CP1407 Assignment 1

Note: This is an individual assignment. While it is expected that students will discuss their ideas with one another, students need to be aware of their responsibilities in ensuring that they do not deliberately or inadvertently plagiarize the work of others.

Assignment 1 – Building and Testing Classifiers in WEKA

Due date: 17th January 2018

Assessment Weight: 20%

In this assignment you will run a machine learning experiment using Weka. You will generate a model that predicts the quality of wine based on its chemical attributes. You will train the model on the supplied training data and use the model to predict the correct output for unlabeled test data.

Submission Instructions

You'll turn in your assignment as a single zip file, in LearnJCU. Specifically:

1. Create a *text* file (Word or PDF) with your answers to the questions given. Be sure you have answered all the questions. Name this file `A1-firstName-lastName.txt`.
2. Create a file containing your Weka model. Be sure this file can be loaded into Weka and that it runs. Name this file `A1-firstName-lastName.model`.
3. Create a text file in ARFF format with your predicted labels for the test set. Name this file `A1-firstName-lastName.arff`.
4. Create a single ZIP file containing (for example):
    - `A1-Joanne-Lee.txt`
    - `A1-Joanne-Lee.model`
    - `A1-Joanne-Lee.arff`
5. Turn the zip file in under Assignment 1 dropbox on LearnJCU.

Marks Breakdown

This assignment consists of nine (9) questions worth 100 marks, divided among two tasks:

- Task 1 **80 marks**
    - Part 1 (Q1) - 10 marks
    - Part 2 (Q2-4) -30 marks
    - Part 3 (Q5-7) -40 marks
- Task 2 **20 marks**
    - Q8-9

Your answers will be marked according to the rubric at the end of this document.

Task #1 - Classification for Wine

The Wine Dataset

The dataset files are provided for you:

- train.arff (labeled training set, 1890 instances)
- test.arff (unlabeled test set, 810 instances)

This dataset is adapted from:

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. *Modeling wine preferences by data mining from physicochemical properties.* In Decision Support Systems, Elsevier, 47(4):547-553. ISSN: 0167-9236.

This dataset contains data for 2700 white variants of the Portuguese "Vinho Verde" wine. For each variant, 11 chemical features were measured. Each of these is a numeric attribute. They are:

- fixed acidity
- volatile acidity
- citric acid
- residual sugar
- chlorides numeric
- free sulfur dioxide
- total sulfur dioxide
- density
- pH
- sulphates
- alcohol

Each variant was tasted by three experts. Their ratings have been combined into a single quality label: "good" or "bad" Therefore this is a *binary classification* problem with *numeric attributes*.

The dataset has been randomly split into a training set (1890 variants) and a test set (810 variants). The training set contains both chemical features and quality labels. The test set contains only the chemical features.

Task 1 - Part 1: Examine/Explore the Data

It is a good idea to inspect your data by hand before running any machine learning experiments, to ensure that the dataset is in the correct format and that you understand what the dataset contains.

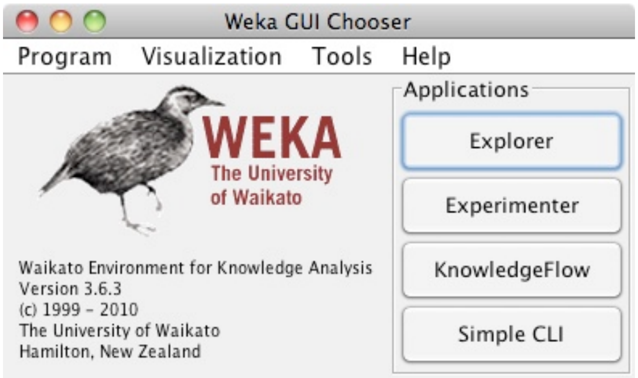
Firstly, view `train.arff` and `test.arff` in a text editor. You should see something like this:

```
@relation training

@attribute 'fixed acidity' numeric
@attribute 'volatile acidity' numeric
@attribute 'citric acid' numeric
@attribute 'residual sugar' numeric
@attribute chlorides numeric
@attribute 'free sulfur dioxide' numeric
@attribute 'total sulfur dioxide' numeric
@attribute density numeric
@attribute pH numeric
@attribute sulphates numeric
@attribute alcohol numeric
@attribute quality {good,bad}

@data
8.1,0.27,0.41,1.45,0.033,11,63,0.9908,2.99,0.56,12,bad
8.6,0.23,0.4,4.2,0.035,17,109,0.9947,3.14,0.53,9.7,bad
7.9,0.18,0.37,1.2,0.04,16,75,0.992,3.18,0.63,10.8,bad
6.6,0.16,0.4,1.5,0.044,48,143,0.9912,3.54,0.52,12.4,good
8.3,0.42,0.62,19.25,0.04,41,172,1.0002,2.98,0.67,9.7,bad
6.6,0.17,0.38,1.5,0.032,28,112,0.9914,3.25,0.55,11.4,good
6.2,0.66,0.48,1.2,0.029,29,75,0.9892,3.33,0.39,12.8,good
```

