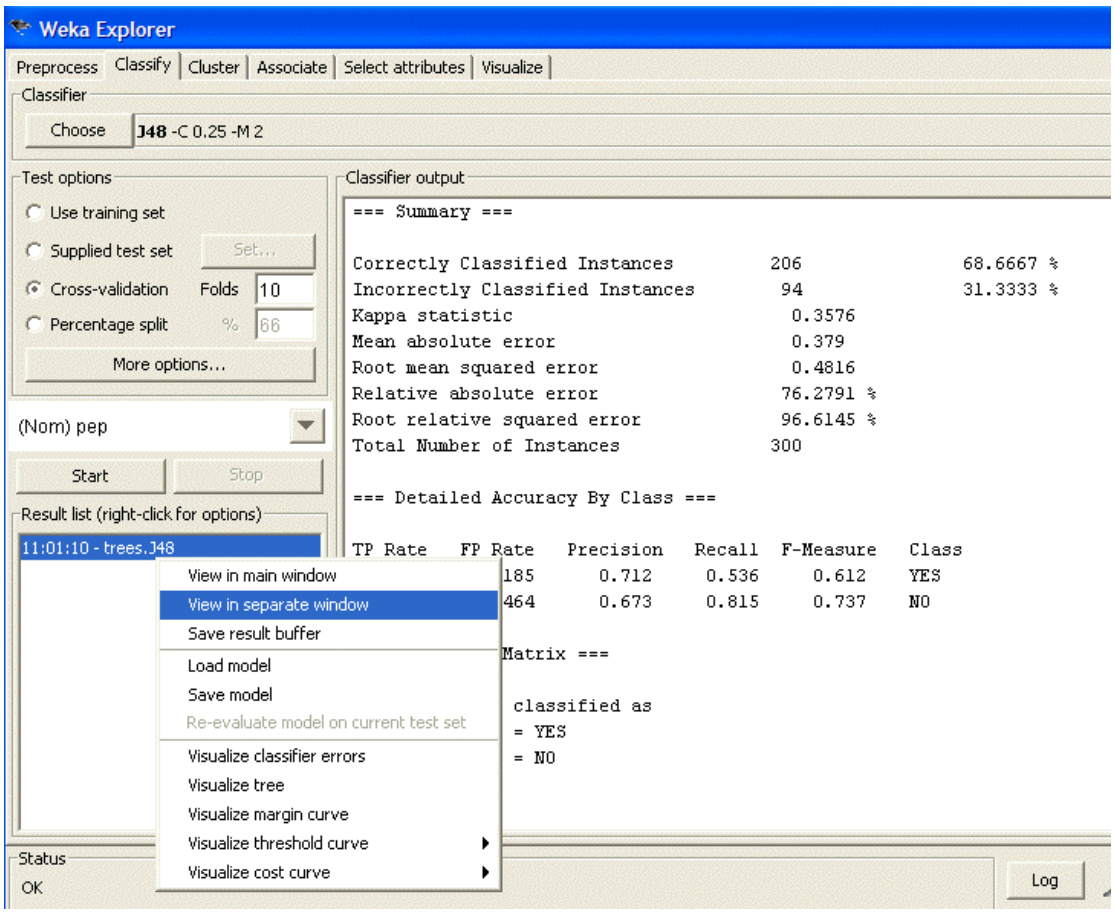


Figure 24-a

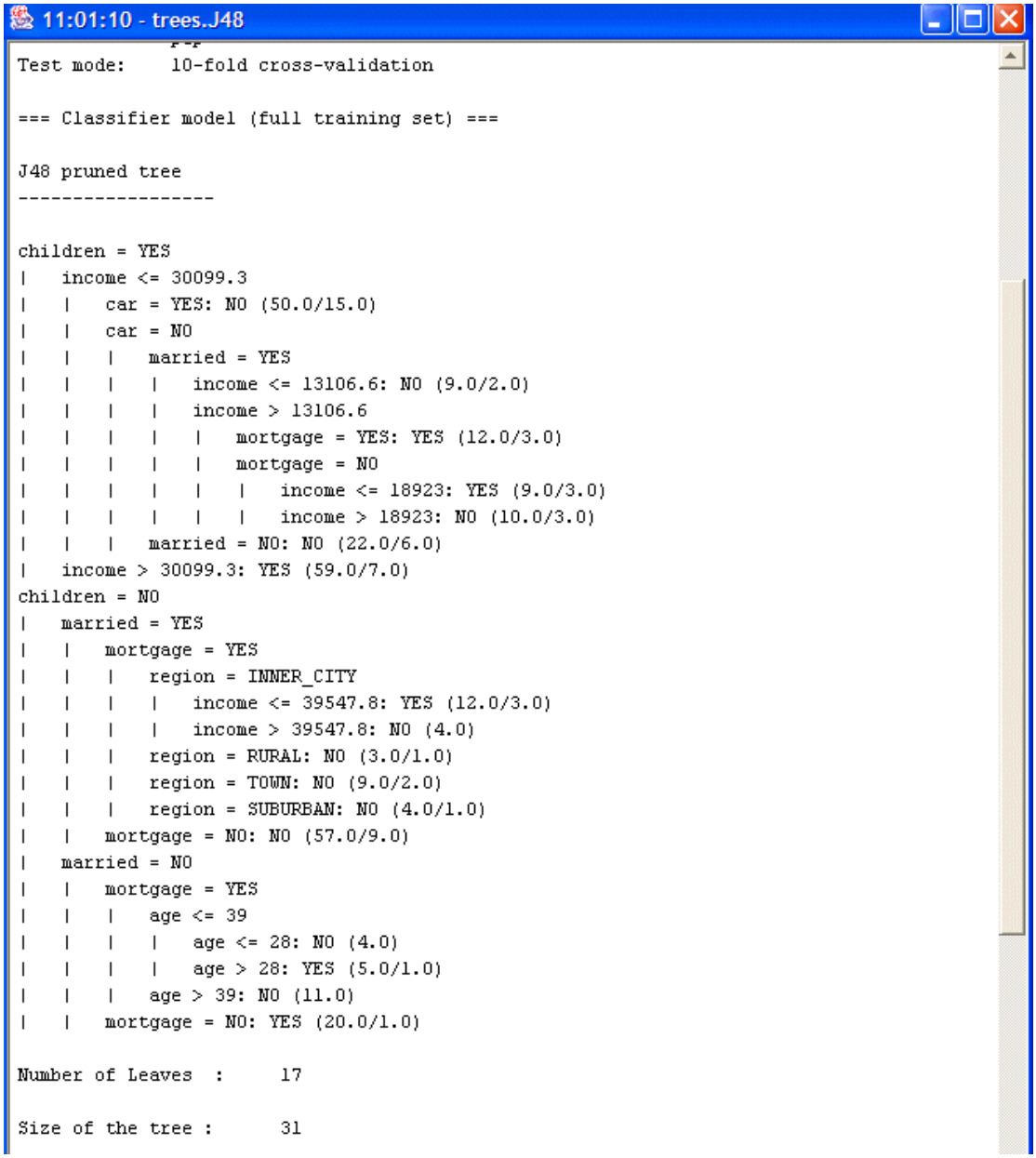
11:01:10 - trees.J48
Test mode: 10-fold cross-validation
=== Classifier model (full training set) ===
J48 pruned tree
children = YES
| income <= 30099.3
| | car = YES: NO (50.0/15.0)
| | car = NO
| | | married = YES
| | | | income <= 13106.6: NO (9.0/2.0)
| | | | income > 13106.6
| | | | | mortgage = YES: YES (12.0/3.0)
| | | | | mortgage = NO
| | | | | | income <= 18923: YES (9.0/3.0)
| | | | | | income > 18923: NO (10.0/3.0)
| | | married = NO: NO (22.0/6.0)
| income > 30099.3: YES (59.0/7.0)
children = NO
| married = YES
| | mortgage = YES
| | | region = INNER_CITY
| | | | income <= 39547.8: YES (12.0/3.0)
| | | | income > 39547.8: NO (4.0)
| | | region = RURAL: NO (3.0/1.0)
| | | region = TOWN: NO (9.0/2.0)
| | | region = SUBURBAN: NO (4.0/1.0)
| | mortgage = NO: NO (57.0/9.0)
| married = NO
| | mortgage = YES
| | | age <= 39
| | | | age <= 28: NO (4.0)
| | | | age > 28: YES (5.0/1.0)
| | | age > 39: NO (11.0)
| | mortgage = NO: YES (20.0/1.0)
Number of Leaves : 17
Size of the tree : 31

Figure 24-b

Note that the classification accuracy of our model is only about 88%. This may indicate that we may need to do more work (either in preprocessing or in selecting the correct parameters for classification), before building another model. In this example, however, we will continue with this model despite its inaccuracy.

