



温州大學
WENZHOU UNIVERSITY

机器学习-第四章 朴素贝叶斯

黄海广 副教授

2021年03月

- 01** 贝叶斯方法
- 02** 朴素贝叶斯原理
- 03** 朴素贝叶斯案例
- 04** 朴素贝叶斯代码实现

1.贝叶斯方法

3

01 贝叶斯方法

02 朴素贝叶斯原理

03 朴素贝叶斯案例

04 朴素贝叶斯代码实现

1.贝叶斯方法-背景知识

4

贝叶斯分类: 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。

先验概率: 根据以往经验和分析得到的概率。我们用 $P(Y)$ 来代表在没有训练数据前假设 Y 拥有的初始概率。

后验概率: 根据已经发生的事件来分析得到的概率。以 $P(Y|X)$ 代表假设 X 成立的情下观察到 Y 数据的概率，因为它反映了在看到训练数据 X 后 Y 成立的置信度。

1.贝叶斯方法-背景知识

5

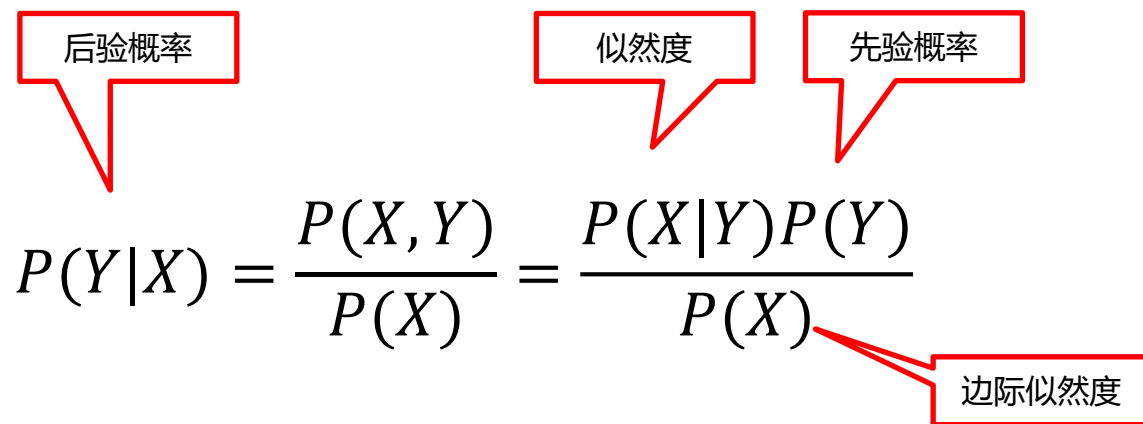
联合概率： 联合概率是指在多元的概率分布中多个随机变量分别满足各自条件的概率。 X 与 Y 的联合概率表示为 $P(X, Y)$ 、 $P(XY)$ 或 $P(X \cap Y)$ 。

假设 X 和 Y 都服从正态分布，那么 $P(X < 5, Y < 0)$ 就是一个联合概率，表示 $X < 5, Y < 0$ 两个条件同时成立的概率。表示两个事件共同发生的概率。

1. 贝叶斯方法

6

贝叶斯公式



The diagram shows the Bayes' formula with three red callout boxes. The first box, labeled '后验概率' (Posterior Probability), points to $P(Y|X)$. The second box, labeled '似然度' (Likelihood), points to $P(X|Y)$. The third box, labeled '先验概率' (Prior Probability), points to $P(Y)$. A fourth box, labeled '边际似然度' (Marginal Likelihood), points to the denominator $P(X)$ in the second fraction.

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

朴素贝叶斯法是典型的生成学习方法。生成方法由训练数据学习联合概率分布 $P(X, Y)$ ，然后求得后验概率分布 $P(Y|X)$ 。

具体来说，利用训练数据学习 $P(X|Y)$ 和 $P(Y)$ 的估计，得到联合概率分布：

$$P(X, Y) = P(X|Y) P(Y)$$

2.朴素贝叶斯原理

7

01 贝叶斯方法

02 朴素贝叶斯原理

03 朴素贝叶斯案例

04 朴素贝叶斯代码实现

2.朴素贝叶斯原理

8

判别模型和生成模型

监督学习方法又分

生成方法 (Generative approach) 和**判别方法** (Discriminative approach)

所学到的模型分别称为

生成模型 (Generative Model) 和**判别模型** (Discriminative Model)。

| 判别模型 | 生成模型 |
|--|---|
| 由数据直接学习决策函数 $Y=f(X)$ 或者条件概率分布 $P(Y X)$ 作为预测的模型，即判别模型。基本思想是有限样本条件下建立判别函数，不考虑样本的产生模型，直接研究预测模型。 | 由训练数据学习联合概率分布 $P(X, Y)$ ，然后求得后验概率分布 $P(Y X)$ 。具体来说，利用训练数据学习 $P(X Y)$ 和 $P(Y)$ 的估计，得到联合概率分布： $P(X, Y) = P(Y)P(X Y)$ ，再利用它进行分类。 |
| 线性回归、逻辑回归、感知机、决策树、支持向量机..... | 朴素贝叶斯、HMM、深度信念网络(DBN)..... |