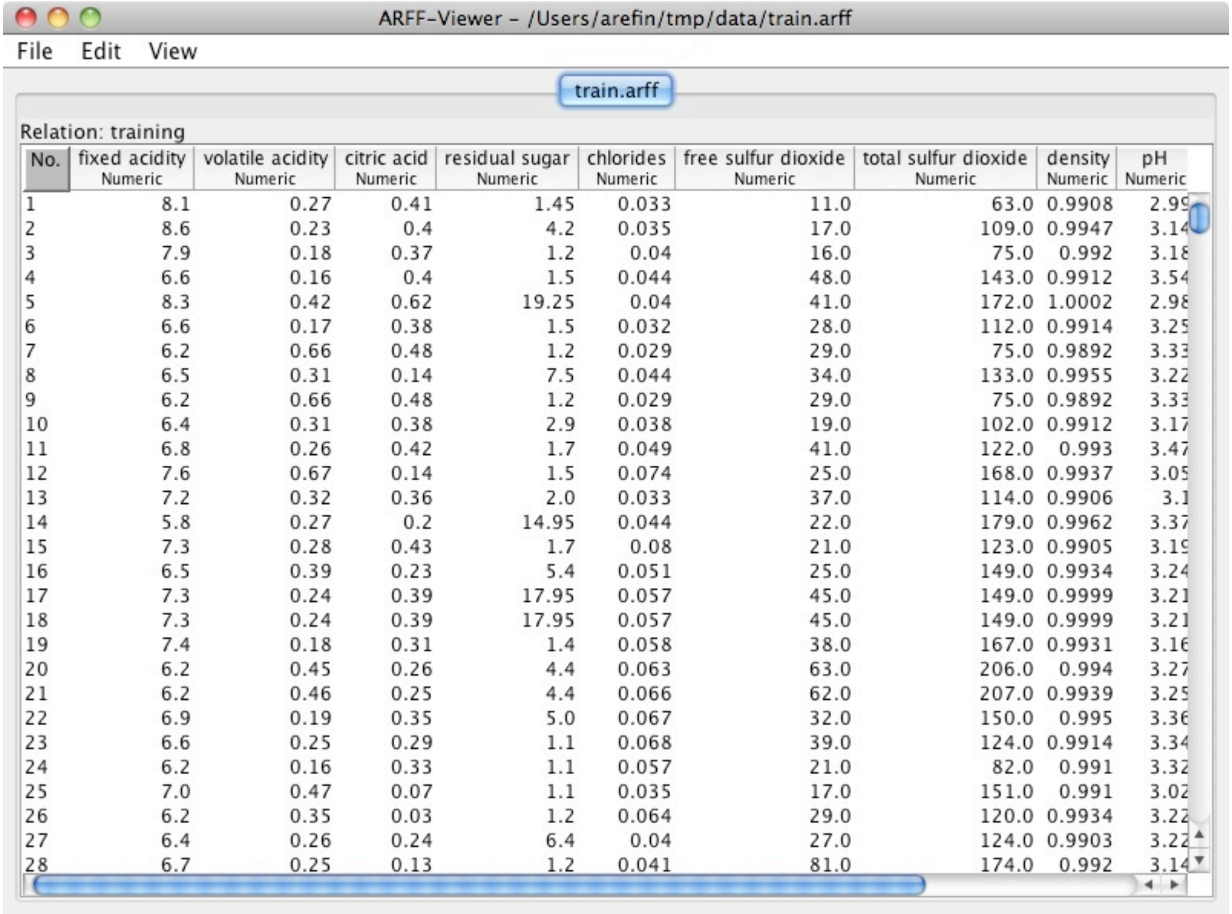
The files are in ARFF (Attribute-Relation File Format), a text format developed for Weka. At the top of each file you will see a list of attributes, followed by a data section with rows of comma separated values, one for each instance. The text and training files look similar, except that the last value for each training instance is a quality label and the last value for each test instance is a question mark, since these instances are unlabeled.

For this assignment you will not need to deal with the ARFF format directly, as Weka will handle reading and writing ARFF files for you. In future experiments you may have to convert between ARFF and another data format. (You can close the text editor.)