WEKA also let's us view a graphical rendition of the classification tree. This can be done by right clicking the last result set (as before) and selecting "Visualize tree" from the pop-up menu. The tree for this example is depicted in Figure 25. Note that by resizing the window and selecting various menu items from inside the tree view (using the right mouse button), we can adjust the tree view to make it more readable.

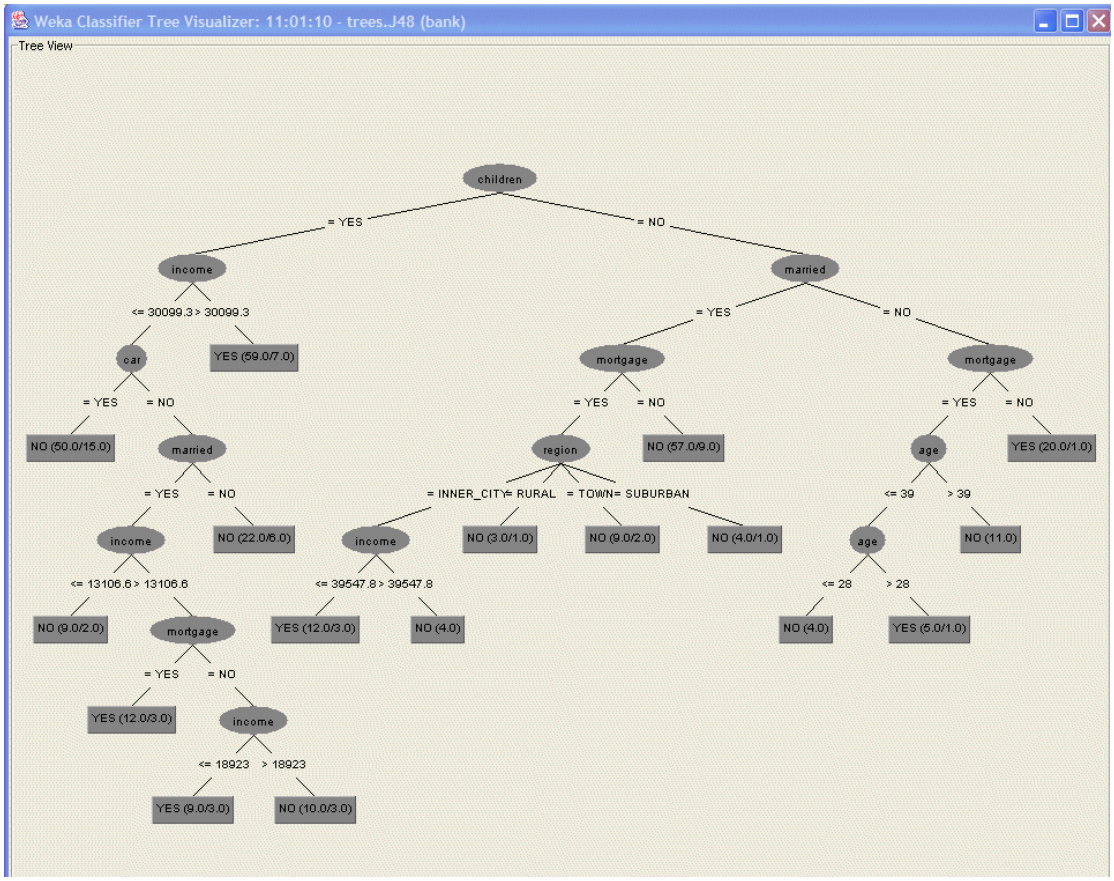


Figure 25

We will now use our model to classify the new instances. A portion of the new instances ARFF file is depicted in Figure 26. Note that the attribute section is identical to the training data (bank data we used for building our model). However, in the data section, the value of the "pep" attribute is "?" (or unknown).

