第5章

分类算法

分类是监督学习的核心问题之一。在分类问题中，数据的输入属性既可以是连续的，也可以是离散的，而数据的输出属性是离散的。一般来说，在分类问题中，给定一系列数据，我们从数据中学习得到一个分类模型，该模型可以对新的输入进行输出的预测。当分类的类别是两个时，为二分类问题；当分类的类别大于两个时，为多分类问题。举几个例子说明，在邮件分类问题中，我们需要将收到的邮件进行分析，可以分成垃圾邮件和非垃圾邮件。因此，邮件分类是一个典型的二分类问题。多分类问题的例子则更为多样。比如，在天气预报中，我们可以将本地的天气数据样本分成晴天、下雨、下雪、阴天等。所以，天气数据的分类是一个多分类问题。

需要指出的是，在二分类问题中，预测输出变量 y 只有两个值，一般情况下，我们用 0 来表示负类（注：有些书也用 -1 表示负类），用 $+1$ 表示正类。对于这种二分类问题，我们只需要构建一个分类器即可。

对于多分类问题，输出预测变量则有多个值，即 $y\in\{0,1,2,3,\cdots\}$ 等。在现实生活中，我们遇到更多的是多分类问题。考虑到现有的分类方法大多都是基于二分类问题构建的，因此，我们在面对多分类问题时，更多的是将多分类问题转换成二分类问题进行解决。总体来说，有 3 种策略。考虑到纠错输出编码（Error Correcting Output Codes，ECOC）方法比较复杂，在本书中，只介绍以下两种策略。

1. “一对一”策略（One vs One）

假定某个多分类问题中有 n 个类别，我们将这 n 个类别进行两两配对，就可以将该多分类问题转换成 $\binom{n}{2}$ 个二分类问题。组合完成之后，我们选择其中的一个类别作为正类，另一个类别作为负类。分别对每一个二分类器进行训练，从而可以得到 $\binom{n}{2}$ 个二分类器，然后把测试的样本在所有的二分类器上进行预测，并将预测结果通过投票的方式来确定多分类器上的类别。

以图 5-1 所示为例，我们利用“一对一”策略将一个四分类问题转换成 6 个二分类问题。在得到 6 个二分类器之后，对测试样本进行预测，从最后的结果来看，类别 1 被二分类器预测的最多，因此测试样本属于类别 1。

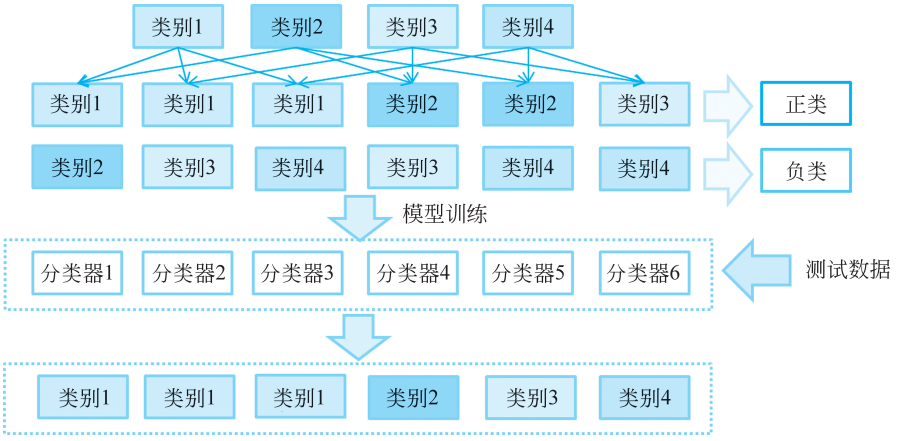

图 5-1　“一对一”策略

2. “一对剩余”策略(One vs Rest)

相较于“一对一”策略,“一对剩余”策略更容易理解,每次将一个类别作为正类,剩余的所有类别作为负类。此时,我们可以构建总共 n 个二分类器。在对测试数据进行分类的时候,如果有一个分类器被预测为正类,则对应的标签就可以作为最终的分类结果。