Another way to view .arff files is using the WEKA *ArffViewer* tool. Once you start WEKA, you will get a screen like the following:



From the *Tools* menu choose *ArffViewer*. In the window that opens, choose *File→Open* and open one of the data files. You should see something like the following:



No.	fixed acidity Numeric	volatile acidity Numeric	citric acid Numeric	residual sugar Numeric	chlorides Numeric	free sulfur dioxide Numeric	total sulfur dioxide Numeric	density Numeric	pH Numeric
1	8.1	0.27	0.41	1.45	0.033	11.0	63.0	0.9908	2.9
2	8.6	0.23	0.4	4.2	0.035	17.0	109.0	0.9947	3.14
3	7.9	0.18	0.37	1.2	0.04	16.0	75.0	0.992	3.1
4	6.6	0.16	0.4	1.5	0.044	48.0	143.0	0.9912	3.54
5	8.3	0.42	0.62	19.25	0.04	41.0	172.0	1.0002	2.9
6	6.6	0.17	0.38	1.5	0.032	28.0	112.0	0.9914	3.2
7	6.2	0.66	0.48	1.2	0.029	29.0	75.0	0.9892	3.3
8	6.5	0.31	0.14	7.5	0.044	34.0	133.0	0.9955	3.2
9	6.2	0.66	0.48	1.2	0.029	29.0	75.0	0.9892	3.3
10	6.4	0.31	0.38	2.9	0.038	19.0	102.0	0.9912	3.1
11	6.8	0.26	0.42	1.7	0.049	41.0	122.0	0.993	3.4
12	7.6	0.67	0.14	1.5	0.074	25.0	168.0	0.9937	3.0
13	7.2	0.32	0.36	2.0	0.033	37.0	114.0	0.9906	3.1
14	5.8	0.27	0.2	14.95	0.044	22.0	179.0	0.9962	3.37
15	7.3	0.28	0.43	1.7	0.08	21.0	123.0	0.9905	3.1
16	6.5	0.39	0.23	5.4	0.051	25.0	149.0	0.9934	3.24
17	7.3	0.24	0.39	17.95	0.057	45.0	149.0	0.9999	3.21
18	7.3	0.24	0.39	17.95	0.057	45.0	149.0	0.9999	3.21
19	7.4	0.18	0.31	1.4	0.058	38.0	167.0	0.9931	3.1
20	6.2	0.45	0.26	4.4	0.063	63.0	206.0	0.994	3.2
21	6.2	0.46	0.25	4.4	0.066	62.0	207.0	0.9939	3.2
22	6.9	0.19	0.35	5.0	0.067	32.0	150.0	0.995	3.3
23	6.6	0.25	0.29	1.1	0.068	39.0	124.0	0.9914	3.34
24	6.2	0.16	0.33	1.1	0.057	21.0	82.0	0.991	3.3
25	7.0	0.47	0.07	1.1	0.035	17.0	151.0	0.991	3.0
26	6.2	0.35	0.03	1.2	0.064	29.0	120.0	0.9934	3.2
27	6.4	0.26	0.24	6.4	0.04	27.0	124.0	0.9903	3.2
28	6.7	0.25	0.13	1.2	0.041	81.0	174.0	0.992	3.14

Here you see the same data as in the text editor, but parsed into a spreadsheet-like format. Although you will not need the ArffViewer for this assignment, it is a useful tool to know about when working with Weka. (You can close the ArffViewer window.)

