朴素贝叶斯算法

陈 鑫

2023 年 4 月 28 日

目录

目录

贝叶斯分类是一类分类算法的总称，这类算法均以贝叶斯定理为基础，故统称为贝叶斯分类。朴素贝叶斯分类是贝叶斯分类中最简单，也最常见的一种分类方法，是基于贝叶斯定理与特征条件独立假设的分类方法。

朴素贝叶斯分类器（Naive Bayes Classifier，NBC）发源于古典数学理论，有着坚实的数学基础，以及稳定的分类效率。同时，NBC 模型所需估计的参数很少，对缺失数据不太敏感，算法也比较简单。

贝叶斯方法的特点是结合先验概率和后验概率，即避免了只使用先验概率的主观偏见，也避免了单独使用样本信息的过拟合现象。贝叶斯分类算法在数据集较大的情况下表现出较高的准确率，同时算法本身也比较简单。

理论上，NBC 模型与其他分类方法相比具有最小的误差率。但是实际上并非总是如此，这是因为 NBC 模型假设属性之间相互独立，这个假设在实际应用中往往是不成立的，这给 NBC 模型的正确分类带来了一定影响。

在所有的机器学习分类算法中，朴素贝叶斯和其他绝大多数的分类算法都不同。对于大多数的分类算法，比如决策树，KNN，Logistic 回归，支持向量机等都是判别方法，即直接学习出特征输出 Y 和特征 X 之间的关系，要么是决策函数 $Y = f(X)$，要么是决策条件分布 $P(Y|X)$，但是朴素贝叶斯却是生成方法，也就是直接找出特征输出 Y 和特征 X 的联合分布 $P(X, Y)$，然后得出分类的结果 $P(Y|X) = P(XY)/P(X)$。

贝叶斯决策理论（Bayesian Decision Theory）是概率框架下实施决策的基本方法，对于分类问题来说，基于贝叶斯的分类器都是在概率已知的理想情况下，贝叶斯决策理论考虑如何基于概率和误判损失来标记数据的类别，朴素贝叶斯法（Naive Bayes）是基于贝叶斯定理与特征条件独立假设的分类方法。对于给定的训练数据集，首先基于特征条件独立假设学习输入/输出的联合概率分布；然后基于此模型，对给定的输入 x，利用贝叶斯定理求出后验概率最大的输出 y 。

目录

数学基础

独立性：若干事件 $A_1, A_2, \cdots, A_n$ 之积的概率，等于各事件概率的乘积，$P(A_1, A_2, \cdots, A_n) = \prod_{i=1}^{n} P(A_i)$

条件概率：假设有 A，B 两个事件，且 $P(B) \neq 0$，则在 B 事件发生的条件下，A 事件发生的条件概率，记为 $P(A \mid B)$，定义为 $P(A \mid B) = \frac{P(AB)}{P(B)}$，当 A，B 相互独立时，有 $P(A \mid B)$=$P(A)$。

全概率公式：设 $B_1, B_2, \cdots$ 为有限或无限个事件，它们两两互斥且在每次试验中至少发生一个，即 $B_iB_j = \emptyset$（不可能事件），$B_1 + B_2 + \cdots = \Omega$（必然事件），把这组事件称为完备事件群。则对于任一事件 A，有 $P(A) = P(AB_1) + P(AB_2) + \cdots$，再由条件概率的定义，有 $P(AB_i) = P(B_i)P(A \mid B_i)$，所以有：

$$P(A) = P(B_1)P(A \mid B_1) + P(B_2)P(A \mid B_2) + \cdots = \sum_{i=1}^{\infty} P(B_i)P(A \mid B_i)$$

称为全概率公式。

贝叶斯公式：在全概率公式的假定下，有

$$\begin{aligned} P(B_i \mid A) &= \frac{P(AB_i)}{P(A)} \\ &= \frac{P(B_i)P(A \mid B_i)}{\sum_j P(B_j)P(A \mid B_j)}, \end{aligned}$$

这个公式就是著名的贝叶斯公式。

贝叶斯公式的解释：先看 $P(B_1), P(B_2), \cdots$，它是在没有进一步的信息（不知事件 A 是否发生）的情况下，人们对各事件 $B_1, B_2, \cdots$ 发生可能性大小的认识，现在有了新的信息（知道 A 发生），人们对 $B_1, B_2, \cdots$ 发生的可能性大小有了新的估计。这种情况在日常生活中屡见不鲜：原以为不太可能的一种情况，可以因某种事件的发生而变得很有可能，或者相反，贝叶斯公式从数量上刻画了这种变化。

