Naive Bayes

Naive Bayes is another generative learning algorithm. The first step in naive Bayes is to convert an input into a feature vector x. Once we do this, we want to model $P(x \mid y)$ as well as $P(y)$. Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by an n-dimensional feature vector $\vec{x} = (x_1, \cdots, x_n)$, it assigns some instance probability to a set of outcomes. For k possible outcomes, there is a class y_k. In a binary classifier, this would be $k = \{0, 1\}$. The probability of a sample existing in some class y_k given the feature vector is $P\left(y_k \mid \vec{x}\right)$ and the class prior is $P(y_k)$. Using Bayes' rule, we derive that

$$P(y_k \mid \vec{x}) = \frac{P(y_k)P(\vec{x} \mid y_k)}{P(\vec{x})}$$

The denominator does not contain y_k, which is what we are interested in; thus, it is effectively constant. The numerator can be represented as the joint probability $P(\vec{x}, y_k)$. Using the chain rule for conditional probability, this expands into

$$P(\vec{x}, y_k) = P(x_1 \mid x_2, \cdots, x_n, y_k) \cdots P(x_n \mid y_k)P(y_k)$$

However, the features of $\vec{x}$ are assumed to be mutually independent. Under this assumption,

$$P(x_i \mid x_{i+1}, \cdots, x_n, y_k) = P(x_i \mid y_k)$$

and therefore the joint model can be simplified to

$$P(y_k \mid \vec{x}) \propto P(y_k) \prod_{i=1}^{n} P(x_i \mid y_k)$$

The total probability of the features $P(\vec{x})$ is the same as the sum of the probabilities for all outcomes k. We can write this as

$$P(\vec{x}) = \sum_k P(y_k)P(\vec{x} \mid y_k)$$

and so

$$P(y_k \mid \vec{x}) = \frac{P(y_k) \prod_{i=1}^{n} P(x_i \mid y_k)}{\sum_k P(y_k)P(\vec{x} \mid y_k)}$$

To construct a classifier from this, we would use the arg max on the probability model as such

$$y = \arg\max_y P(y_k) \prod_{i=1}^{n} P(x_i \mid y_k)$$