Important Note

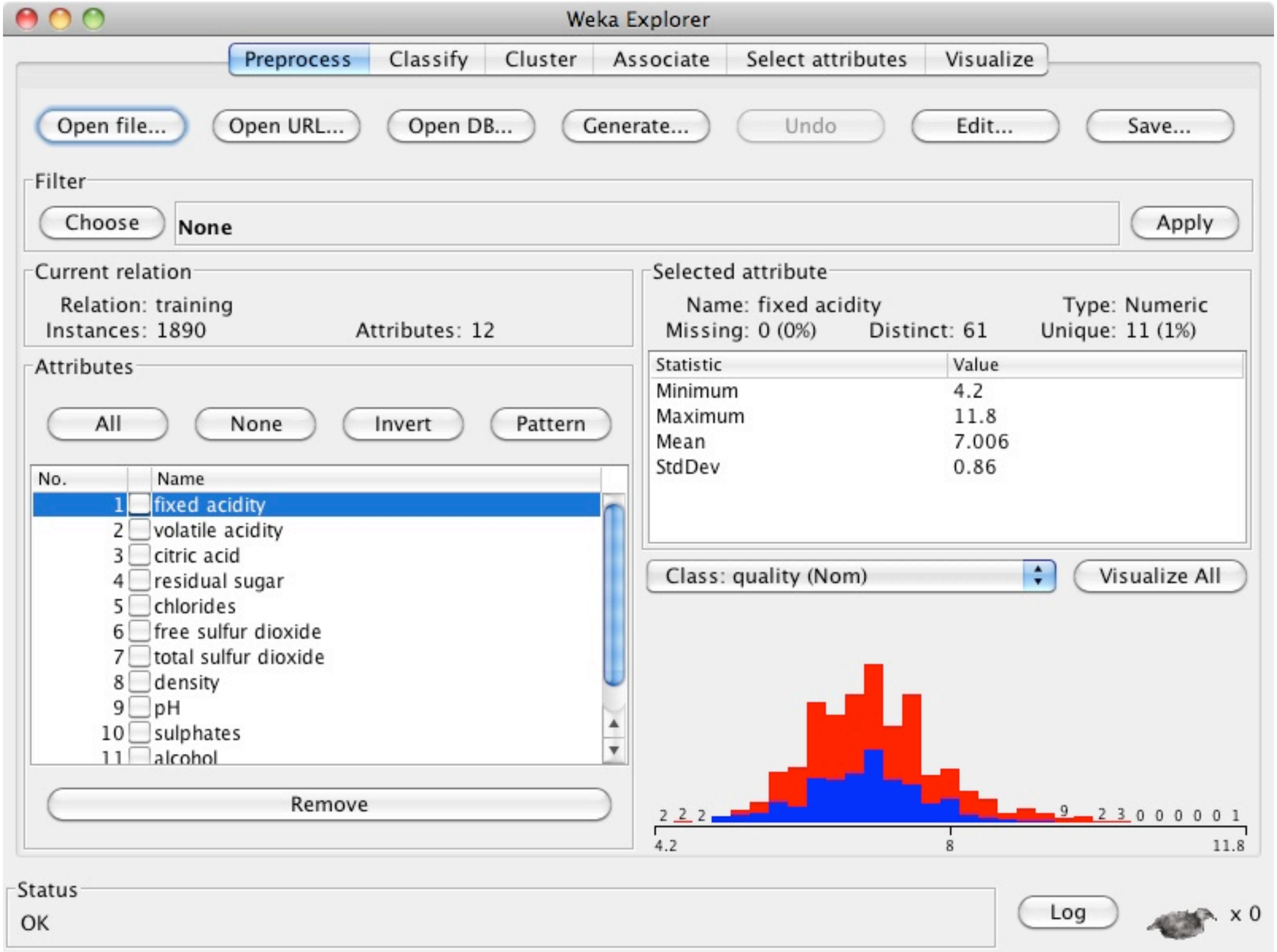
You may find that the ARFF files are grayed out and that the *All Files* option needs to be selected from the *File Format* dropdown menu for the files to be selectable. However, the ARFF Viewer may still not read the files properly. If such is the case, it is likely that a .txt extension got appended to the filename when the files were downloaded. However, even if the files are downloaded without .txt getting appended or an inadvertently added .txt extension is removed, the ARFF Viewer may have trouble reading the files properly. The following steps should resolve the issue:

1. View the ARFF in your Web browser by clicking on the link in the instructions or open the downloaded ARFF file in a text editor.
2. Copy all the text and paste it to a new text file.
3. If you copied the ARFF contents from the downloaded ARFF file, it is recommended that you do not overwrite the downloaded ARFF file when saving the new file on the next step. Instead, delete the downloaded ARFF file.
4. Save the new text file with a **.arff** extension, carefully making sure that a .txt extension does not get appended.
5. Open the newly saved ARFF file in the Weka ARFF Viewer to verify the Viewer can display the file in the manner illustrated in the image above.

CP1407 Assignment 1

After getting the initial examination on the data files provided using a general text editor or ArffViewer tool, choose the Explorer interface in WEKA. (From the Weka GUI Choose click on the *Explorer* button to open the Weka Explorer.) The Explorer is the main tool in Weka, and the one you are most likely to work with when setting up an experiment. For the remainder of this assignment you will work within the Weka Explorer.

The Explorer should open to the "Preprocess" tab. The Preprocess tab allows you to inspect and modify your dataset before passing it to a machine learning algorithm. Click on the button that says "Open file..." and open `train.arff`. You should see something like this when the dataset is correctly uploaded:



The attributes are listed in the bottom left, and summary statistics for the currently selected attribute are shown on the right side, along with a histogram. Click on each attribute (or use the down arrow key to move through them) and look at the corresponding histogram. You will notice that many numeric attributes have a "hump" shape; this is a common pattern for numeric attributes drawn from real-world data.

You will also notice that some attributes appear to have outliers on one or both sides of the distribution. The proper treatment of outliers varies from one experiment to another. For this assignment you can leave the outliers alone.

Now answer the question below:

Question #1:

Which attributes in the training set do not appear to have a "hump" distribution? Which attributes appear to have outliers? (Do not worry too much about being precise here. The point is for you to inspect the data and interpret what you see.)

