Advantages of Naive Bayes Algorithms

The advantages of the naive Bayes algorithms are enlisted as follows [4]:

- These algorithms work well for real-world applications despite the oversimplified assumptions.
- A small amount of training data is sufficient for estimating the parameters.
- These algorithms are very fast in comparison to more sophisticated methods.
- Each feature distribution can be independently estimated as a 1D distribution, which avoids dimensionality.

However, a disadvantage of the Naive Bayes algorithm is that it is a lousy estimator because it is based on the assumption that the features are independent.

3.4.2.5 Gaussian Naive Bayes

This is a special type of naive Bayes algorithm. The Gaussian Naive Bayes algorithm assumes that the continuous features associated with each class are distributed in a normal or a Gaussian distribution. The dataset is first distributed into classes, and then the mean and variance of each class are determined. Then the probability density of x_i of class y is given by the following equation:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right). \tag{3.22}$$

The mean of the values is μ_y and the Bessel corrected variance of the values is σ_y^2. Here, the maximum likelihood is used to estimate the parameters σ_y and μ_y.

3.4.2.6 Decision Tree Classification

As described in Sect. 3.4.1.7, a decision tree algorithm repeatedly splits the dataset until a pure leaf node is obtained in each part. For classification problems, different algorithms of decision tree construction use different attribute selection measures (ASMs). In this section, we will talk about five ASMs.

Entropy

Entropy is the measure of randomness or disorder in the dataset. The formula to calculate the entropy is

$$Entropy(A) = -P_{yes}\log_2(P_{yes}) - P_{no}\log_2(P_{no}), \tag{3.23}$$

where P_{yes} is the probability of the attribute being yes, and P_{no} is the probability of the attribute being no.

The entropy of an attribute can be measured using the following equation:

$$E(T, X) = \sum P(X)E(X), \tag{3.24}$$

where X is the attribute whose entropy we want to calculate, T is the target feature, $P(X)$ is the probability of the attribute, and $E(X)$ is the entropy of the attribute.

Information Gain

The information gain is used to measure how much information can be obtained from a certain feature based on its entropy. For instance, the ID3 algorithm uses entropy and information gain as ASM. To calculate the information gain, the following equation is used:

$$Information\ Gain(T,\ X) = Entropy(T) - Entropy(T, X), \tag{3.25}$$

where T is the feature variable and X is the attribute.

Split Information

The formula to calculate split information is given below:

$$Split\ Info_A(D) = -\sum \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right), \tag{3.26}$$

where A is the attribute, D_j is the frequency of attribute values, and D is the total number of attributes.

Gain Ratio

If the information gain of an attribute is divided by the split information, then we get the gain ratio of that attribute. The C4.5 algorithm uses the gain ratio and split information as the ASM. The gain ratio is calculated using the following equation:

$$Gain\ Ratio(A) = \frac{Information\ Gain(A)}{Split\ Info\ (A)}, \tag{3.27}$$

where A is the attribute.

Gini Index

The *Gini index* is simply the measure of impurity in the dataset. The Classification And Regression Tree (CART) algorithm uses the Gini index. The formula to calculate the Gini index is given below:

$$Gini = 1 - \sum (P_i)^2, \tag{3.28}$$

where P_i is the probability of the ith attribute.

Example 3.4 A dataset for playing golf is given in Table 3.6. Construct a decision tree using entropy and information gain as the attribute selection measures.

Following the data-fitting models and regression in the previous chapter, we now introduce logistic regression and other models for data analysis.

5.1 Logistic regression

In the previous analysis, all the dependent variable values y_i are continuous. In some applications such as biometrics and classifications, the dependent variable is just discrete or simply a binary categorical variable, taking two values 1 (yes) and 0 (no). In this case, a more appropriate regression tool is the logistic regression developed by David Cox in 1958. The logistic regression is a form of supervised learning [1,3,38].

Before we introduce the formulation of logistic regression, let us define two functions: the logistic function S and logit function. A logistic function (see Fig. 5.1), also called the sigmoid function, is defined as

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}, \quad x \in \mathbb{R}, \tag{5.1}$$

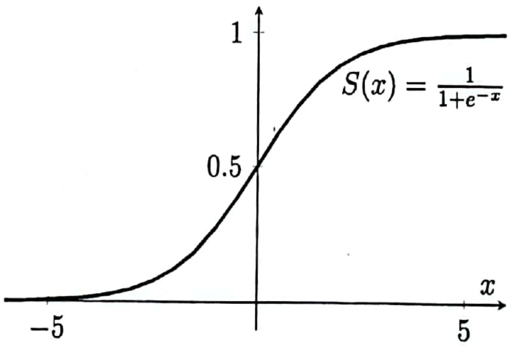

Figure 5.1 Logistic regression and its function.

$$x = \ln \frac{}{1-S}, \tag{5.7}$$

which is the well-known logit function in probability and statistics. In fact, the logit function can be defined as

$$\text{logit}\,(P) = \log \frac{P}{1-P} = \ln \frac{P}{1-P}, \tag{5.8}$$

which is valid for $0 < P < 1$.