以图 5-2 所示为例,利用“一对剩余”策略可以得到 4 个二分类器,根据二分类器的预测结果,可以判定测试数据属于类别 2。

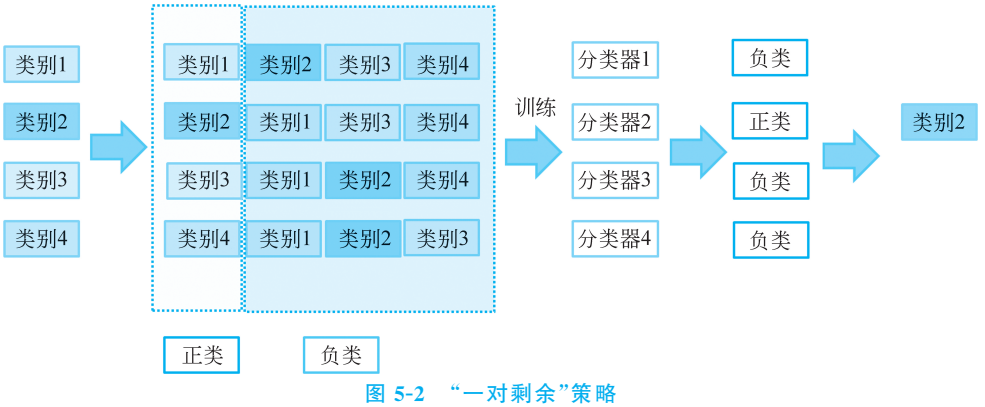

图 5-2　“一对剩余”策略

由于可以基于上述策略进行问题转换,因此,本章只介绍一系列的二分类算法。

5.1　逻辑回归

逻辑回归(Logistic Regression, LR)是一个经典的统计模型,最早是由统计学家 David Cox 在 20 世纪 50 年代提出,目前仍然广泛应用于二分类问题。以下详细介绍逻辑回归的思想。

给定一个由 d 个属性组成的样本 $\boldsymbol{x}=(x_1,x_2,\cdots,x_d)^{\mathrm{T}}$,其中 x_i 是第 i 个属性值。此时,通过对数据进行训练可得一个通过属性线性组合来进行预测的线性模型,即

$$z=w_1x_1+w_2x_2+\cdots+w_dx_d+b \tag{5-1}$$

若改用向量的形式,可以将式(5-1)改写为

$$z = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \tag{5-2}$$

其中,模型参数 $\boldsymbol{w}=(w_1,w_2,\cdots,w_d)^{\mathrm{T}}$ 和 b 可以学习得到。

对于一个二分类任务,它的输出标记为 $y\in\{0,1\}$,而线性回归模型产生的预测值是连续值,为了构建一个二分类器,需要将输出值 z 转换成离散的{0,1}值。最简单的转换方式是阶跃函数,即

$$y=\begin{cases}0, & z<0\\ 0.5, & z=0\\ 1, & z>0\end{cases} \tag{5-3}$$

其中,如果预测值 z 大于 0,分类器判为正类;如果预测值 z 小于 0,分类器判为负类;若预测值 z 为 0,分类器可以判为任意类别。

考虑到阶跃函数不是连续的,因此无法对其直接进行求导。为了解决上述问题,我们试图找到在一定程度上近似于阶跃函数的替代值。逻辑函数就是这样一个函数,即

$$y=\frac{1}{1+\mathrm{e}^{-z}} \tag{5-4}$$

显然,Sigmoid 函数是一个可用的替代方案,如图 5-3 所示,它的输出值在 0 附近变化很陡,利用 Sigmoid 函数将 z 值转换成一个接近于 0 或 1 的 y 值。将线性判别模型式(5-2)代入式(5-4),可得

$$y=\frac{1}{1+\mathrm{e}^{-(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b)}} \tag{5-5}$$

两边取对数,再经过转换,可得

$$\log\frac{y}{1-y}=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b \tag{5-6}$$