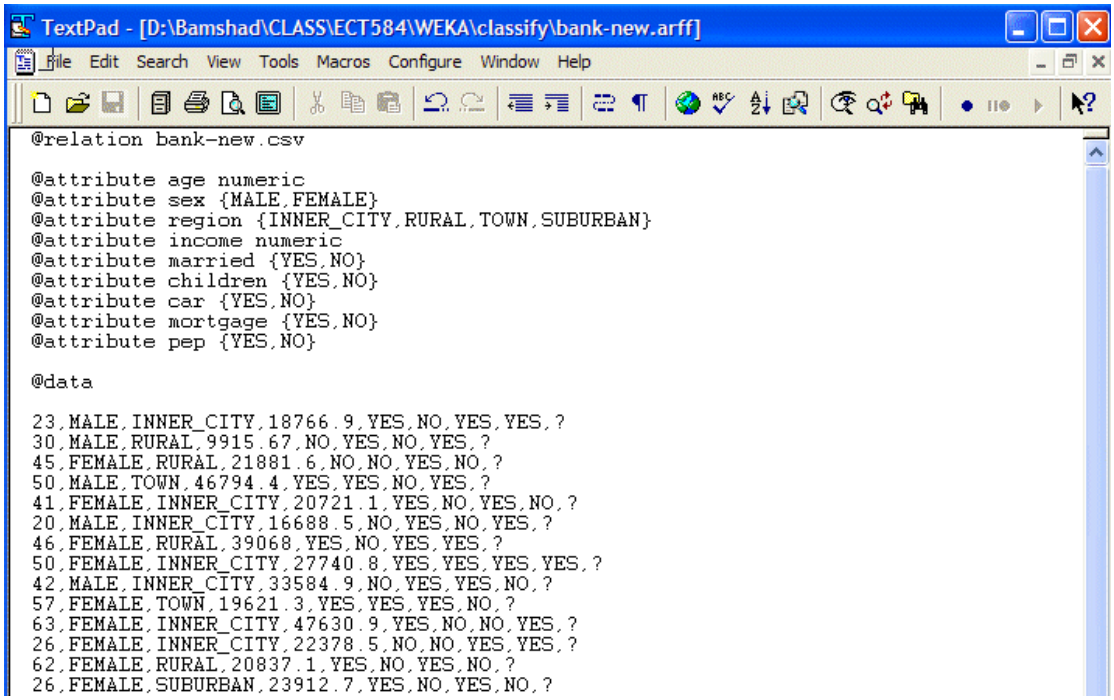


Figure 26

In the main panel, under "Test options" click the "Supplied test set" radio button, and then click the "Set..." button. This will pop up a window which allows you to open the file containing test instances, as in Figures 27-a and 27-b.