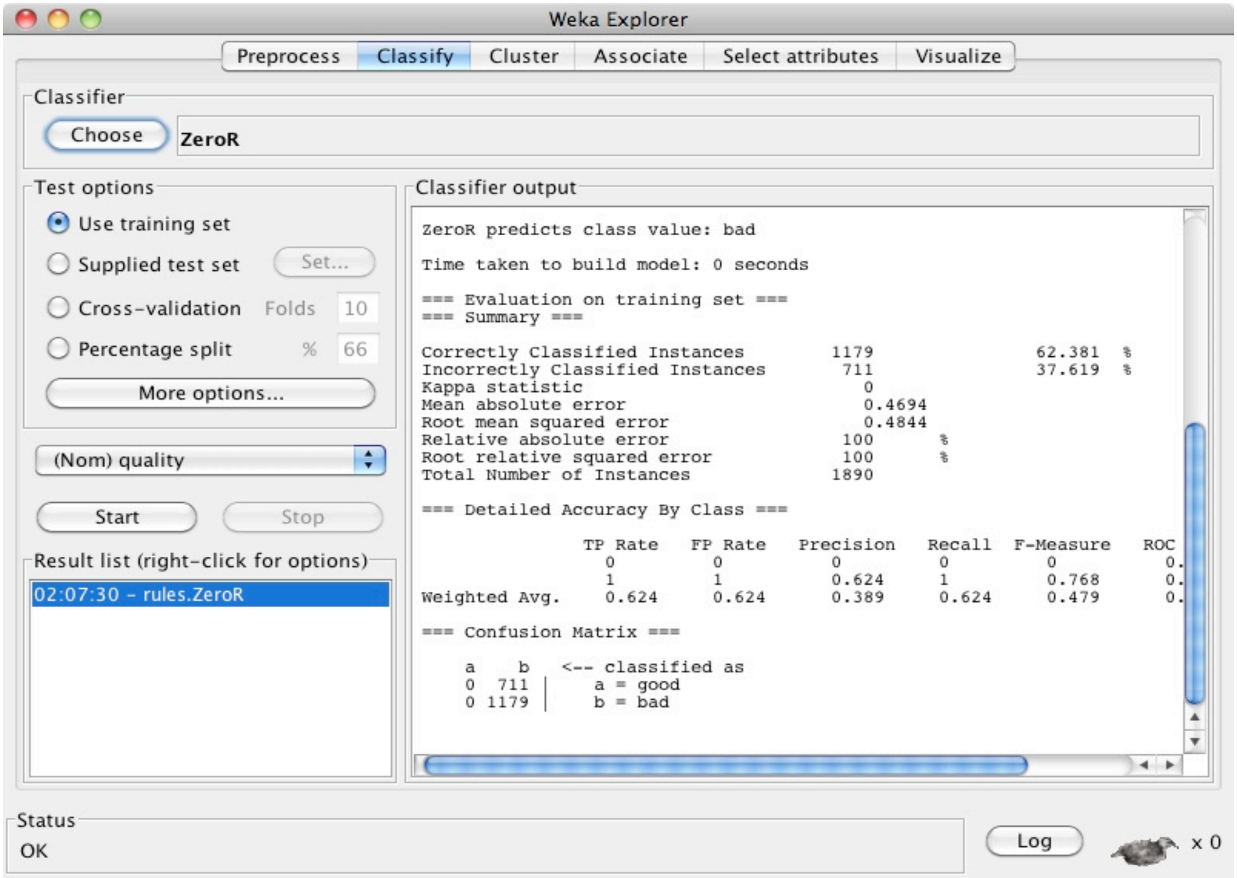
Based on the histogram, which attribute appears to be the most useful for classifying wine, and why?

Task 1 – Part 2: Classifier Basics

In this section you will train couple of basic classifiers on the data.

Baseline Classifier

Click on the "Classify" tab. Choose *ZeroR* as the Classifier if it is not already chosen (it is under the "rules" subtree when you click on the "Choose" button). When used in a classification problem, *ZeroR* simply chooses the majority class. Under "Test options" select "Use training set", then click the "Start" button to run the classifier. You should see something like this:



The classifier output pane displays information about the model created by the classifier as well as the evaluated performance of the model. In the Summary section, the row "Correctly Classified Instances" reports the accuracy of the model.

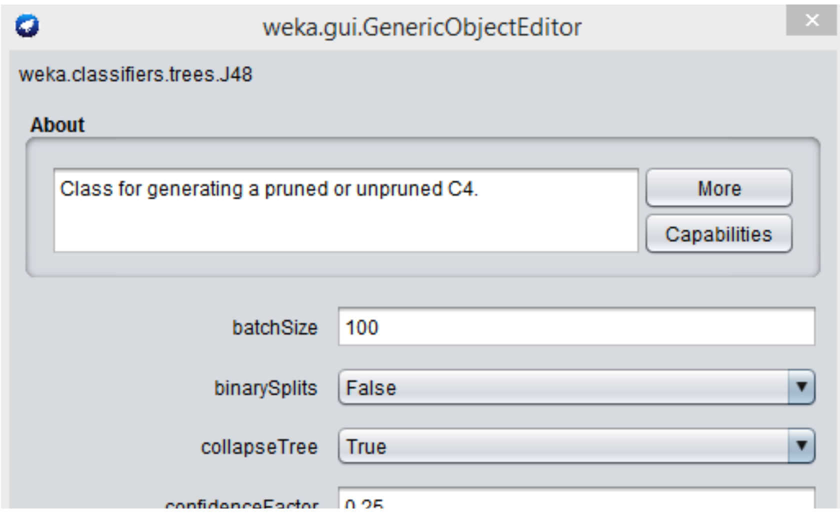
Now answer the question below:

> **Question #2:**
>
> What is the *accuracy* - the percentage of correctly classified instances - achieved by *ZeroR* when you run it on the training set? Explain this number (what this number means …). How is the accuracy of *ZeroR* a helpful baseline for interpreting the performance of other classifiers?