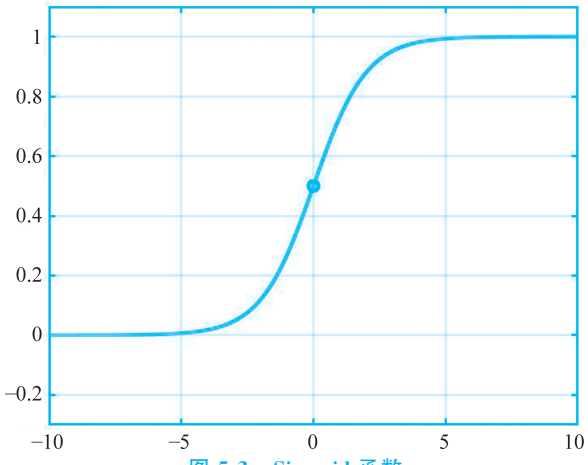

图 5-3 Sigmoid 函数

换个角度看式(5-6),可以将 y 视作 $\boldsymbol{x}$ 隶属于正类的概率,则 $1-y$ 视作 $\boldsymbol{x}$ 隶属于负类的概率。由此可知,$\frac{y}{1-y}$ 是 $\boldsymbol{x}$ 正负类概率的比值,有些参考书也称其为“几率”。

在逻辑回归中,利用一个线性分类模型的预测结果去估计标签“几率”的对数值。所以,

有些参考书也把上述方法称为“对数几率回归”。但这里需要额外指出的是，虽然逻辑回归中有“回归”二字，但是它本质上是一个分类方法。

这种方法具备诸多优点。首先，它无须事先对数据分布进行评估，这就可以在最大限度上避免分布估计不准确带来的风险。其次，逻辑回归利用的是软标签，可以将其视作概率的预测，这对于许多利用概率辅助决策会有帮助。最后，逻辑回归是一个凸函数，因此容易求导，许多的数值优化方法都可以用于求解最优值。

下面我们利用相关凸优化理论去求解 $\boldsymbol{w}$ 和 b，如果将式(5-6)中的 y 看作后验概率 $p(y=1|\boldsymbol{x})$的估计值，则我们可以将式(5-6)改写为

$$\log\frac{p(y=1\mid\boldsymbol{x})}{p(y=0\mid\boldsymbol{x})}=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b \tag{5-7}$$

显然，我们可以有

$$p(y=1\mid\boldsymbol{x})=\frac{\mathrm{e}^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}{1+\mathrm{e}^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}} \tag{5-8}$$

$$p(y=0\mid\boldsymbol{x})=\frac{1}{1+\mathrm{e}^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}} \tag{5-9}$$

通过最大似然法来对 $\boldsymbol{w}$ 和 b 进行估计。给定规模为 m 的数据集$\{(\boldsymbol{x}_i,y_i)\}_{i=1}^{m}$，我们可得对数似然函数如下：

$$l(\boldsymbol{w},b)=\sum_{i=1}^{m}\log p(y_i\mid\boldsymbol{x}_i;\boldsymbol{w},b) \tag{5-10}$$

观察上式，可知在似然函数中，各个样本属于其真实标签的概率越大则分类性能越好。为简单起见，我们可以令 $\boldsymbol{\beta}=[\boldsymbol{w}^{\mathrm{T}},b]^{\mathrm{T}}$ 和 $\widetilde{\boldsymbol{x}}=[\boldsymbol{x}^{\mathrm{T}};1]^{\mathrm{T}}$。此时，我们可将 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$ 改写为 $\boldsymbol{\beta}^{\mathrm{T}}\widetilde{\boldsymbol{x}}$。同理，我们可以令 $p_1(\widetilde{\boldsymbol{x}};\boldsymbol{\beta})=p(y=1|\widetilde{\boldsymbol{x}};\boldsymbol{\beta})$，$p_0(\widetilde{\boldsymbol{x}};\boldsymbol{\beta})=p(y=0|\widetilde{\boldsymbol{x}};\boldsymbol{\beta})=1-p_1(\widetilde{\boldsymbol{x}};\boldsymbol{\beta})$。基于此，我们可以将似然项改写为

$$p(y_i\mid\boldsymbol{x}_i;\boldsymbol{w},b)=y_ip_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})+(1-y_i)p_0(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta}) \tag{5-11}$$

