COC131 Data Mining - Classification II

Martin D. Sykora m.d.sykora@lboro.ac.uk
Tutorial 07, Friday 24th April 2009

Validation

In tutorial 3 we touched upon classification and looked at outputs provided under "Summary", "Detailed accuracy by class" and "Confusion Matrix" sections of the "Classifier output" window. In this workshop we look at classifiers again and their validation. Holdout (Percentage split), Cross-validation are common techniques for assessing accuracy based on randomly sampled partitions of the given data. The use of such techniques to estimate accuracy increases the overall computation time, yet is usefull for model selection.

Cross-validation

Cross-validation, is the statistical practice of partitioning a sample of data into subsets such that the analysis is initially performed on a single subset, while other subset(s) are retained for subsequent use in testing and validating the initial analysis.

K-fold cross-validation

In K-fold cross-validation, the original sample is partitioned into K subsamples. Of the K sub-samples, a single sub-sample is retained as the testing data, and the remaining K-1 sub-samples are used as training data. The cross- validation process is then repeated K times (the folds), with each of the K sub-samples used exactly once as the validation data. The K results from the folds then can be averaged (or otherwise combined) to produce a single estimation.

Leave-one-out cross-validation

As the name suggests, leave-one-out cross-validation involves using a single observation from the original sample as the testing data, and the remaining

observations as the training data. This is repeated such that each obseration in the sample is used once as the testing data. This is the same as K-fold cross-validation where K is equal to the number of observations in the original sample.

Holdout validation

Holdout (Percentage split in WEKA) validation is not strictly cross-validation, because the data never are crossed over. Observations are chosen randomly from the initial sample to form the testing data, and the remaining observations are retained as the training data. Normally, less than a third of the initial sample is used for testing data.

Exercises

1. Fire up Weka (Waikako Environment for Knowledge Analysis) software, launch the explorer window and select the "Preprocess" tab. Open the iris training data-set ("iris-training.arff", get this from http://www-staff.lboro.ac.uk/ comds2/).

2. Select the "Classify" tab. Under the "Classifier" section select "MultilayerPerceptron" *(if you click on the textbox next to the choose button, the object editor will appear, clicking the "More" button brings up information about the classifier itself)*, leave the parameters as the defaults.

3. Run the model using each of the different "Test options" in turn.

   (a) For "Supplied test set" use "iris-testing.arff", get this from http://www-staff.lboro.ac.uk/ corlk/.

   (b) For "Cross-validation" try different "Folds" values e.g. 2, 3, 5, 10, 15, 30, 45, 90. What do these values mean? What is interesting about making "Folds" equal to 90?

   (c) For "Percentage split" try different "%" values e.g. 20, 40, 60, 80. What do these values mean?

   (d) How do each of the test options affect the performance of the model (look at the visualisations as well as the text output)? Which test option gives the best performance? Is it generally a good idea to use the same data-set to both test and train a model, i.e. option "Use training set" *(note: evaluating a model on training data might still be usefull, because it generally represents an upper bound to the model's performance on fresh data)*?

4. Try changing some of the classifier parameters *(read the "more" section for a brief explanation of the classifiers parameters)* and repeat exercise 3. How does changing the classifier parameters affect the performance?

5. Under the "Classifier" section change the classifier to "BayesNet", leave the parameters as the defaults.

6. Repeat exercises 3 and 4 for the new classifier.

7. Under the "Classifier" section change the classifier to "J48" *(this is an implementation of the C4.5 decision tree algorithm, an extension of the ID3 algorithm)*, leave the parameters as the defaults.

8. Run the classifier on the "supplied test set" option. Under "Results list" you should see your model, right click on it and select "Visualize tree", the tree has 5 leaves and 9 nodes altogether. On each of the four levels, what attributes does the tree split on?

9. How does altering the classifier affect performance? How do you determine which is the best classifier, testing/training strategy and parameter set for a particular classification problem?