2.朴素贝叶斯原理

9

1. 朴素贝叶斯法是典型的生成学习方法。

生成方法由训练数据学习联合概率分布 $P(X, Y)$ ，然后求得后验概率分布 $P(Y|X)$ 。具体来说，利用训练数据学习 $P(X|Y)$ 和 $P(Y)$ 的估计，得到联合概率分布：

$$P(X, Y) = P(Y)P(X|Y)$$

概率估计方法可以是极大似然估计或贝叶斯估计。

2.朴素贝叶斯原理

10

2. 朴素贝叶斯法的基本假设是条件独立性。

$$P(X = x|Y = c_k) = P(x^{(1)}, \dots, x^{(n)}|y^k) = \prod_{j=1}^n P(x^{(j)}|Y = c_k)$$

c_k 代表类别， k 代表类别个数。

这是一个较强的假设。由于这一假设，模型包含的条件概率的数量大为减少，朴素贝叶斯法的学习与预测大为简化。因而朴素贝叶斯法高效，且易于实现。其缺点是分类的性能不一定很高。

2.朴素贝叶斯原理

11

3. 朴素贝叶斯法利用贝叶斯定理与学到的联合概率模型进行分类预测

我们要求的是 $P(Y|X)$ ，根据生成模型定义我们可以求 $P(X, Y)$ 和 $P(Y)$ 假设中的特征是条件独立的。这个称作朴素贝叶斯假设。形式化表示为，（如果给定 Z 的情况下， X 和 Y 条件独立）：

$$P(X|Z) = P(X|Y, Z)$$

也可以表示为：

$$P(X, Y|Z) = P(X|Z)P(Y|Z)$$

2.朴素贝叶斯原理

12

用于文本分类的朴素贝叶斯模型，这个模型称作多值伯努利事件模型。

在这个模型中，我们首先随机选定了邮件的类型 $p(y)$ ，然后一个人翻阅词典的所有词，随机决定一个词是否出现依照概率 $p(x^{(1)}|y)$ ，出现标示为1，否则标示为0。假设有50000个单词，那么这封邮件的概率可以表示为：

$$\begin{aligned} & p(x^{(1)}, \dots, x^{(50000)} | y) \\ &= p(x^{(1)} | y) p(x^{(2)} | y, x^{(1)}) p(x^{(3)} | y, x^{(1)}, x^{(2)}) \cdots p(x^{(50000)} | y, x^{(1)}, \dots, x^{(49999)}) \\ &= p(x^{(1)} | y) p(x^{(2)} | y) p(x^{(3)} | y) \cdots p(x^{(50000)} | y) \\ &= \prod_{i=1}^m p(x^{(i)} | y) \end{aligned}$$

2.朴素贝叶斯原理

13

独立性

将输入 x 分到后验概率最大的类 y 。

$$y = \operatorname{argmax}_{c_k} P(Y = c_k) \prod_{j=1}^n P(X_j = x^{(j)} | Y = c_k)$$

后验概率最大等价于0-1损失函数时的期望风险最小化。

2.朴素贝叶斯原理

14

$$y = \operatorname{argmax}_{c_k} P(Y = c_k) \prod_{j=1}^n P(X_j = x^{(j)} | Y = c_k)$$

$X = \{x_1, x_1, \dots, x_n\}$ ，为 n 维向量的集合

$Y = \{c_1, c_2, \dots, c_k\}$ ， K 为类别数

训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ 由 $P(X, Y)$ 独立同分布产生。

2.朴素贝叶斯原理

15

朴素贝叶斯法对**条件概率分布作了条件独立性的假设**。由于这是一个较强的假设，朴素贝叶斯法也由此得名。具体地，条件独立性假设是：

$$\begin{aligned} P(X = x|Y = c_k) &= P(X^{(1)} = x^{(1)}, X^{(2)} = x^{(2)}, \dots, X^{(n)} = x^{(n)}|Y = c_k) \\ &= \prod_{j=1}^n P(X^{(j)} = x^{(j)}|Y = c_k) \end{aligned} \quad (1)$$

2.朴素贝叶斯原理

16

朴素贝叶斯法分类时，对给定的输入 x ，通过学习到的模型计算后验概率分布 $P(Y = c_k|X = x)$ ，将后验概率最大的类作为 x 的类输出。根据贝叶斯定理：

