**IAML LAB 4: CLUSTERING, PCA AND EVALUATION**

In this final lab we consider unsupervised learning in the form of *clustering* methods and *principal component analysis* (PCA), as well as more thorough performance evaluation of classifiers.

**CLUSTERING**

We first consider clustering of the Landsat data. You will need the following dataset for this task:

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

To refresh your memory about the Landsat data you can read [this description](http://www.inf.ed.ac.uk/teaching/courses/dme/html/landsat.html). Since there are 6 classes in the data, it would be interesting to try clustering with *k=6* centres. Load the landsat data into Weka and go to the *Cluster* tab:

1. Select the *SimpleKMeans* clusterer, bring up its options window and set *numClusters* to 6.
2. In the *Cluster mode* panel, select *Classes to clusters evaluation* and hit *Start*
3. Ideally, we would hope to see all instances from a single class assigned to a single cluster, and no instances from different classes assigned to the same cluster. Look at the *Classes to Clusters* confusion matrix. Clearly, we don't have a perfect correspondence between classes and clusters, but:
   * How successful has the clustering been in this regard?
   * Looking at each class individually, can you spot particular classes that are well identified by the clustering? Classes that are poorly identified?
   * Which classes are mostly confused with each other?
   * Does this relate to the observations you made in Lab 2? Have a look at the *Visualization* tab again if you need to.
4. Visualize the cluster assignments. To do this, right-click on the clusterer in the *Result list* panel and select *Visualize cluster assignments*. Plot *Class* against *Cluster*. All the data points will lie on top of each other, so increase the *Jitter* slide bar to about half way to add random noise to each point. This allows us to see more clearly where the bulk of the datapoints lies. In this scatter plot each row represents a class and each column a cluster.

**PCA**

We now consider PCA. Instead of selecting a subset of attributes, PCA allows us to construct a new set of features based on a linear combination of the attributes. We will use the [Landsat](http://archive.ics.uci.edu/ml/datasets/Statlog+%28Landsat+Satellite%29) satellite imaging set. Acquaint yourself with this by reading [this description](http://www.inf.ed.ac.uk/teaching/courses/dme/html/landsat.html). Load up the [Landsat data](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/landsat.arff)

1. On the *Attribute Selection* tab, choose the *PrincipalComponents* attribute evaluator. Use the *Ranker* search method, but set the *numToSelect*option to -1. Hit *Start* and observe the output.
   * How many features does the algorithm select?
   * On what basis does it select these? Could we select fewer? More?   
     *Hint:* Bring up the *PrincipalComponents* options window and click on *More* to read a synopsis of the method's implementation
2. On the *Result list* (at the bottom-left corner of the *Attribute Selection* tab), right-click on the PCA run and select *Visualize transformed data*.
3. Look at the scatter plots of the PCA attributes against each other.
   * Can you make any salient observations?
   * Do you think that the classes are more separable with the PCA features than with the original set of attributes?
4. We can consider training a classifier with the PCA feature representation. Do you expect the resulting model to achieve better performance than when trained on the original set of features?
5. If you have time, go to the *Preprocess* tab and select *Choose > filters > supervised > attribute > AttributeSelection*. Bring up the *AttributeSelection* options window and set the *evaluator* field to *PrincipalComponents* and the *search* field to *Ranker*. Click *OK* and then *Apply*. This filter will replace the original features with the PCA features.
6. Train a Naive Bayes model using the PCA representation and 5-fold cross-validation. Then reload the [Landsat data](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/landsat.arff) and train a Naive Bayes model using the original representation (again use 5-fold CV). Compare the performance of the resulting models. Was this your guess in question 4?

**PERFORMANCE ASSESSMENT #2**

We will continue our performance assessment on the [Splice](http://archive.ics.uci.edu/ml/datasets/Molecular+Biology+(Splice-junction+Gene+Sequences)) data set which was originally introduced in [Lab 3](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/lab3.html). As a reminder: the classification task is to identify *intron* and *exon* boundaries on gene sequences. Read the description at the link above for a brief overview of how this works. The class attribute can take on 3 values: N, IE and EI. Now download the data sets below, converted into ARFF for you, and load the training set into Weka:

* [splice\_train.arff](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/splice_train.arff): training data
* [splice\_test.arff](http://www.inf.ed.ac.uk/teaching/courses/iaml/lab/data/splice_test.arff): test data

1. Set *KNN* to the value that provides the greatest percent of correctly classified instances in Lab 3, Question 5 and re-run the classifier on the held out test dataset *splice\_test.arff*.
2. We wish to understand the output from WEKA with respect to TP rate, FP rate, Precision, Recall etc. Note that these are output on a per-class basis. As the Splice data is a three-class problem, we can consider each of the classes (N, EI, IE) as the "positive" class, and lump the remaining two together as the negative class. Thus to compute TP rate etc we need to reduce the 3 x 3 confusion matrix to a 2 x 2 confusion matrix. So if the positive class is EI, this will mean lumping together the results for the N and IE classes.

Reduce the 3 x 3 confusion matrix to a 2 x 2 confusion matrix using EI as the positive class. To check your reasoning, compute the TPR = TP/(TP+FN), FPR = FP/(FP+TN) and Precision = TP/(TP+FP) and check them against the results output by WEKA. Is the proportion of points correctly classified a sufficient measure of performance? *Hint:* Consider the other data sets that we've looked at. What does a false positive mean in identifying spam? Is this as bad as a false negative?