如果把事件 A 看成“结果”，把各事件 $B_1, B_2, \cdots$ 看成导致这个结果的可能的“原因”，则可以形象地把全概率公式看作“由原因推结果”；而贝叶斯公式则恰好相反，其作用在于“由结果推原因”：现在有一个“结果”A 已发生了，在众多可能的“原因”中，到底是哪一个导致了这个结果？贝叶斯公式说，各原因可能性的大小与 $P(B_i \mid A)$ 成比例。例如：某地区发生了一起刑事案件，按平日掌握的资料，嫌疑犯有张三、李四等人，在不知道案情细节（事件 A）之前，人们对上述各人作案的可能性有个估计（相当于 $B_1, B_2, \cdots$），那是基于他们过去在警察局里的记录。但在知道了案情细节以后，这个估计就有了变化，例如，原来以为不太可能的张三，成了重点嫌疑人。

目录

朴素贝叶斯的相关概念

设输入空间 $\mathcal{X} \subseteq \mathbb{R}^n$ 为 n 维向量的集合，输出空间为标签集合 $\mathcal{Y} = \{y_1, y_2, \cdots, y_k\}$，输入数据 $x = (x_1, x_2, \cdots, x_n) \in \mathcal{X}$，输出标签 $y \in \mathcal{Y}$。X 是定义在输入空间 $\mathcal{X}$ 上的随机向量，Y 是定义在输出空间 $\mathcal{Y}$ 上的随机变量。$P(X, Y)$ 是 X 和 Y 的联合概率分布。

数据集 $S = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$ 由 $P(X, Y)$ 独立同分布产生。朴素贝叶斯法通过训练数据集学习联合概率分布 $P(X, Y)$，其实就是学习先验概率分布和条件概率分布：

先验概率分布：$P(Y = y_i), i = 1, 2, \cdots, k$

条件概率分布：$P(X = x \mid Y = y_i) = P(X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n \mid Y = y_i), i = 1, 2, \cdots, k$，$X_j$ 是 X 的第 j 个特征，$j = 1, 2, \cdots, n$。

由条件概率公式

$$P(X \mid Y) = \frac{P(X, Y)}{P(Y)}$$

可以求出联合概率分布 $P(X, Y)$。

朴素贝叶斯算法对条件概率分布做了条件独立的假设，即

$$P(X = x \mid Y = y_i) = P(X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n \mid Y = y_i) = \prod_{j=1}^{n} P(X_j = x_j \mid Y = y_i)$$

后验概率分布：$P(Y = y_i \mid X = x), i = 1, 2, \cdots, k$

朴素贝叶斯法分类

朴素贝叶斯法分类时，对给定的输入 x，通过上述学习到的模型计算后验概率分布 $P(Y=y_i \mid X=x), i=1,2,\cdots,k$，将后验概率最大的类别作为 x 的类别输出。后验概率根据贝叶斯定理进行计算：

$$
\begin{aligned}
P(Y=y_i \mid X=x) &= \frac{P(Y=y_i)P(X=x \mid Y=y_i)}{P(X=x)} \\
&= \frac{P(Y=y_i)P(X=x \mid Y=y_i)}{\sum_{r=1}^{k} P(Y=y_r)P(X=x \mid Y=y_r)}
\end{aligned}
$$

分母利用到了 $P(X=x)$ 的全概率公式。

由条件独立性假设，上式进一步简化为

$$
P(Y=y_i \mid X=x) = \frac{P(Y=y_i)\prod_{j=1}^{n} P(X_j=x_j \mid Y=y_i)}{\sum_{r=1}^{k} P(Y=y_r)\prod_{j=1}^{n} P(X_j=x_j \mid Y=y_r)}
$$

这就是朴素贝叶斯分类的基本公式。

朴素贝叶斯法分类

朴素贝叶斯分类器就是取后验概率最大时的分类

$$y = f(x) = \arg\max_{y_i} \frac{P(Y = y_i)\prod_{j=1}^{n} P(X_j = x_j \mid Y = y_i)}{\sum_{r=1}^{k} P(Y = y_r)\prod_{j=1}^{n} P(X_j = x_j \mid Y = y_r)}$$

