1 Pattern Recognition

Pattern recognition is a technique for classification of samples based on a certain pattern. Nowadays there are many different models, i.e. classifiers available. They differ from each other by the shape of the boundary, complexity of the model, computational speed. However practice often shows that given the appropriate set of features, all classifiers tend to have a similar performance.

In a classical problem with n samples x, which consist of m features, the dataset of available samples is a matrix $\mathbf{x}_{[n\times m]}$, whereas the vector of labels that describe the belonging of each sample to one of the classes is $\mathbf{y}$, where $y \in (1, 2, 3, ..., K)$.

Bayesian equation laid the basic principle of the pattern recognition. According to the equation, the probability that a sample x_0 belongs to a class k is equivalent to the:

$$P(y = k \mid x = x_0) = \frac{P(x = x_0 \mid y = k)\ P(y = k)}{P(x = x_0)} \tag{1}$$

Term $P(x = x_0 \mid y = k)$ is called the *class-conditional probability* and describes the probability that the sample with exact features x_0 is encountered within the group of samples belonging only to the class C. Term $(P(y = C))$ is called the *a priori probability* and describes the probability that the sample with class C is found within the group of all samples, regardless of the features. Finally, the term $P(x = x_0)$ is called *marginal probability* and describes the probability of finding the exact set of features in the dataset, regardless of the class. Marginal probability can be writen as a sum of class-conditional probabilities multiplied by the a priori probabilities for each class:

$$\begin{aligned} P(x = x_0) = P(x = x_0 \mid y = 1)\ P(y = 1) + \\ P(x = x_0 \mid y = 2)\ P(y = 2) + \cdots + \\ P(x = x_0 \mid y = K)\ P(y = K) \end{aligned} \tag{2}$$

For classification task, the hypothesis, i.e., the predicted class of a sample x_0 is chosen as the class which has the highest probability $P(y = k \mid x = x_0)$:

$$h(x_0) = \underset{k}{\operatorname{argmax}}\, P(y = k \mid x = x_0) \tag{3}$$

Statistically speaking, this is the best possible classifier. The problem arises in implementation. The exact probability density functions are unknown and have to be estimate from the available data. Estimated version of the stated probability will be marked with a different symbols to stress out the fact they are just an estimates:

$$p_k(x) \coloneqq P(y = k \mid x) \tag{4}$$

$$g_k(x) \coloneqq P(x \mid y = k) \tag{5}$$

$$\pi_k \coloneqq P(y = k) \tag{6}$$

1.1 Linear Discriminant Analysis

All models are wrong; some models are useful.
George E. P. Box

Linear Discriminant Analysis is a computationally simple and efficient classifier with linear decision boundary. It is based on the Bayesian equation and estimates class-conditial probability from the available dataset as a Gaussian function:

$$g_k(x) = \frac{1}{(2\pi)^{1/2}\,|\Sigma_k|^{1/2}}\; e^{-1/2(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)} \tag{7}$$

, where μ_k and Σ_k are the mean and covariance matrix for class k, respectively, and they are estimated from the available data as:

$$m_k = \frac{1}{n_k}\sum_i x_i \Bigg|_{\forall x \in k} \tag{8}$$

$$\Sigma_k = \frac{1}{n_k - K}\sum_i \Big(x_i - \mu_k\Big)\Big(x_i - \mu_k\Big)^T \Bigg|_{\forall x \in k} \tag{9}$$

To achieve linearity, LDA assumes that the covariance matrices Σ are the same for all classes:

$$\Sigma_0 = \Sigma_1 = \cdots = \Sigma_K = \Sigma \tag{10}$$

and they are usually calculated using weighted average:

$$\Sigma = \frac{\sum_{k=1}^K n_k \Sigma_k}{\sum_{k=1}^K n_k} \tag{11}$$

where n_k represents the number of samples belonging to a class k.

In a two class example ($y \in \{0, 1\}$), all samples on the decision boundary ($D.B.$) will have the same probability of belonging to class 0 or 1:

$$D.B. = \Big\{ x \;\Big|\; P\big(y = 0 \,\big|\, x = x_0\big) = P\big(y = 1 \,\big|\, x = x_0\big) \Big\} \tag{12}$$

Following this idea, the decision boundary can be estimated by solving the equation:

$$\frac{g_0(x)\,\pi_0}{\cancel{\sum_{k=1}^K g_k\,\pi_k}} = \frac{g_1(x)\,\pi_1}{\cancel{\sum_{k=1}^K g_k\,\pi_k}} \tag{13}$$

$$\begin{aligned} \frac{1}{(2\pi)^{d/2}\,|\Sigma_0|^{1/2}}\; e^{-1/2(x-\mu_0)^T\Sigma_0^{-1}(x-\mu_0)}\;\; \pi_0 = \\ \frac{1}{(2\pi)^{d/2}\,|\Sigma_1|^{1/2}}\; e^{-1/2(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)}\;\; \pi_1 \end{aligned} \tag{14}$$

If making the assumption on the equal covariance matrices for both classes:

$$\Sigma_0 = \Sigma_1 = \Sigma \tag{15}$$

and taking the logarithm, the equation takes the form:

$$\begin{aligned} -\frac{1}{2}\Big(x-\mu_0\Big)^T\Sigma^{-1}\Big(x-\mu_0\Big) + \log\Big(\pi_0\Big) = \\ -\frac{1}{2}\Big(x-\mu_1\Big)^T\Sigma^{-1}\Big(x-\mu_1\Big) + \log\Big(\pi_1)\Big) \end{aligned} \tag{16}$$

This equation can be written as the linear function $x^T\beta + \alpha = 0$ as:

$$x^T\Big(\Sigma^{-1}\mu_0 - \Sigma^{-1}\mu_1\Big) + \frac{1}{2}\Big(\mu_1^T\Sigma^{-1}\mu_1 - \mu_0^T\Sigma^{-1}\mu_0\Big) + \log\Big(\frac{\pi_0}{\pi_1}\Big) \tag{17}$$

The equation represents the decision boundary between classes, i.e., all samples lying on this line will have equal probability of belonging to class 0 and class 1. It is interesting to note that the slope of the line depends only on the class means and covariance matrix, whereas a priori probabilities (which are the result of number of samples belonging to class 0 or 1) have effect only on the y-intercept term, i.e., the offset of the function.

When considering multiclass classification problem, probability of a sample belonging to each class is firstly estimated by the equation:

$$p_k = -\frac{1}{2}\log\big|\Sigma\big| - \frac{1}{2}\Big(x-\mu_k\Big)^T\Sigma^{-1}\Big(x-\mu_k\Big) + \log\Big(\pi_k\Big) \tag{18}$$

and then the class is estimated as the one with the highest probability as:

$$h(x) = \underset{k}{\operatorname{argmax}}\, p_k(x) \tag{19}$$

1.2 Support Vector Machine

When solving a problem of interest, do not solve a more general problem as an intermediate step.
Vladimir Vapnik

Supoort vector machine is nowadays known as a very powerful classifier with a lot of different applications. The big advantage over LDA is the fact that it is a *non-paramteric* classifier. This implies that the model is not obtained

using assumption of the form of the class density function and estimation of it's parameters, which is inevitably erraneous. Instead, SVM forms the decision boundary between samples by maximizing the distance between samples and the boundary. As Vladimir Vapnik said "When solving a problem of interest, do not solve a more general problem as an intermediate step."

This classification method esentially has a very simple principle, but using sever mathemathical trics, it became a very powerful tool. The optimization problem does not depend on x, but on $x^T x$. This enables the use of *kernel trick* and allows nonlinear transform of the feature space at no additional cost.