将式(5-11)代入式(5-10)，可将对数似然函数改写为

$$l(\boldsymbol{\beta})=\sum_{i=1}^{m}y_i\log p_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})+(1-y_i)\log p_0(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta}) \tag{5-12}$$

最大化对数似然函数等价于最小化如下损失函数：

$$l'(\boldsymbol{\beta})=-\sum_{i=1}^{m}y_i\log p_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})+(1-y_i)\log p_0(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta}) \tag{5-13}$$

此时，当 $y_i=1$ 时，损失函数等于 $l'(\boldsymbol{\beta})=-\log p_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$，如果想要损失函数尽可能小，则要求 $p_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$尽可能大。注：根据 Sigmoid 函数的性质，$p_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$尽可能大也就是尽可能接近于 1。

当 $y_i=0$ 时，损失函数等于 $l'(\boldsymbol{\beta})=-\ln p_0(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$，如果想要损失函数尽可能小，则要求 $p_0(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$尽可能大。注：根据 Sigmoid 函数的性质，$p_0(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$尽可能大也就是尽可能接近于 1，即 $p_1(\widetilde{\boldsymbol{x}}_i;\boldsymbol{\beta})$尽可能接近于 0。

显然，式(5-13)是一个关于 $\boldsymbol{\beta}$ 的凸函数，根据凸优化理论，诸如梯度下降法、牛顿法都可以用于求解 $\boldsymbol{\beta}$ 的最优解。

以最简单的梯度下降法为例，其迭代公式如下：

$$\boldsymbol{\beta}_{t+1}=\boldsymbol{\beta}_t-\alpha\frac{\partial l'(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} \tag{5-14}$$

其中,α 为学习率,$\frac{\partial l'(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}}$为梯度,具体的表达式为

$$\frac{\partial l'(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}}=\sum_{i=1}^{m}(\boldsymbol{\beta}^{\mathrm{T}}\tilde{\boldsymbol{x}}_i-y_i)\tilde{\boldsymbol{x}}_i \tag{5-15}$$

5.2 K 最近邻算法

K 最近邻算法(K-Nearest Neighbors, KNN)是机器学习算法中最简单、最基础的有监督分类算法之一。它通过测量不同特征之间的距离确定邻近数据,并使用邻近数据的标签对单个数据点进行预测。既可以用于回归问题,也能用于分类问题。

KNN 算法是一种比较特殊的机器学习算法,它没有一般意义上的学习过程,其工作原理是利用数据对特征向量空间进行划分,并将划分结果作为最终的模型。具体步骤如下。

(1) 确定距离度量。

为了确定哪些数据点是邻近数据,我们首先需要计算出查询点和其他数据点之间的距离。以下罗列一些常用的距离指标。

- 欧几里得距离。是一种最常用的距离度量。以 n 维空间为例,两个向量 $\boldsymbol{a}=[x_{1,1},x_{1,2},\cdots,x_{1,n}]^{\mathrm{T}}$ 和 $\boldsymbol{b}=[x_{2,1},x_{2,2},\cdots,x_{2,n}]^{\mathrm{T}}$ 之间的距离

$$d(\boldsymbol{a},\boldsymbol{b})=\sqrt{\sum_{k=1}^{n}(x_{1k}-x_{2k})^2} \tag{5-16}$$

- 曼哈顿距离。也被称为城市街区距离,是一种流行的距离度量,用于测量两点之间距离的绝对值。

$$d(\boldsymbol{a},\boldsymbol{b})=\sum_{k=1}^{n}|x_{1k}-x_{2k}| \tag{5-17}$$

- 闵可夫斯基距离。闵可夫斯基距离是欧几里得和曼哈顿距离度量的广义形式。

$$d(\boldsymbol{a},\boldsymbol{b})=\left(\sum_{k=1}^{n}|x_{1k}-x_{2k}|\right)^{\frac{1}{p}} \tag{5-18}$$