显然，上式中的分母对于所有类别来说都是一样的，对计算结果不会产生影响，所以，朴素贝叶斯分类器可以简化为：

$$y = f(x) = \arg\max_{y_i} P(Y = y_i)\prod_{j=1}^{n} P(X_j = x_j \mid Y = y_i)$$

朴素贝叶斯分类器的后验概率最大化等价于 $0-1$ 损失函数时的期望损失最小化。

朴素贝叶斯算法：实例 x 类别的判断

Algorithm 1: 朴素贝叶斯算法

Input: 训练数据 $S=\left\{(\boldsymbol{x}^{(1)},y^{(1)}),(\boldsymbol{x}^{(2)},y^{(2)}),\cdots,(\boldsymbol{x}^{(m)},y^{(m)})\right\}$, 其中 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T$，$x_j^{(i)}$ 是第 i 个样本的第 j 个特征，$x_j^{(i)}\in\left\{a_{j1},a_{j2},\cdots,a_{js_j}\right\}$，$a_{jr}$ 是第 j 个特征可能取的第 r 个值，s_j 代表第 j 个特征的取值个数，$j=1,2,\cdots,n$，$r=1,2,\cdots,s_j$，$y^{(i)}\in\{y_1,y_2,\cdots,y_k\}$；实例 $\boldsymbol{x}$；

Output: 实例 $\boldsymbol{x}$ 的分类。

1. 计算先验概率及条件概率

$$P(Y=y_i)=\frac{1}{m}\sum_{j=1}^{m}I(y^{(j)}=y_i)\quad i=1,2,\cdots,k$$

$$P(X_j=a_{jr}\mid Y=y_i)=\frac{\sum\limits_{t=1}^{m}I(x_j^{(t)}=a_{jr},y^{(t)}=y_i)}{\sum\limits_{t=1}^{m}I(y^{(t)}=y_i)}，\ i=1,2,\cdots,k；\ j=1,2,\cdots,n；\ r=1,2,\cdots,s_j；$$

$t=1,2,\cdots,m$

2. 对于给定的实例 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T$，计算

$$P(Y=y_i)\prod_{j=1}^{n}P(X_j=x_j\mid Y=y_i),i=1,2,\cdots,k$$

3. 确定实例 $\boldsymbol{x}$ 的类别

$$y=\arg\max_{y_i}P(Y=y_i)\prod_{j=1}^{n}P(X_j=x_j\mid Y=y_i)$$

由 Python 实现高斯分布的朴素贝叶斯算法

```python
import math
class NaiveBayes:
    def __init__(self):
        self.model = None
    # 数学期望
    @staticmethod
    def mean(X):
        return sum(X) / float(len(X))
    # 标准差
    def stdev(self, X):
        avg = self.mean(X)
        return math.sqrt(sum([pow(x - avg, 2) for x in X]) / float(len(X)))

    # 概率密度函数
    def gaussian_probability(self, x, mean, stdev):
        exponent = math.exp(-(math.pow(x - mean, 2) / (2 * math.pow(stdev, 2))))
        return (1 / (math.sqrt(2 * math.pi) * stdev)) * exponent

    # 求训练集X_train各个特征的均值和标准差
    def summarize(self, X_train):
        summaries = [(self.mean(data), self.stdev(data)) for data in zip(*X_train)]
        return summaries
    # 分类别 求出数学期望和标准差
    def fit(self, X, y):
        labels = list(set(y))
        data = {label: [] for label in labels}
        for x, label in zip(X, y):
            data[label].append(x) # 将相同标签的数据写到相同的字典项中（这里的字典项为列表）
        self.model = {label: self.summarize(value) for label, value in data.items()}
```

```
def calculate_probabilities(self, input_data): # 计算概率
    probabilities = {} # 空字典
    for label, value in self.model.items():
        probabilities[label] = 1
        for i in range(len(value)): # 对样本的各个特征分量计算其概率密度，然后再连乘在一起，得到最终的联合
            概率（密度）
            mean, stdev = value[i]
            probabilities[label] *= self.gaussian_probability(
                input_data[i], mean, stdev)
    return probabilities

# 先根据测试样本计算出其属于各个类别的概率，再排序，并取出最后一个（概率值最大），最后再获得其标签
def predict(self, X_test): # 类别预测
    label = sorted(self.calculate_probabilities(X_test).items(), key=lambda x: x[-1])[-1][0]
    return label

def score(self, X_test, y_test): # 准确率 accuracy
    right = 0
    for X, y in zip(X_test, y_test):# 统计测试数据预测值与真实标签相同的个数，获得准确率
        label = self.predict(X)
        if label == y:
            right += 1
    return right / float(len(X_test))
```