The simple logistic regression with one independent variable x and a binary dependent variable $y \in \{0, 1\}$ with data points (x_i, y_i) $(i = 1, 2, \ldots, n)$ tries to fit a model of logistic probability

$$P = \frac{1}{1+e^{a+bx}}, \tag{5.9}$$

which can be written as

$$S(x) = \frac{1}{2}\Big[1 + \tanh\frac{x}{2}\Big], \quad \tanh x = \frac{e^x - x^{-x}}{e^x + e^{-x}}. \tag{5.2}$$

It is easy to see that $S \to +1$ as $x \to +\infty$, whereas $S \to 0$ as $x \to -\infty$. Thus the range of S is $(0, 1)$.

This function has an interesting property for differentiation. From the differentiation rules we have

$$\begin{aligned} S'(x) &= \Big[\frac{1}{1+e^{-x}}\Big]' = \frac{-1}{(1+e^{-x})^2}(-e^{-x}) = \frac{(1+e^{-x})-1}{(1+e^{-x})^2} \\ &= \frac{1}{(1+e^{-x})} - \frac{1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}}\Big[1 - \frac{1}{(1+e^{-x})}\Big] \\ &= S(x)[1 - S(x)], \end{aligned} \tag{5.3}$$

which means that its first derivative can be obtained by multiplication. This property can be very useful for finding the weights of artificial neural networks and machine learning to be introduced in Chapter 8.

To get the inverse of the logistic function, we can rewrite (5.1) as

$$S(1 + e^{-x}) = S + Se^{-x} = 1, \tag{5.4}$$

which gives

$$e^{-x} = \frac{1-S}{S}, \tag{5.5}$$

or

$$e^{x} = \frac{S}{1-S}. \tag{5.6}$$

Taking the natural logarithm, we have

$$x = \ln\frac{S}{1-S}, \tag{5.7}$$

which is the well-known logit function in probability and statistics. In fact, the logit function can be defined as

$$\text{logit}\,(P) = \log\frac{P}{1-P} = \ln\frac{P}{1-P}, \tag{5.8}$$

which is valid for $0 < P < 1$.

The simple logistic regression with one independent variable x and a binary dependent variable $y \in \{0, 1\}$ with data points (x_i, y_i) $(i = 1, 2, \ldots, n)$ tries to fit a model of logistic probability

$$P = \frac{1}{1 + e^{a+bx}}, \tag{5.9}$$

which can be written by using the logit function as

$$\ln \frac{P}{1-P} = a + bx, \tag{5.10}$$

and thus it becomes a linear model in terms of the logit of probability P. In fact, the odds can be calculated from probability by

$$O_d(\text{odd}) = \frac{P}{1-P} \tag{5.11}$$

or

$$P = \frac{O_d}{1+O_d}, \tag{5.12}$$

which means that the logistic regression can be considered as a linear model of log(odds) to x.

One naive way to solve the regression model (5.9) is to convert it to a nonlinear least squares, and we have

$$\text{minimize} \sum_{i=1}^{n} \Big[y_i - \frac{1}{1+e^{a+bx_i}}\Big]^2, \tag{5.13}$$

so as to find the optimal a and b. This is equivalent to fitting the logistic model to the data directly so as to minimize the overall fitting errors. This can give a solution to the parameters, but this is not the true logistic regression.

However, a more rigorous mathematical model exists for the binary outcomes y_i and the objective is to maximize the log-likelihood of the model with the right parameters to explain the data. Thus, for a given data set (x_i, y_i) with binary values of $y_i \in \{0, 1\}$, the proper binary logistic regression is to maximize the log-likelihood function, that is,

$$\text{maximize} \;\; \log(L) = \sum_{i=1}^{n} \Big[y_i \ln P_i + (1-y_i)\ln(1-P_i)\Big], \tag{5.14}$$

where

$$P_i = \frac{1}{1+e^{a+bx_i}} \quad (i = 1, 2, \ldots, n). \tag{5.15}$$

This is based on the theory of the maximum likelihood probability. Since $y_i = 1$ (yes or true) or 0 (no or false), the random variable Y for generating y_i should obey a Bernoulli distribution for probability P_i, that is,

$$B_P(Y = y_i) = P_i^{y_i}(1-P_i)^{1-y_i}, \tag{5.16}$$

so the joint probability of all data gives the likelihood function

$$L = \prod_{i=1}^{n} P(x_i)^{y_i}(1 - P(x_i))^{1-y_i}, \tag{5.17}$$

whose logarithm is given in (5.14). The maximization of L is equivalent to the maximization of $\log L$. Therefore, the binary logistic regression is to fit the data so that the log-likelihood is maximized.