当 $p=1$ 时,上述公式表示曼哈顿距离;当 $p=2$ 时,上述公式表示欧几里得距离。

- 汉明距离。汉明距离表示的是两个相同长度字不同位的数量,主要用于数据传输的纠错编码。具体地说,我们可对两个字符串进行异或运算,并统计结果为 1 的位数。

$$d(\boldsymbol{a},\boldsymbol{b})=\frac{1}{n}\sum_{x_{1k}\neq x_{2k}}1 \tag{5-19}$$

- 切比雪夫距离。切比雪夫距离有时也称为 L_∞ 度量,定义为两个向量各个坐标差绝对值的最大值。若将国际象棋棋盘放入一个二维坐标中,各个格子的边长定为 1,则国王从一个位置到另一个位置需要的步数恰好是两个位置的切比雪夫距离,所以切比雪夫距离也称为棋盘距离。

$$d(\boldsymbol{a},\boldsymbol{b})=\max_k|x_{1k}-x_{2k}| \tag{5-20}$$

(2) 计算测试数据到训练数据集中各个数据之间的距离。

(3) 按照距离的远近进行排序。

(4) 统计 k 个最近邻的类别。

(5) 选择频次最高的类别作为测试数据的类别。

从上述步骤可知，KNN 算法的核心思想是：当预测一个新样本的类别时，根据距离它最近的 k 个样本点是什么类别，通过投票的方式，来判断新样本属于哪个类别。

如图 5-4 所示，当选择 $k=3$ 时，可以预测中间的圆隶属于正方形这个类别。

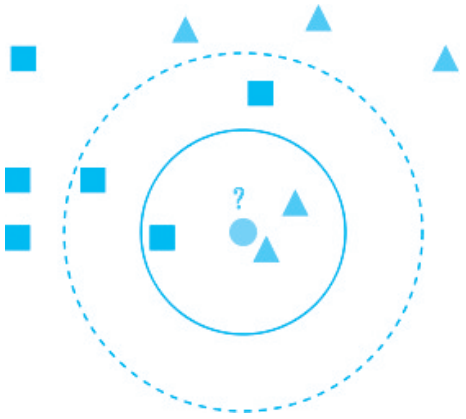

图 5-4　KNN 算法

显然，k 的取值对于 KNN 算法的性能非常重要。那么如何选择 k 的值呢？答案是通过交叉验证来选择。

具体来说，首先将样本数据按照一定比例(如 6∶4)拆分成训练数据集合验证数据集，从选取一个较小的 k 值开始，不断增加 k 的值，然后计算验证数据集的错误率。最终选择一个合适的 k 值。

一般来说，增大 k 值的时候，错误率会先降低，因为周围有更多的样本可以用于训练，因此可以提升分类性能。但注意，k 值进一步增大之后，错误率会升高。

对 KNN 算法进行总结，可知它的优点如下。

- 简单易用，模型训练时间快。
- 对异常值不敏感。
- 随着新样本的增加，KNN 算法会根据新数据对模型进行调整，因此适应性比较强。

KNN 算法的缺点如下。

- 对数据规模比较敏感，KNN 本质上是一种惰性算法，与其他分类器相比，它占用了更多的内存。
- KNN 算法在高位数据输入时表现不佳，由于维数灾难，KNN 算法容易出现过拟合的情况。利用特征选择和降维技术可以在一定程度上解决这些问题。

5.3　朴素贝叶斯分类器

贝叶斯分类器是在既定概率框架下进行决策的方法。对于一般的分类任务而言，假定相关的概率全部已知，我们可以利用这些概率来判断最优的标签。总体来说，贝叶斯分类器是以贝叶斯定理为基础得到的一系列分类算法的总称。其中，朴素贝叶斯是最经典的贝叶