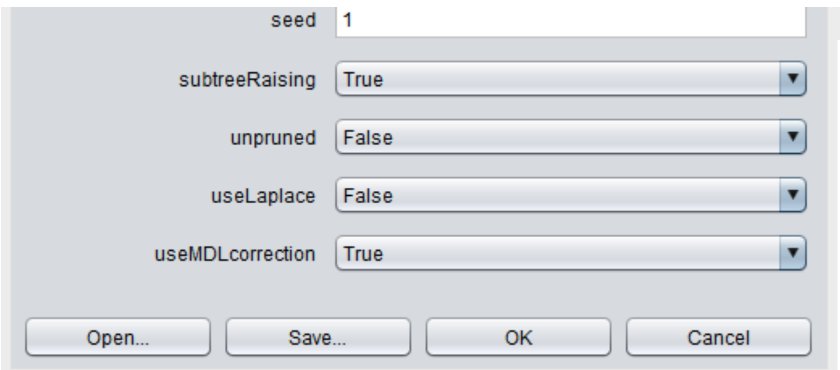
Decision Trees

J48 is the Weka implementation of the C4.5 decision tree algorithm.

Click on the "Choose" button and select *J48* under the "trees" section. Notice that the field to the right of the "Choose" button updates to say "J48 -C 0.25 -M 2". This is a command-line representation of the current settings of *J48*. Click on this field to open up the configuration dialog for *J48*:



…



Each classifier has a configuration dialog such as this that shows the parameters of the algorithm as well as buttons at the top for more information. When you change the settings and close the dialog, the command line representation updates accordingly. For now we will use the default settings, so hit "Cancel" to close the dialog.

Under "Test options" select "Use training set", then click the "Start" button to run the classifier. After the classifier finishes, scroll up in the output pane. You should see a textual representation of the generated decision tree.

Now answer the question below:

Question #3:

Using a decision tree Weka learned over the training set, what is the most informative single feature for this task, and what is its influence on wine quality? Does this match your answer from Question #1?

Scroll back down and record the percentage of Correctly Classified Instances. Now, under "Test options", select "Cross-validation" with 10 folds. Run the classifier again and record the percentage of Correctly Classified Instances.

In both cases, the final model that is generated is based on all of the training data. The difference is in how the accuracy of that model is estimated.

Now answer the question below:

Question #4:

What is 10-fold cross-validation? What is the *main* reason for the difference between the percentage of Correctly Classified Instances when you used the entire training set directly versus when you ran 10-fold cross-validation on the training set? Why is cross-validation important?

Task 1 – Part 3: Build Your Own Classifier

This is the main part of the assignment. Search through the classifiers in Weka and run some of them on the training set. You may want to try varying some of the classifier parameters as well. **Choose the one** you feel is most likely to generalize well to unseen examples--namely the unlabeled examples in the test set. Feel free to use validation strategies other than 10-fold cross-validation.

When you have built the classifier you want to submit, move on to the following sections.

Saving the Model

To export a classifier model you have built:

1. Right-click on the model in the "Result list" in the bottom left corner of the Classify tab.
2. Select "Save model".
3. In the dialog that opens, ensure that the File Format is "Model object files"
4. Save the model using the naming convention given as instructed (e.g. `A1-Joanne-Lee.model`).

In order to grade your assignment it must be possible to load your model file in Weka and run it on a labeled version of `test.arff`. You can load your model by right-clicking in the Result list pane and selecting "Load model".

Generating Predictions

To generate an ARFF file with predictions for the test data, perform the following steps from within the Classify tab. This assumes you already have a trained model in the Result list, which you will run on the test set. You will produce either an ARFF file or a CSV file containing your predictions, *either* is fine for the assignment, but regardless submit your output file as `A1-firstName-lastName.arff`. Our recommended steps depend on which version of Weka you're using, as detailed below.

