**IAML LAB 2: ATTRIBUTE SELECTION & REGRESSION**

In this lab we will examine two methods of feature selection. First, we will use weka's visualization tool to manually assess the attributes' ability to discriminate between classes. Then we will use an automated attribute selector, the Information Gain measure. We will also deal with a simple regression problem in detail. For attribute selection we will examine the Landsat satellite image dataset. For regression we will use a CPU performance data set and try to predict the value of a real-valued attribute given the rest of the attributes.

**LANDSAT DATA**

The second data set to be used is the [Landsat](http://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite)) satellite imaging set. Acquaint yourself with this by reading [this description](http://www.inf.ed.ac.uk/teaching/courses/dme/html/landsat.html).

Most importantly, note that each instance is a 3 × 3 region of pixels with recordings in 4 different spectral bands for each pixel. The task is to classify the instances according to the soil type of the **centre** pixel.

A slightly modified data set is used for this lab. Download the following file:

* [landsat.arff](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/landsat.arff): landsat data

Load this dataset into Weka through the *Preprocess* tab. In this case no further preprocessing is required.

In any real world application it is worth visualising a data set before attempting to train a model with it. Bring up the *Visualize* tab in Weka. This plots all attributes against each other, which is useful for visually assessing correlations between attributes, or which attributes may be most important in discriminating classes. The colour of each point represents its class label in the training set.

In this case the large number of attributes can be stifling. Select any square on the plot to bring up the alternative interface.

1. Using the *X* and *Y* select boxes at the top, try plotting various pairs of attributes against each other to get a feel for the data.
2. Are there any significant correlations?
3. Can you find a pair of attributes that seems particularly effective at discriminating between the classes? In such a case there should only be a little mixing of the colours. Make a note of these for later.  
   **Hint:** recalling the format of the data, concentrate on pixel 5, the centre pixel, which we might expect to be most telling given it contains the soil type we are classifying.

We'll now attempt to build a Naive Bayes classifier over the data. We will apply our visual observations in a moment.

Return to the *Classify* tab and select the Naive Bayes classifier again if it is not already selected. Let's use 5-fold cross-validation (default option is 10). Hit *Start* to run the cross-validation.

Observe the results:

* Note the percentage of correctly and incorrectly classified instances.

Now we'll apply the knowledge of our visual observations:

1. Return to the *Preprocess* tab.
2. Check the box next to the two attributes you identified as being most useful above.  
   **Cheat:** If you're not sure, try pixel5\_2 and pixel5\_4.
3. Check the box next to the class attribute.
4. Click the *Invert* button and then the *Remove* button. This should remove all attributes except for your two most important and the class label.
5. Return to the *Classify* tab and select the 5-fold cross-validation again.

**Attribute Selection via Information Gain**: Weka has a whole range of functionalities for attribute selection. We will use Information Gain, i.e the mutual information between two variables. In attribute selection we are interested in the mutual information between the class and each of the other attributes.

* Go to the 'Select attributes' tab sheet
* Choose the Information Gain attribute selection method from the Attribute Evaluator box.
* Choose the Ranker search method in the Search Method box; you can indicate the number of attributes to retain to be 10 or so.
* Click 'Start'. Does the ranking make sense based on your previous observations? Which attributes obtain the highest score?
* Would you logically expect pixel6\_1 and pixel6\_2 to be more important than pixel5\_4? Go back to visualisation and plot pixel6\_1 against pixel5\_1, or pixel6\_2 against pixel5\_2. Can you spot a problem with information gain attribute selection here?

Run the classifier again and note the percentage of correctly classified instances:

* How does this compare to the result using all attributes?
* What does this tell you about these two particular attributes?
* Why might this approach be useful?  
  **Hint:** consider larger, more complex problems.
* What distribution is used by Weka to model the attribute values?

When you perform attribute selection in the 'Select attributes' tab sheet, you only get a list of ranked attributes. You may want the attribute selection to have an effect on all tabs for you to work on the reduced data set. To achieve this, you have to perform attribute selection in the 'Preprocess' tab sheet.

* Select the supervised 'AttributeSelectionFilter'.
* Set its settings as before (InfoGain + Ranker, numToSelect: 10)
* Apply the filter.

You have now replaced the working dataset for the whole weka environment.

* Train a NaiveBayes classifier on the reduced data.
* Was the performance drop significant? Why?

**CPU PERFORMANCE**

In this section we use a CPU Performance data set. The task is to predict the Estimated Relative Performance (ERP) based on a number of attributes, namely vendor, MYCT, MMIN, MMAX, CACH, CHMIN, CHMAX. More information about this data set can be found [here](http://archive.ics.uci.edu/ml/datasets/Computer+Hardware).

1. Download the data set [cpu.arff](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/cpu.arff) and load it into Weka as usual
2. Notice that the vendor attribute is nominal, not numeric. This will give problems when using a linear regression model. For now we can simply remove this attribute. Select the check box for *vendor* and click *Remove*.
3. Now use the *Visualize* tab to explore the data. First look at plots of the input attributes against the target variable ERP (shown in the top row of scatterplots) and then look at plots of pairs of input attributes.
   * Do you think that ERP should be at least partially predictable from the input attributes?
   * Do any attributes exhibit significant correlations?
4. Now we have a feel for the data and we will try fitting a simple linear regression model to the data. On the *Classify* tab, select *Choose > functions > LinearRegression*. Use the default options and click *Start*. This will use 10-fold cross-validation to fit the linear regression model. Examine the results:
   * Record the Root relative squared error and the Relative absolute error. The Relative squared error is computed by dividing (normalizing) the sum of the squared prediction errors by the sum of the prediction errors obtained by always predicting the mean. The Root relative squared error is obtained by taking the square root of the Relative squared error. The Relative absolute error is similar to the Relative squared error, but uses absolute values rather than squares. See Table 5.8 in Witen and Frank (second edition). Therefore, if we have a relative error of 100%, the learned model is no better than this very dumb predictor.
5. Above we deleted the *vendor* variable. However, we can use nominal attributes in regression by converting them to numeric. The standard way of so doing is to replace the nominal variable with a bunch of binary variables of the form *"is\_first\_nominal\_value, is\_second\_nominal\_value"*and so on. Reload the unmodified data file [cpu.arff](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/cpu.arff). On the *Preprocess* tab select *Choose > filters > unsupervised > attribute > NominaltoBinary* and click *Apply*. This replaces the *vendor* variable with 30 binary variables and we now have 37 attributes (we started with 8).
6. Now train a linear regression model as in (4) and examine the results.
   * Record the Relative absolute error and the Root relative squared error.
   * Compare the performance to the one we had previously. Did adding the binarized *vendor* variable help?
7. So far, we have made use of Linear Regression. One could also try fitting nonlinear models. If you have time, experiment with a non-linear predictor that has been discussed in class - a kNN regression *IBk*. Do you get better performance with this non-linear predictor?