斯分类器算法。在介绍算法之前,首先对贝叶斯定理相关的概率公式进行说明。

(1) 先验概率。即基于统计的概率,是基于以往历史经验和分析得到的结果,不需要依赖当前发生的条件。

(2) 后验概率。是从条件概率而来,由因推果,是基于当下发生了事件之后计算的概率,依赖于当前发生的条件。

(3) 条件概率。记事件 A 发生的概率 $P(A)$,事件 B 发生的概率 $P(B)$,则在事件 B 发生的前提下,事件 A 发生的概率即为条件概率,记为 $P(A|B)$。

$$P(A \mid B)=\frac{P(AB)}{P(B)} \tag{5-21}$$

贝叶斯公式:基于条件概率,通过 $P(B|A)$来求取 $P(A|B)$,即

$$P(A \mid B)=\frac{P(B \mid A)P(A)}{P(B)} \tag{5-22}$$

将 A 看成是“规律”,B 看成是“现象”,那么贝叶斯公式看成:

$$P(\text{规律} \mid \text{现象})=\frac{P(\text{现象} \mid \text{规律})P(\text{规律})}{P(\text{现象})} \tag{5-23}$$

全概率公式:表示若事件 $A_1,A_2,\cdots,A_n$ 构成一个完备事件组且都有正概率,则对任意一个事件 B 都有公式成立:

$$P(B)=\sum_{i=1}^{n}P(B \mid A_i)P(A_i) \tag{5-24}$$

将全概率公式代入贝叶斯公式中,得到

$$P(A \mid B)=\frac{P(B \mid A)P(A)}{\sum_{i=1}^{n}P(B \mid A_i)P(A_i)} \tag{5-25}$$

朴素贝叶斯(Naive Bayes)算法是一种经典的分类算法,它的经典案例是用于文本分类。朴素贝叶斯算法本质上是典型的生成式方法,首先由训练数据学习联合概率分布 $P(A,B)$,然后求解后验概率分布 $P(B|A)$。

朴素贝叶斯算法的基本假设是特征条件假设,即假设每个特征之间没有联系,给定训练数据集,其中每个样本 x 都包括 n 维特征,即 $x=(x_1,x_2,\cdots,x_n)$,类标签集合含有 k 种类别,即 $c=(c_1,c_2,\cdots,c_k)$。

对于给定的新样本 x,判断其属于哪个标记的类别,根据贝叶斯定理,可以得到 x 属于类别 c 的概率,即 $P(c|x)=\dfrac{P(x|c)P(c)}{P(x)}$。

后验概率最大的类别记为预测类别,即

$$\text{argmax}_{c_k}P(c_k \mid x) \tag{5-26}$$

朴素贝叶斯算法对条件概率分布做出独立性假设,通俗地讲,假设各个维度的特征 x_1,$x_2,\cdots,x_n$ 互相独立,在这个假设的前提下,条件概率可以转换为

$$P(x \mid c_k)=P(x_1,x_2,\cdots,x_n \mid c_k)=\prod_{i=1}^{n}P(x_i \mid c_k) \tag{5-27}$$

代入上述贝叶斯公式,得到

$$P(c_k \mid x)=\frac{P(c_k)\times\prod_{i=1}^{n}P(x_i \mid c_k)}{\sum_{k}P(c_k)\times\prod_{i=1}^{n}P(x_i \mid c_k)} \tag{5-28}$$

由此，朴素贝叶斯分类器可以被表示为

$$f(x)=\operatorname{argmax}_{c_k}P(c_k \mid x)=\operatorname{argmax}_{c_k}\frac{P(c_k)\times\prod_{i=1}^{n}P(x_i \mid c_k)}{\sum_{k}P(c_k)\times\prod_{i=1}^{n}P(x_i \mid c_k)} \tag{5-29}$$

因为对所有的类别，上式中的分母的值是一样的，所以在优化问题中，可以忽略分母部分，朴素贝叶斯分类器最终可以表示为

$$f(x)=\operatorname{argmax}_{c_k}P(c_k)\times\prod_{i=1}^{n}P(x_i \mid c_k) \tag{5-30}$$

朴素贝叶斯只适用于特征之间是条件独立的情况下，否则效果不好。注意：这里的朴素指的是条件独立，是一个很强的假设。在实际应用中，一般情况下无法满足。

以西瓜数据集对朴素贝叶斯分类器进行说明，拟利用表 5-1 所示数据集训练一个朴素贝叶斯分类器，并对测试样本(表 5-2)进行分类。

表 5-1　西瓜数据集——训练集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软黏	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软黏	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
10	青绿	硬挺	清脆	清晰	平坦	软黏	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	硬挺	浊响	模糊	平坦	软黏	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软黏	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

表 5-2 西瓜数据集——测试集

测试样本	青绿	蜷缩	浊响	清晰	凹陷	硬滑	?

