Classifier Training and Evaluation

TYPES OF CLASSIFIERS

Bayes – Always the optimal (minimum error rate or minimum risk) but requires exact knowledge of class prior probabilities and class conditional probabilities of features. Seldom possible because exact knowledge rarely exists.

Bayes linear – Assumes Gaussian distribution of features with equal covariance matrices for each class. A modest number of parameters to estimate. Fast training and classifying. In general, performance is limited.

Bayes quadratic – Assumes Gaussian distribution of features with a separate covariance matrix for each class. Requires many parameters (feature covariances) to be estimated. Fast training and classifying. Performance may be poor when data is significantly non-Gaussian.

Nearest neighbor (1-nearest neighbor) – A simple nonparametric method that uses all the training data for classification. Has high computational complexity for classification, though some acceleration methods exist. Must select a metric. Upper bound on error rate approaches twice that of ideal Bayes classifier.

k\textsubscript{n}-Nearest neighbor – A robust non-parametric classifier. Classification has high computational complexity when. Must select metric and value of k. k must be set using validation. Can have excellent performance for arbitrary class conditional pdfs.

Parzen window – Robust non-parametric. Must select form of kernel and size parameter h. Complexity and performance is similar to k-NN method.

Neural network – The multi-layer perceptron (a non-parametric classifier) is the standard network to use for supervised learning. Other types of neural networks are useful for unsupervised learning. Training can be very slow, but classification is fast. The number of hidden nodes must be set using validation (see below). Can have excellent performance. Impossible for a human to “understand” the classifier. Performance is vulnerable to unforeseen input data.

Decision tree – non-metric method. Gives a set of rules that can be understood.
CLASSIFIER EVALUATION METHODS

Goals
Estimate the performance (error rate) of a classifier. The lower the error, the better.
Often used to compare two or more types of classifiers.
Compare the performance of two classifiers.

Assumptions
Classifiers are trained using real data, not simulated data.
There are a limited number of samples to work with (for both training and testing).

Methods:
1. Resubstitution
First uses all available data to design a classifier.
Then uses the same data again to test the classifier.
Produces an “optimal” classifier in the sense that it uses all available data for design.
Easy and fast.
Using the same data for training and testing gives an optimistically biased estimate of the error rate, but variance is relatively smaller since all data is used for training. It is useful for determining a lower bound on classifier error rate. Since it suffers from “testing on the training data,” resubstitution is not recommended unless the training set is very large.

2. Data Partition (Holdout)
First separate data into two groups for training and testing. Often the training set will be selected to be twice the size of the test set.
The classifier is sub-optimal in the sense that it uses only part of available data for training.
Testing also suffers from a small test sample.
Easy and fast.
Result estimate of error rate is unbiased, but has a large variance (i.e., uncertainty).
3. Cross-validation (see Wikipedia)
   A generalization of the holdout method.
   N total samples are divided into $m$ groups of equal size.
   $m$ different classifiers are trained each using $m - 1$ groups, holding out each of
   the groups.
   For each of the $m$ classifiers, the group left out is tested.
   The $m$ test results are averaged.
   All samples get used for both training and testing.
   The result is unbiased and with minimum variance.
   Good method when a large number of samples are available.
5x2 cross-validation: Randomly divide data set into two parts equal sized parts: $D_1$ and $D_2$. First train on $D_1$ and validate (test) on $D_2$, then reverse the roles. Repeat this process 4 more times for a total of 10 train and test runs. Five random divisions is chosen as a compromise between getting a large enough sample of results (the more, the better) and diminishing returns (since there is much redundancy in the train/validate sets). From Apaydin chapter 14.

A good method to use for selecting the appropriate classifier type to use and for determining certain classifier “super”-parameters, e.g., $k$ for nearest neighbor, $h$ for Parzen, number of hidden nodes in neural net.

4. Jackknife (Leave-one-out)
   A limiting case of cross-validation. Where $m = N$.
   N different classifiers are trained each using $N-1$ samples.
   For each of the $N$ classifiers, the one left out sample is tested.
   The $N$ test results are averaged.
   Classifiers are very close to optimal.
   All samples are used for testing.
   Result is unbiased and with minimum variance.
   If a fast leave-one-out algorithm is available (to estimate necessary parameters using an update scheme: e.g., mean, covariance and its inverse and determinant), jackknife is fairly fast.
   But, if no fast algorithm is available, then it is very slow, such as neural networks.
   Fast algorithms exist for estimating mean, covariance matrix, as well as inverse and determinant of covariance matrix. So, useful for:
Bayes quadratic, k-nearest neighbor (using Euclidean or Mahalanobis distance), Parzen with Gaussian window (see Fukunaga for the algorithms).

The best method for estimating performance.

5. Bootstrap

Various forms exist. A common one is:

A training set is generated by randomly selecting N samples using replacement (i.e., samples can be selected more than once).

The samples not selected for training are used for testing.

The process is repeated many times, e.g., 200.

The results are averaged to give final estimate of the error rate.

The classifiers produced are never optimal.

The resulting estimate is unbiased.

This method is very computationally intensive (slow).

The method very good for use when only a few samples are available, say less than roughly 30.

Notes on training, validation, and testing:

Some classifier methods need certain “super”-parameters, as mentioned above. In this case one should divide the training data into two groups, one for actually training the classifier and one for validation purposes. The results of the validation tests are used for selecting the needed super-parameters; this is still part of the overall training process. The test data should be separate and not used for selecting the super-parameters. It should be use only after all the parameters have been decided.

After all super-parameters of classifier are selected, then a classifier is designed using the all available training data. The resulting classifier can be tested on the test data to give an unbiased estimate of its error rate.

See also: http://en.wikipedia.org/wiki/Resampling_%28statistics%29