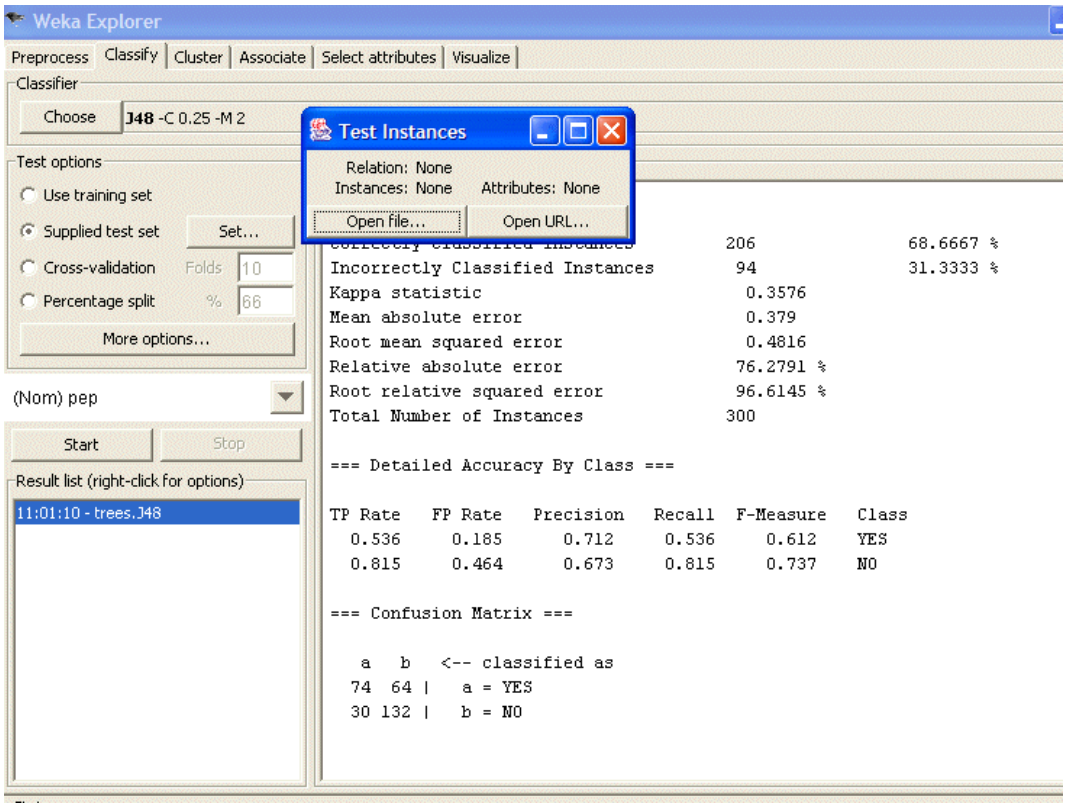


Figure 27-a

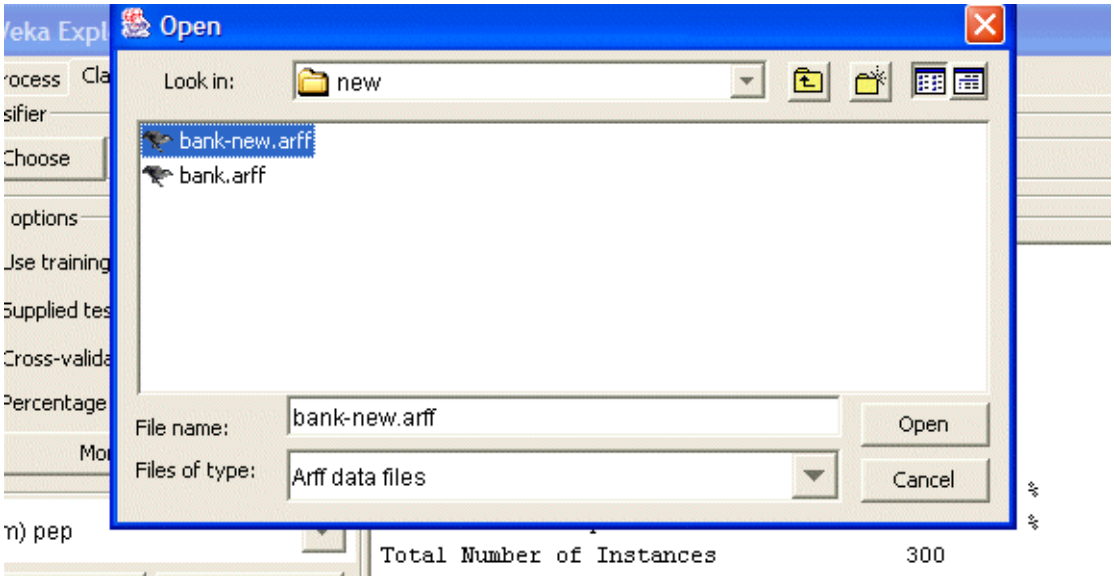


Figure 27-b

In this case, we open the file "bank-new.arff" and upon returning to the main window, we click the "start" button. This, once again generates the models from our training data, but this time it applies the model to the new unclassified instances in the "bank-new.arff" file in order to predict the value of "pep" attribute. The result is depicted in Figure 28. Note that the summary of the results in the right panel does not show any statistics. This is because in our test instances the value of the class attribute ("pep") was left as "?", thus WEKA has no actual values to which it can compare the predicted values of new instances.

