Introduction to Naive Bayes Classification

Naive Bayes is a simple, yet effective and commonly-used, machine learning classifier. It is a probabilistic classifier that makes classifications using the Maximum A Posteriori decision rule in a Bayesian setting. It can also be represented using a very simple Bayesian network. Naive Bayes classifiers have been especially popular for text classification, and are a traditional solution for problems such as spam detection.

The Model

The goal of any probabilistic classifier is, with features x_0 through x_n and classes c_0 through c_k, to determine the probability of the features occurring in each class, and to return the most likely class. Therefore, for each class, we want to be able to calculate P(c_i | x_0, ..., x_n).

In order to do this, we use **Bayes rule**. Recall that Bayes rule is the following:

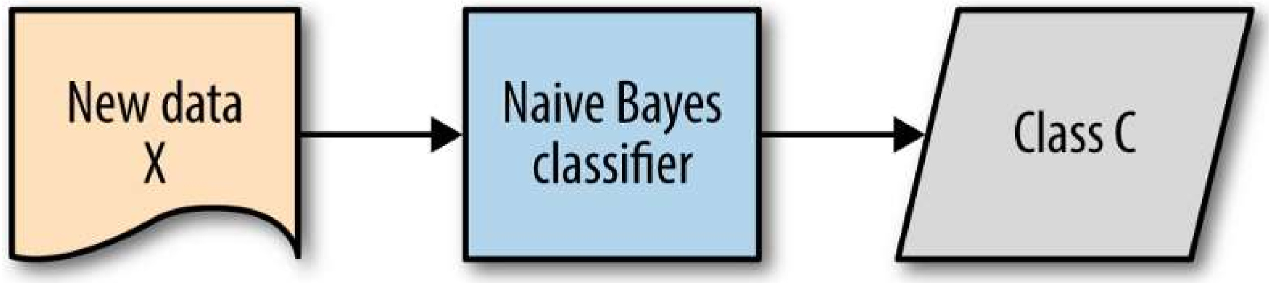
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

In the context of classification, you can replace A with a class, c_i, and B with our set of features, x_0 through x_n. Since P(B) serves as normalization, and we are usually unable to calculate P(x_0, ..., x_n), we can simply ignore that term, and instead just state that P(c_i | x_0, ..., x_n) ∝ P(x_0, ..., x_n | c_i) * P(c_i), where ∝ means "is proportional to". P(c_i) is simple to calculate; it is just the proportion of the data-set that falls in class i. P(x_0, ..., x_n | c_i) is more difficult to compute. In order to simplify its computation, we make the assumption that x_0 through x_n are **conditionally independent** given c_i, which allows us to say that P(x_0, ..., x_n | c_i) = P(x_0 | c_i) * P(x_1 | c_i) * ... * P(x_n | c_i). This assumption is most likely not true—hence the name *naive* Bayes classifier, but the classifier nonetheless performs well in most situations. Therefore, our final representation of class probability is the following:

$$\begin{aligned} P(c_i|x_0,\ldots,x_n) &\propto P(x_0,\ldots,x_n|c_i)P(c_i) \\ &\propto P(c_i)\prod_{j=1}^{n} P(x_j|c_i) \end{aligned}$$

Calculating the individual P(x_j | c_i) terms will depend on what distribution your features follow. In the context of text classification, where features may be word counts, features may follow a **multinomial distribution**. In other cases, where features are continuous, they may follow a **Gaussian distribution**.

Note that there is very little explicit training in Naive Bayes compared to other common classification methods. The only work that must be done before prediction is finding the parameters for the features' individual probability distributions, which can typically be done quickly and deterministically. This means that Naive Bayes classifiers can perform well even with high-dimensional data points and/or a large number of data points.



Classification

Now that we have a way to estimate the probability of a given data point falling in a certain class, we need to be able to use this to produce classifications. Naive Bayes handles this in a very simple manner; simply pick the c_i that has the largest probability given the data point's features.

$$y = \underset{c_i}{argmax}\ P(c_i) \prod_{j=1}^{n} P(x_j|c_i)$$

This is referred to as the **Maximum A Posteriori** decision rule. This is because, referring back to our formulation of Bayes rule, we only use the P(B|A) and P(A) terms, which are the likelihood and prior terms, respectively. If we only used P(B|A), the likelihood, we would be using a **Maximum Likelihood** decision rule.