我们首先评估先验概率 $P(c)$,可知 $P(\text{好瓜})=\frac{8}{17}$和 $P(\text{坏瓜})=\frac{9}{17}$。然后,我们为每个属性估计得出条件概率 $P(x_i|c)$。

$$P_{\text{青绿}|\text{好瓜}}=P(\text{色泽}=\text{青绿}|\text{好瓜})=\frac{3}{8} \quad P_{\text{青绿}|\text{坏瓜}}=P(\text{色泽}=\text{青绿}|\text{坏瓜})=\frac{3}{9}$$

$$P_{\text{蜷缩}|\text{好瓜}}=P(\text{根蒂}=\text{蜷缩}|\text{好瓜})=\frac{5}{8} \quad P_{\text{蜷缩}|\text{坏瓜}}=P(\text{根蒂}=\text{蜷缩}|\text{坏瓜})=\frac{3}{9}$$

$$P_{\text{清晰}|\text{好瓜}}=P(\text{纹理}=\text{清晰}|\text{好瓜})=\frac{7}{8} \quad P_{\text{清晰}|\text{坏瓜}}=P(\text{纹理}=\text{清晰}|\text{坏瓜})=\frac{2}{9}$$

$$P_{\text{凹陷}|\text{好瓜}}=P(\text{脐部}=\text{凹陷}|\text{好瓜})=\frac{6}{8} \quad P_{\text{凹陷}|\text{坏瓜}}=P(\text{脐部}=\text{凹陷}|\text{坏瓜})=\frac{2}{9}$$

$$P_{\text{硬滑}|\text{好瓜}}=P(\text{触感}=\text{硬滑}|\text{好瓜})=\frac{6}{8} \quad P_{\text{硬滑}|\text{坏瓜}}=P(\text{触感}=\text{硬滑}|\text{坏瓜})=\frac{6}{9}$$

基于上述结果,我们可得

$$P(\text{好瓜})=P_{\text{青绿}|\text{好瓜}}\times P_{\text{蜷缩}|\text{好瓜}}\times P_{\text{浊响}|\text{好瓜}}\times P_{\text{清晰}|\text{好瓜}}\times P_{\text{凹陷}|\text{好瓜}}\times P_{\text{硬滑}|\text{好瓜}}=0.087$$

$$P(\text{坏瓜})=P_{\text{青绿}|\text{坏瓜}}\times P_{\text{蜷缩}|\text{坏瓜}}\times P_{\text{浊响}|\text{坏瓜}}\times P_{\text{清晰}|\text{坏瓜}}\times P_{\text{凹陷}|\text{坏瓜}}\times P_{\text{硬滑}|\text{坏瓜}}=1.62\times10^{-3}$$

显然,由于 $0.087>1.62\times10^{-3}$,朴素贝叶斯分类器可以将测试样本 1 划分为好瓜。

这里需要注意的是,若某个属性值在训练集中没有与某个类同时出现过,这时候上述判别方法会出现问题。例如,在使用上述西瓜数据集训练分类器问题时,对于一个“敲声=清脆”的测试样本,计算得到的概率是

$$P_{\text{清脆}|\text{好瓜}}=P(\text{敲声}=\text{清脆}|\text{好瓜})=\frac{0}{8}$$

有了上述数值,不论该样本的其他属性是什么,哪怕其他的属性上明显是好瓜,分类的结果都会是“坏瓜”,这显然有问题。

为了解决零概率的问题,法国数学家拉普拉斯最早提出用加 1 的方法估计没有出现过的现象的概率,所以加法平滑也称拉普拉斯平滑。假定训练样本很大时,每个分量 x 的计数加 1 造成的估计概率变化可以忽略不计,但可以方便有效地避免零概率的问题。