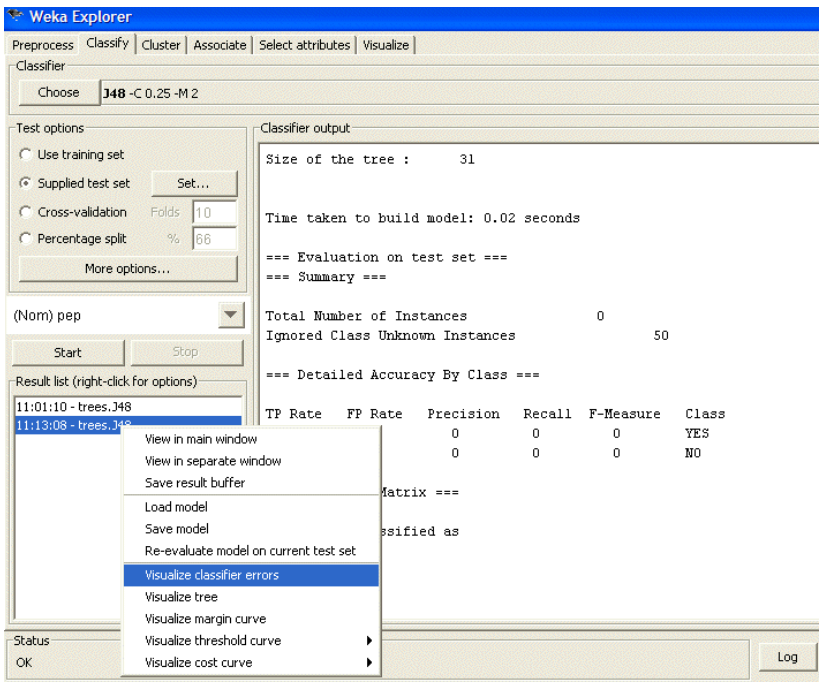


Figure 28

Of course, in this example we are interested in knowing how our model managed to classify the new instances. To do so we need to create a file containing all the new instances along with their predicted class value resulting from the application of the model. Doing this is much simpler using the command line version of WEKA classifier application. However, it is possible to do so in the GUI version using an "indirect" approach, as follows.

First, right-click the most recent result set in the left "Result list" panel. In the resulting pop-up window select the menu item "Visualize classifier errors". This brings up a separate window containing a two-dimensional graph. These steps and the resulting window are shown in Figures 28 and 29.