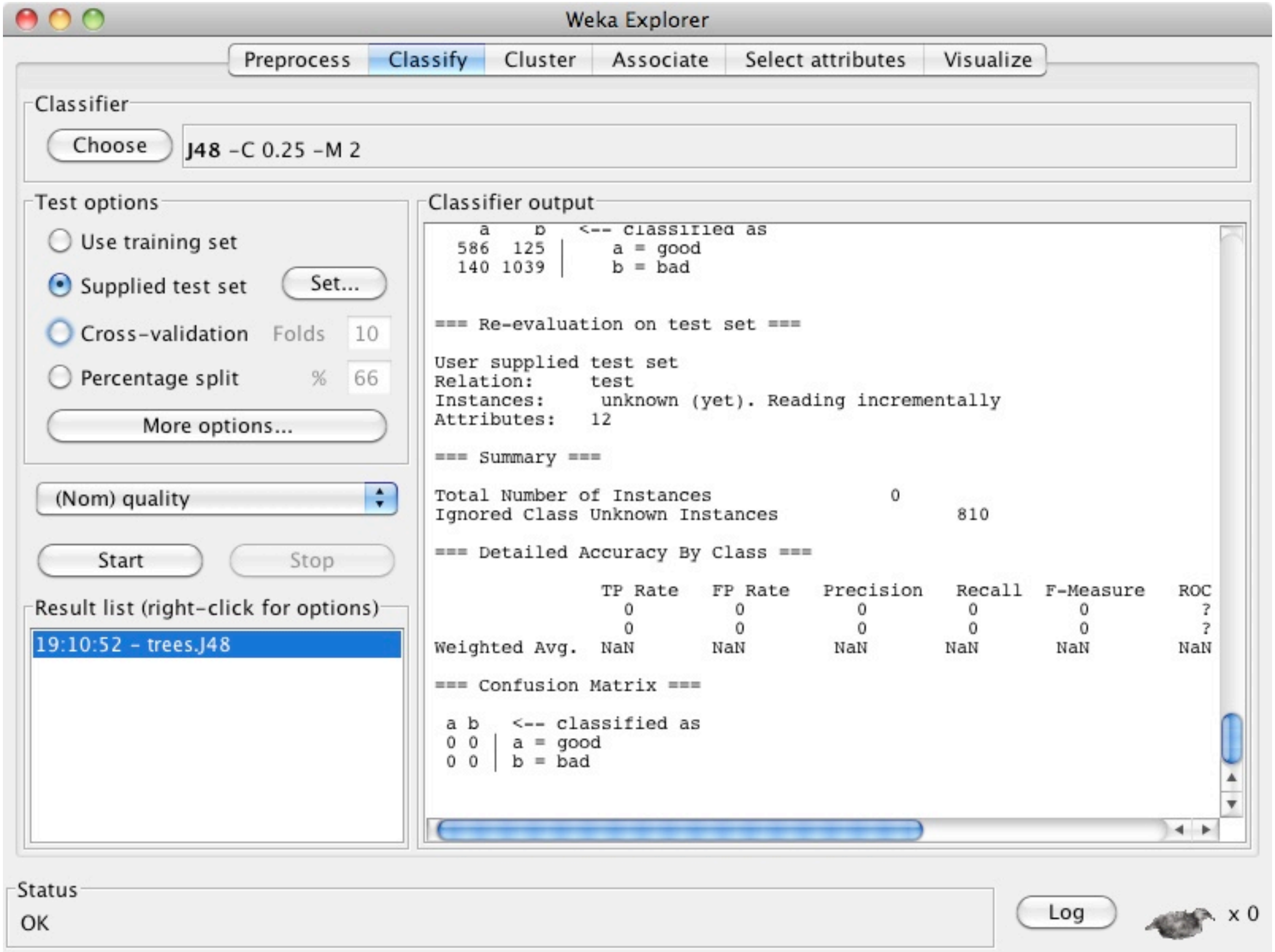
In Weka 3.8:

1. Under "Test options" select "Supplied test set".
2. Click on the "Set..." button.
3. In the "Test Instances" dialog that opens click "Open file...".
4. Open `test.arff`.
5. Close the Test Instances dialog.
6. In Test options, click More options...
7. Click Choose in Output predictions and select CSV
8. Click the text "CSV", click the outputFile box and enter a location to save -- name your file `A1-firstName-lastName.arff` for the purpose of the assignment even though the output will not *actually* be an ARFF file. (It's a CSV File, but Weka doesn't give us the option of outputting ARFF. If you did want an ARFF file to work with, you could convert your CSV into ARFF by naming it A1.csv and opening it in the ARFF viewer, and then saving as ARFF. You could also turn that file in as A1-firstName-lastName.arff, but you don't have to.)
9. In attributes, enter the string "1-11"; this will output the other attributes for each test example
10. Right-click on your model in the Result list, select Re-evaluate current model on the test set.
11. Submit `A1-firstName-lastName.arff` as instructed.

In older versions (verified in Weka 3.6):

1. Under "Test options" select "Supplied test set".
2. Click on the "Set..." button.

3. In the "Test Instances" dialog that opens click "Open file...".
4. Open `test.arff`.
5. Close the Test Instances dialog.
6. Right-click on your model in the Result list and select "Re-evaluate model on current test set". Your output will look something like the picture below. Notice that the output contains a bunch of NaNs. This is because the test data is unlabeled and therefore Weka cannot compute the accuracy.
7. Right-click again on your model and select "Visualize classifier errors".
8. In the dialog that opens, click on the "Save" button.
9. Save the ARFF file using the naming convention as instructed (e.g. `A1-Joanne-Lee.arff`).