$P(c_k)\times\prod_{i=1}^{n}P(\boldsymbol{x}_i|c_k)$是一个多项乘法公式,其中有一项数值为 0,则整个公式就为 0,这样显然不合理。为此,我们可以在分子和分母上各自加入一个数值,即

$$P(y)=\frac{|D_c|+1}{|D|+N} \tag{5-31}$$

其中,$|D_c|$表示分类 c 的样本数,$|D|$为样本总数,N 是分类总数。

$$P(\boldsymbol{x}_i\mid D_c)=\frac{|D_{c,x_i}|+1}{|D_c|+N_i} \tag{5-32}$$

其中,$|D_{c,x_i}|$表示分类 c 属性 i 的样本数,$|D_c|$表示分类 c 的样本数,N_i 表示 i 属性可能的取值。

在实际应用中也经常使用加 $\lambda(0\leqslant\lambda\leqslant1)$来代替简单加 1。如果对 N 个计数都加入λ，这个时候分母也要加入 $N\lambda$。

例如，利用西瓜数据集，我们可以计算得到类先验概率为

$$\hat{P}(\text{好瓜})=\frac{8+1}{17+2}\approx0.474 \qquad \hat{P}(\text{坏瓜})=\frac{9+1}{17+2}\approx0.526$$

同理，条件概率可以改为

$$\hat{P}_{\text{青绿}|\text{好瓜}}=\hat{P}(\text{色泽}=\text{青绿}|\text{好瓜})=\frac{3+1}{8+3}\approx0.364$$

$$\hat{P}_{\text{青绿}|\text{坏瓜}}=\hat{P}(\text{色泽}=\text{青绿}|\text{坏瓜})=\frac{3+1}{9+3}\approx0.333$$

上述为 0 的条件概率基于上述方法，可以改写为

$$\hat{P}_{\text{清脆}|\text{好瓜}}=\hat{P}(\text{敲声}=\text{清脆}|\text{好瓜})=\frac{0+1}{8+3}\approx0.091$$

通过拉普拉斯平滑避免了通过训练样本数量不够导致条件概率为 0 的问题。随着训练集规模变大，通过引入先验信息，使得估计值趋近于实际概率值。

朴素贝叶斯算法的优缺点列举如下。

1. 优点

- 朴素贝叶斯模型有稳定的分类效率。
- 对小规模的数据表现很好，能处理多分类任务，适合于增量训练，尤其是数据量超出内存时，可以一批批去增量训练。
- 对缺失数据不太敏感，算法也比较简单，常用于文本分类。

2. 缺点

- 需要知道先验概率，且先验概率很多时候取决于假设，假设的模型可以有很多种，因此在某些时候会由于假设先验模型导致预测效果不佳。
- 对输入数据的表达形式很敏感(如离散、连续、极大值、极小值)。

5.4　决　策　树

决策树(Decision Tree)是一种常用的机器学习算法。决策树将算法组织成一棵树的形式。其实就是将平时所说的 if-then 语句构建成了树的形式。如图 5-5 所示，这个决策树主要包括三部分：内部节点、叶节点和边。内部节点是划分的属性，边代表划分的条件，叶节点表示类别。构建决策树就是一个递归地选择内部节点，计算划分条件的边，最后到达叶节点的过程。

例如，以西瓜的分类为例，在对“这个瓜是不是好瓜”的问题进行判断时，通常需要经过一系列的分析，首先要看“这个瓜是什么颜色”，如果是“青绿色”，再看“这个瓜的根蒂是什么状态”，如果是“蜷缩”，再来判断“这个瓜敲起来是什么声音”，如果是“浑浊”，我们可以得到最后的决策：这是一个好瓜。

显然，上述的决策过程与我们在直观上的决策过程是一致的。例如，最终的结论是判断一个瓜是不是好瓜，在决策过程进行不断地细化分析，即对这个瓜的各个方面的属性进行决策。基于每次测试的结果，要么得到最终的结论，要么基于上次决策的结果进行进一步的决