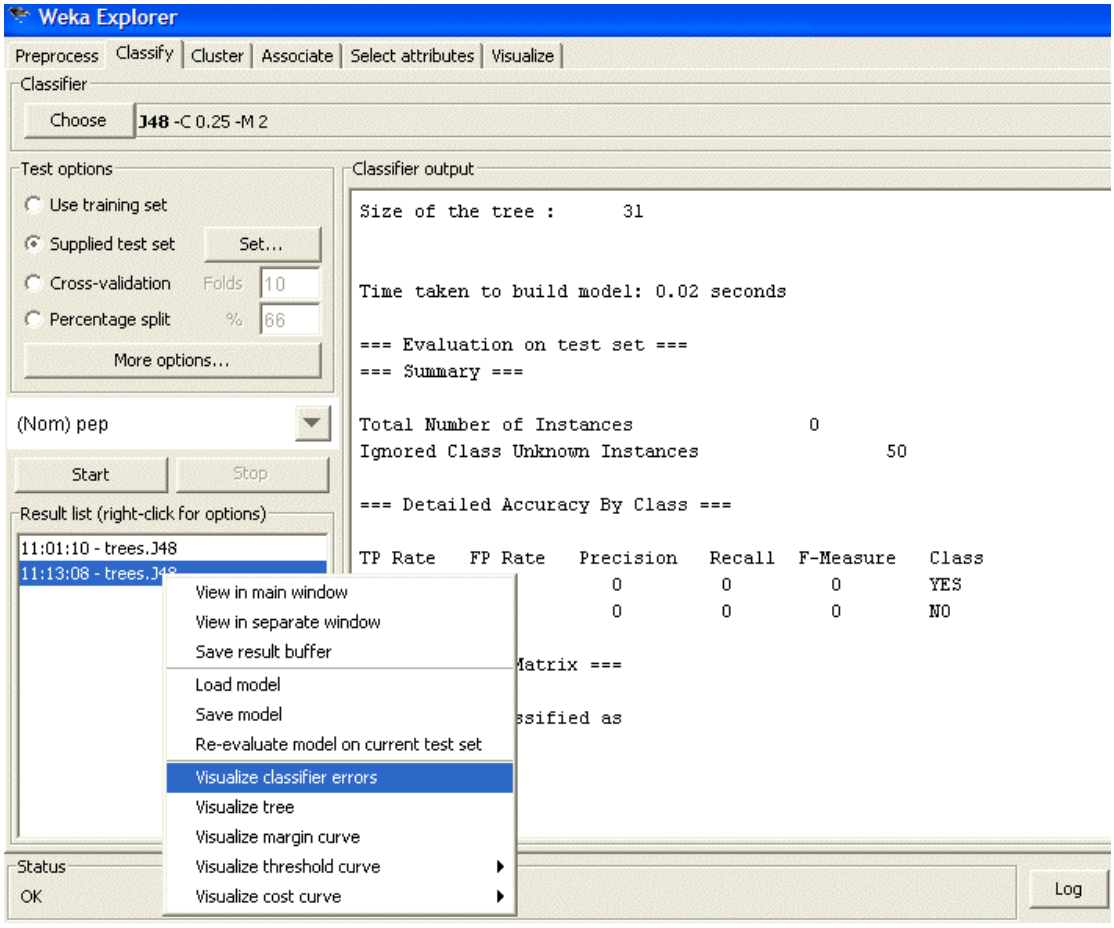


Figure 28

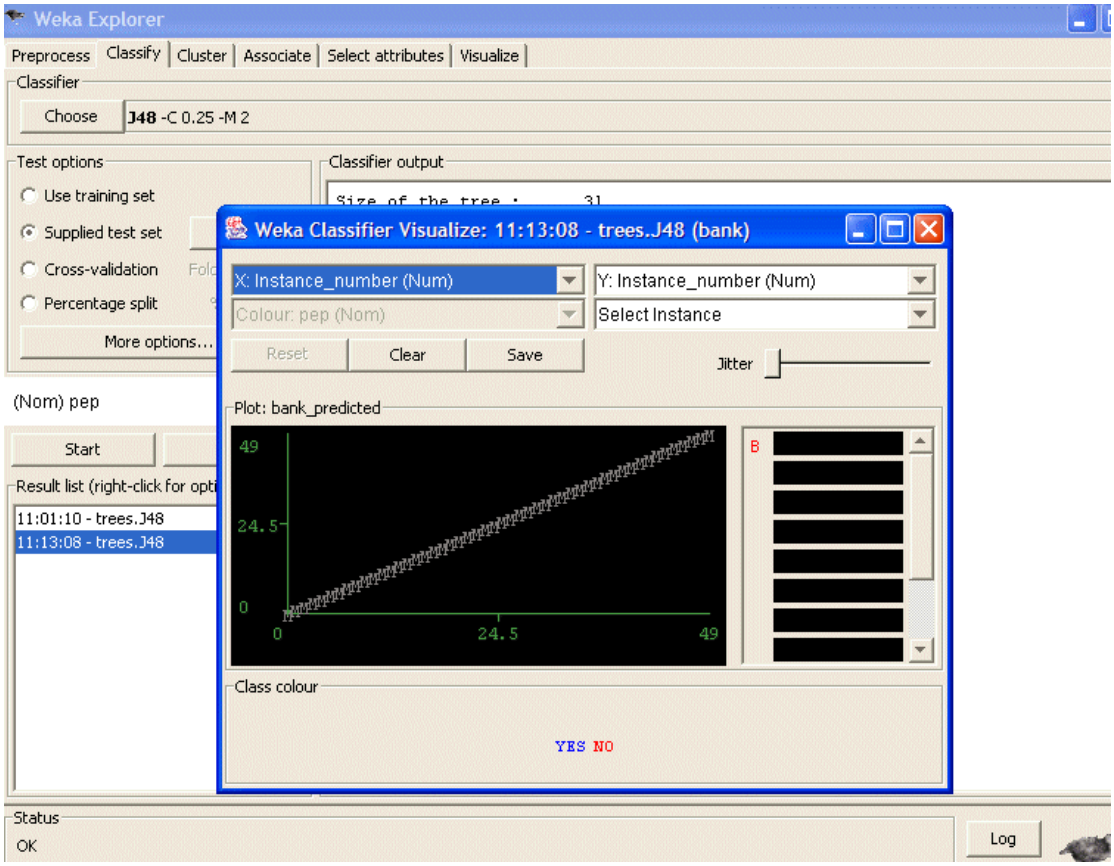


Figure 29

For now, we are not interested in what this graph represents. Rather, we would like to "save" the classification results from which the graph is generated. In the new window, we click on the "Save" button and save the result as the file: "bank-predicted.arff", as shown in Figure 30.