Now answer the question below (Questions #5 and #6):

Question #5:

What is the "command-line" for the model you are submitting? For example, *"J48 -C 0.25 -M 2"*. What is the reported accuracy for your model using 10-fold cross-validation?

Question #6:

In a few sentences, describe how you chose the model you are submitting. Be sure to mention your validation strategy and whether you tried varying any of the model parameters.

Building various versions of your selected model

With the classifier model you selected, try to variate the model by setting various configurations. You can do this by trying different values for parameters. For example, if you chose the J48 decision tree classifier, your initial model might be built using default configuration (parameter setting). To validate your model, you can try to set alternative option "True" for "unpruned" parameter. The result of running this model with the same train/test data set may be different. Try yourself to configure variously parameters and save them as different versions of your classifier model.

Now answer the question below:

Question #7:

For your selected classifier model, investigate what kind of parameters are available to be set in WEKA and briefly summarise of the role of each parameter.
Throughout the various configuration testing, summarise your findings about the effect (on the result of testing the model using test data) of three different parameters of your choice.

Task #2 – Try Another Data Set

You will now build a classifier for a second data set concerning the evaluation of cars, following which you will answer the last two questions.

In order to answer the questions, perform the following steps:

1. Download the Car Evaluation Dataset

The car evaluation dataset files are provided (see important note below):

- car_train.data (labeled training set, 1190 instances)
- car_train.names (auxiliary file for training set)
- car_test.data (unlabeled test set, 510 instances)
- car_test.names (auxiliary file for test set)

This dataset is adapted from:

Car Evaluation Database, which was derived from a simple hierarchical decision model originally developed for the demonstration of DEX, M. Bohanec, V. Rajkovic: *Expert system for decision making.* Sistemica 1(1), pp. 145-157, 1990.).

Important Note

The main data file for the car evaluation data set ends in a **.data** extension and has an associated auxiliary data file ending in a **.names** extension. However, the usage for the .data file is the same as for the .arff file you are already familiar with, though you have to pay special attention to the following:

1. When opening the main data file (car_train.data or car_test.data) in the Weka Explorer, the *C4.5 data files (*.data)* option needs to be selected from the *File Format* dropdown menu.
2. The auxiliary data files (car_train.names and car_test.names) must be located in the same folder as the main data files. (You do not need to take any action on these auxiliary data files other than to keep them in the same folder as the main data files, but inspecting the contents of the files should help you interpret how the main data files work.)

2. Build Classifiers

You will perform four experiments, measuring the 10-fold cross-validation accuracy of two types of classifiers (call them classifiers A and B) on two data sets (cars and wine). You can choose A and B however you like -- they can be different classifiers (nearest-neighbor vs. decision trees) or the same classifier with different settings (different numbers of nearest neighbors, for example). Your goal is to choose settings such that the A classifier performs great for wine evaluation, but poorly for car evaluation, and vice-versa for classifier B. In other words, you should strive to find a value as large as you can for the expression below:

wine_acc(A) + car_acc(B) – wine_acc(B) – car_acc(A)

where *wine_acc(A)* refers to the accuracy of Classifier A on the wine data set, and *car_acc(B)* refers to the accuracy of Classifier B on the car data set, and so on.

Note: You do not need to obtain the largest possible quantity for the above expression, and it is okay to use classifiers as long as you can achieve some positive value for the above expression (a value of 2% is sufficient for the assignment).

Note that you will only need to use the training data (`car_train.data`) for this task. The test data (`car_test.data`) is provided for your personal reference, should you choose to try your car evaluation classifier on it to gain experience. Therefore, you do not need to perform the steps under "Generating Predictions" (you have done for Task#1) for this task.

Now answer the question below (Questions #8 and #9):

Question #8: Briefly explain what strategy you used to obtain the Classifiers A and B that performed well on one of the car or wine data sets, and not the other.

Question #9: Name one major difference between the output space for the car data set vs. the wine data set, that might make some classifiers that are applicable to the wine data not applicable to the car data.

Rubrics for Assignment

Grade		Criteria
Exemplary	100%	Answer demonstrates excellent knowledge of machine learning and data science, is well-written, and very well-justified.
Good	80%	Answer demonstrates good knowledge of machine learning and data science, and is generally well-written and well-justified.
Fair	60%	Answer demonstrates sound knowledge of machine learning and data science and provides justification.
Limited	40%	Answer demonstrates some knowledge of machine learning and data science, but may contain inaccuracies. An attempt at justification is made.
Very Limited	20%	Answer demonstrates flawed knowledge of machine learning and/or provides incoherent justification.
Unacceptable	0%	Not attempted or totally irrelevant answer