In principle, we can solve the optimization problem (5.14) by Newton's method or any other optimization techniques. Let us use an example to explain the procedure in detail.

Example 22

To fit a binary logistic regression using

$$x: 0.1, \quad 0.5, \quad 1.0, \quad 1.5, \quad 2.0, \quad 2.5,$$
$$y: 0, \quad 0, \quad 1, \quad 1, \quad 1, \quad 0,$$

we can use the following form:

$$P_i = \frac{1}{1 + \exp(a + bx_i)} \quad (i = 1, 2, \ldots, 6), \tag{5.18}$$

starting with initial values $a = 1$ and $b = 1$.

Then we can calculate P_i with $a = 1$ and $b = 1$, and we have

$$P_i = \begin{pmatrix} 0.2497 & 0.1824 & 0.1192 & 0.0759 & 0.0474 & 0.0293 \end{pmatrix}.$$

The log-likelihood for each datapoint can be calculated by

$$L_i = y_i \ln P_i + (1 - y_i)\ln(1 - P_i),$$

and we have

$$L_i = \begin{pmatrix} -0.2873 & -0.2014 & -2.1269 & -2.5789 & -3.0486 & -0.0298 \end{pmatrix}$$

with the log-likelihood objective

$$\sum_{i=1}^{6} L_i = -8.2729.$$

If we try to modify the values of a and b by Newton's method, then after about 20 iterations, we should have

$$a = 0.8982, \quad b = -0.7099, \quad L_{\max} = -3.9162.$$

This means that the logistic regression model is

$$P = \frac{1}{1 + \exp(0.8982 - 0.7099x)}.$$

This logistic regression has only one independent variable. In the case of multiple independent variables $\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_m$, we can extend the model as

$$y = \frac{1}{1 + e^{w_0 + w_1\tilde{x}_1 + w_2\tilde{x}_2 + \ldots + w_m\tilde{x}_m}}. \tag{5.19}$$

Here we use $\tilde{x}$ to highlight its variations. To write them compactly, let us define

$$\tilde{\boldsymbol{x}} = [1,\ \tilde{x}_1,\ \tilde{x}_2,\ \ldots, \tilde{x}_m]^T \tag{5.20}$$

and

$$\boldsymbol{w} = [w_0,\ w_1,\ w_2,\ \ldots,\ w_m]^T, \tag{5.21}$$

where we have used 1 as a variable $\tilde{x}_0$ so as to eliminate the need to write w_0 everywhere in the formulas. Thus the logistic model becomes

$$P = \frac{1}{1 + \exp(\boldsymbol{w}^T\tilde{\boldsymbol{x}})}, \tag{5.22}$$

which is equivalent to

$$\text{logit} P = \ln\frac{P}{1-P} = \boldsymbol{w}^T\tilde{\boldsymbol{x}}. \tag{5.23}$$

For all the data points $\tilde{\boldsymbol{x}}_i = [1, \tilde{x}_1^{(i)}, \ldots, \tilde{x}_m^{(i)}]$ with $y_i \in \{0, 1\}$ $(i = 1, 2, \ldots, n)$, we have

$$\text{maximize } \log(L) = \sum_{i=1}^{n}\Big[y_i \ln P_i + (1 - y_i)\ln(1 - P_i)\Big], \tag{5.24}$$

where $P_i = 1/[+\exp(\boldsymbol{w}^T\tilde{\boldsymbol{x}}_i)]$. The solution procedure is the same as before and can be obtained by any appropriate optimization algorithm.

Obviously, the binary logistic regression can be extended to the case with multiple categories, that is, y_i can take $K \geq 2$ different values. In this case, we have to deal with the so-called multinomial logistic regression.

Though logistic regression can work well in many applications, it does have serious limitations [120]. Obviously, it can only work for discrete dependent variables, whereas a correct identification of independent variables is a key for the model. It usually requires a sufficiently large sample size, and the sample points should be independent of each other, so that repeated observations may cause problems. In addition, it can also vulnerable to overfitting.