The Tree Classifiers

When both the number of classes c and the number of features n are large, the feature selection and classification discussed before encounter difficulties because

- feature selection is no longer effective as it is difficult to find m features from n which are suitable for separating all the c classes (some features may be good from some classes but not good for others).
- classification is costly as a large number of features are necessary.

The solution is to do classification in several steps implemented as a *tree classifier*. One method to design the tree classifier is the bottom-up merge algorithm described in the following steps, which is considered as the training process.

1. From the training samples of each class $\omega_i \quad (i = 1, \cdots, c)$, estimate the mean and covariance:
$$M_i = \frac{1}{N_i} \sum_{X \sim \omega_i} X$$
and
$$\Sigma_i = \frac{1}{N_i} \sum_{X \sim \omega_i} (X - M_i)(X - M_i)^T$$

2. Compute Bhattacharrya distances for every pair of different classes ($c(c-1)/2$ of them in total):
$$D_{ij} = \frac{1}{4}(M_i - M_j)^T \left[\frac{\Sigma_i + \Sigma_j}{2}\right]^{-1} (M_i - M_j) + log \left[\frac{\left|\frac{\Sigma_i + \Sigma_j}{2}\right|}{(|\Sigma_i| \ |\Sigma_j|)^{1/2}}\right]$$
$$for \ \ all \ \ i \neq j$$

3. Merge the two classes with the smallest D_{ij} to form a new class: $\omega_i \cup \omega_j = \omega_n$ and compute its mean and covariance:
$$M_n = \frac{1}{N_i + N_j}[N_i M_i + N_j M_j]$$

and

$$\Sigma_n = \frac{1}{N_i + N_j}[N_i(\Sigma_i + (M_i - M_n)(M_i - M_n)^T) + N_j(\Sigma_j + (M_j - M_n)(M_j - M_n)^T)]$$

Delete the old classes ω_i and ω_j.

4. Compute the distance between the new class ω_n and all other classes (excluding ω_i and ω_j).

5. Repeat the above steps until eventually all classes are merged into one and a binary tree structure is thus obtained.

6. At each node of the tree build a 2-class classifier to be used to classify a sample into one of the two children G_l and G_r representing the two groups of classes. According to the classification method used, we find the discriminant functions $D_l(X)$ and $D_r(X)$.

7. At each node of the tree adaptively select features that are best for separating the two groups of classes G_l and G_r. Any feature selection method can be used here, such as directly choosing m from n features using between-class distance (Mahananobis distance) as the criterion, or feature selection using some orthogonal transform (KLT, DFT, WHT, etc.). Only a small number of selected features may be needed as here only two groups of classes need to be distinguished.

After the classifier is built and trained, the classification is carried out in the following manner:

A testing sample X of unknown class enters the classifier at the root of the tree and is classified to either the left or the right child of the node according to

$$X \sim \begin{cases} G_l & if \quad D_l(X) > D_r(X) \\ G_r & if \quad D_l(X) < D_r(X) \end{cases}$$

This process is repeated recursively at the child node (either G_l or G_r) and its child and so on, until eventually X reaches a leaf node corresponding to a single class, to which the sample X is therefore classified.