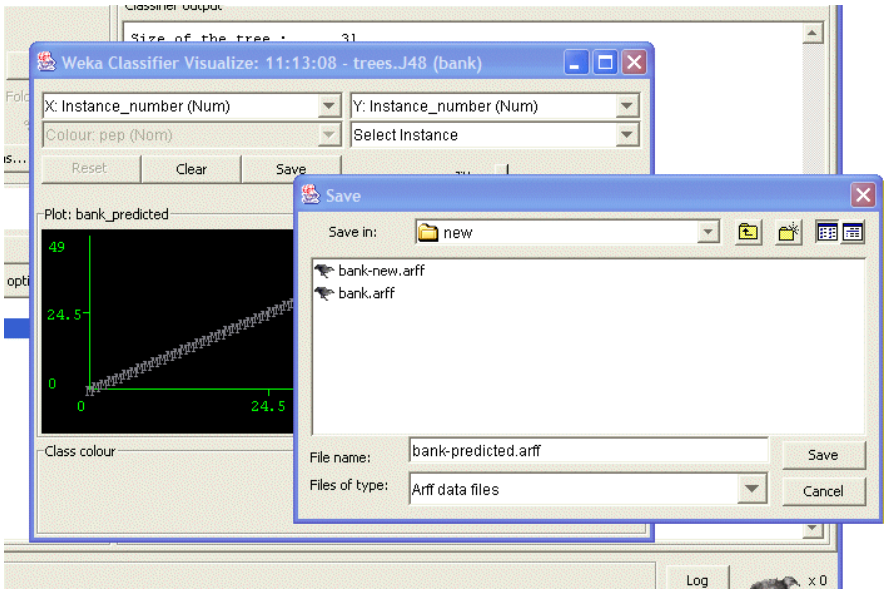


Figure 30

This file contains a copy of the new instances along with an additional column for the predicted value of "pep". The top portion of the file can be seen in Figure 31.

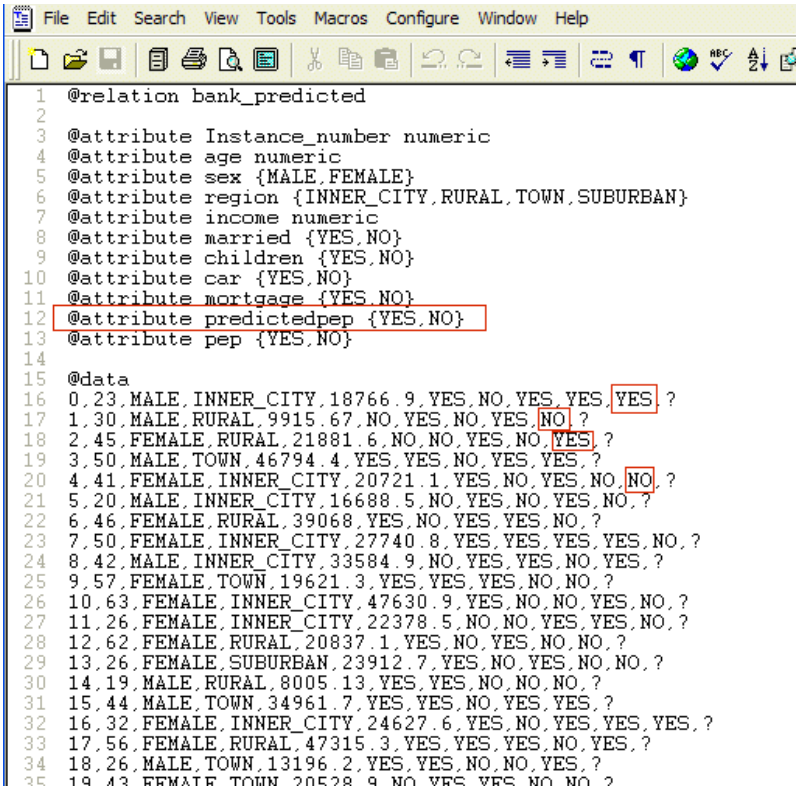


Figure 31

Note that two attributes have been added to the original new instances data: "Instance_number" and "predictedpep". These correspond to new columns in the data portion. The "predictedpep" value for each new instance is the last value before "?" which the actual "pep" class value. For example, the predicted value of the "pep" attribute for instance 0 is "YES" according to our model, while the predicted class value for instance 4 is "NO".