案例实战 7.1 由书中表的训练数据，每一列为一个样本，包含两个特征 x_1, x_2 以及标签 y。表中特征 x_1 和 x_2 的取值分别为 $A_1 = \{1, 2, 3\}$ 和 $A_2 = \{S, M, L\}$，Y 为类型标签 $y \in \{1, -1\}$。学习一个朴素贝叶斯分类器并确定 $\boldsymbol{x} = (2, S)^T$ 的类别 y。详见书中解答。

基于极大似然估计的朴素贝叶斯算法的结果不是很理想，因为在利用基于极大似然估计的朴素贝叶斯分类器时，要计算多个概率的乘积以获得某个类别的概率，如果其中一个概率值为 0，那么最后的乘积也为 0。为降低这种影响，接下来我们采用贝叶斯估计对上述算法做一下微小的修改。

Algorithm 2: 基于贝叶斯估计的朴素贝叶斯算法

Input: 训练数据 $S=\{(\boldsymbol{x}^{(1)},y^{(1)}),(\boldsymbol{x}^{(2)},y^{(2)}),\cdots,(\boldsymbol{x}^{(m)},y^{(m)})\}$, 其中 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T$，$x_j^{(i)}$ 是第 i 个样本的第 j 个特征，$x_j^{(i)}\in\{a_{j_1},a_{j_2},\cdots,a_{j_{s_j}}\}$，$a_{j_r}$ 是第 j 个特征可能取的第 r 个值，s_j 代表第 j 个特征的取值个数，$j=1,2,\cdots,n$，$r=1,2,\cdots,s_j$，$y^{(i)}\in\{y_1,y_2,\cdots,y_k\}$；实例 $\boldsymbol{x}$；

Output: 实例 $\boldsymbol{x}$ 的分类。

1. 计算先验概率及条件概率
   $P(Y=y_i)=\frac{\sum_{j=1}^m I(y^{(j)}=y_i)+\lambda}{m+k\lambda}, i=1,2,\cdots,k$
   $P(X_j=a_{j_r}\mid Y=y_i)=\frac{\sum_{t=1}^m I(x_j^{(t)}=a_{j_r},y^{(t)}=y_i)+\lambda}{\sum_{t=1}^m I(y^{(t)}=y_i)+s_j\lambda}$，$i=1,2,\cdots,k$；$j=1,2,\cdots,n$；$r=1,2,\cdots,s_j$；$t=1,2,\cdots,m$。当 $\lambda=0$ 时，就是极大似然估计，常取 $\lambda=1$，这时称为拉普拉斯平滑。
2. 对于给定的实例 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T$，计算
   $P(Y=y_i)\prod_{j=1}^n P(X_j=x_j\mid Y=y_i), i=1,2,\cdots,k$
3. 确定实例 $\boldsymbol{x}$ 的类别
   $y=\arg\max_{y_i} P(Y=y_i)\prod_{j=1}^n P(X_j=x_j\mid Y=y_i)$

案例实战 7.2 问题通 7.1 一样，按照拉普拉斯平滑估计概率，即 $\lambda = 1$。详见书中解答

案例 7.3 鸢尾花的类型判断。参考 naïve_bayes.py 和 gaussian_naïve_bayes.py

案例实战 7.4 由于鸢尾花的数据集都是连续属性，适合采用 GaussianNB 来进行实现。gaussian_naive_bayes_sklearn.py

案例实战 7.5 多项式朴素贝叶斯实现 multinomial_naive_bayes_sklearn.py

案例 7.6 使用 sklearn 自带的数据集。使用 fetch_20newsgroups 中的数据，包含了 20 个主题的 18000 个新闻组的帖子，利用多项式朴素贝叶斯进行分类。multinomial_naive_bayes_sklearn_news.py

案例实战 7.7 简单伯努利朴素贝叶斯算法测试。bernoulli_naive_bayes.py