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

可以计算后验概率

$$P(Y = c_k|X = x) = \frac{P(X = x|Y = c_k)P(Y = c_k)}{\sum_{k=1}^K P(X = x|Y = c_k)P(Y = c_k)} \quad (2)$$

2.朴素贝叶斯原理

17

将式(1)代入公式(2)，可以得到

$$P(Y = c_k | X = x) = \frac{\prod_{j=1}^n P(X^{(j)} = x^{(j)} | Y = c_k) P(Y = c_k)}{\sum_{k=1}^K \prod_{j=1}^n P(X^{(j)} = x^{(j)} | Y = c_k) P(Y = c_k)}$$

贝叶斯分类器可以表示为：

$$y = f(x) = \operatorname{argmax}_{c_k} \frac{\prod_{j=1}^n P(X^{(j)} = x^{(j)} | Y = c_k) P(Y = c_k)}{\sum_{k=1}^K \prod_{j=1}^n P(X^{(j)} = x^{(j)} | Y = c_k) P(Y = c_k)}$$

上式中分母中 c_k 都是一样的，即不会对结果产生影响，即

$$y = f(x) = \operatorname{argmax}_{c_k} \prod_{j=1}^n P(X^{(j)} = x^{(j)} | Y = c_k) P(Y = c_k)$$

3.朴素贝叶斯案例

18

01 贝叶斯方法

02 朴素贝叶斯原理

03 朴素贝叶斯案例

04 朴素贝叶斯代码实现

3.朴素贝叶斯案例

19

假设我们正在构建一个分类器，该分类器说明文本是否与运动(Sports)有关。我们的训练数据有5句话：

| 文本 | 标签 |
|------------------------------|------------|
| A great game | Sports |
| The election was over | Not Sports |
| Very clean match | Sports |
| A clean but forgettable game | Sports |
| It was a close election | Not Sports |

我们想要计算句子 “A very close game” 是 Sports 的概率以及它不是 Sports 的概率。

即 $P(\text{Sports} \mid \text{a very close game})$ 这个句子的类别是Sports的概率

3.朴素贝叶斯案例

20

特征：单词的频率

已知贝叶斯定理 $P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$ ，则：

$$\begin{aligned} &P(\text{Sports} \mid \text{a very close game}) \\ &= \frac{P(\text{a very close game} \mid \text{Sports}) \times P(\text{Sports})}{P(\text{a very close game})} \end{aligned}$$

由于我们只是试图找出哪个类别有更大的概率，可以舍去除数，只是比较

$P(\text{a very close game} \mid \text{Sports}) \times P(\text{Sports})$ 和

$P(\text{a very close game} \mid \text{Not Sports}) \times P(\text{Not Sports})$

3.朴素贝叶斯案例

21

我们假设一个句子中的每个单词都与其他单词无关。

$$\begin{aligned} &P(a \text{ very close game }) \\ &= P(a) \times P(\text{very}) \times P(\text{close}) \times P(\text{game}) \end{aligned}$$

$$\begin{aligned} &P(a \text{ very close game} | \text{Sports}) \\ &= P(a | \text{Sports}) \times P(\text{very} | \text{Sports}) \times P(\text{close} | \text{Sports}) \times P(\text{game} | \text{Sports}) \end{aligned}$$

3.朴素贝叶斯案例

22

计算每个类别的先验概率:

对于训练集中的给定句子,

$P(\text{Sports})$ 的概率为 $\frac{3}{5}$ 。

$P(\text{Not Sports})$ 是 $\frac{2}{5}$ 。

| 文本 | 标签 |
|------------------------------|------------|
| A great game | Sports |
| The election was over | Not Sports |
| Very clean match | Sports |
| A clean but forgettable game | Sports |
| It was a close election | Not Sports |

然后, 在计算 $P(\text{game}|\text{Sports})$ 就是 “game” 有多少次出现在 Sports 的样本, 然后除以 sports 为标签的文本的单词总数 ($3+3+5=11$) 。

因此, $P(\text{game}|\text{Sports}) = \frac{2}{11}$ 。

“close” 不会出现在任何 sports 样本中! 那就是说 $P(\text{close}|\text{Sports}) = 0$ 。

3.朴素贝叶斯案例

23

通过使用一种称为**拉普拉斯平滑**的方法：我们为每个计数加1，因此它永远不会为零。为了平衡这一点，我们将可能单词的数量添加到除数中，因此计算结果永远不会大于1。



14个单词

在这里的情况下，可能单词是['a', 'great', 'very', 'over', 'it', 'but', 'game', 'election', 'clean', 'close', 'the', 'was', 'forgettable', 'match']。

由于可能的单词数是14，因此应用平滑处理可以得到

$$P(\text{game} \mid \text{sports}) = \frac{2+1}{11+14}$$

3.朴素贝叶斯案例

24

拉普拉斯平滑是一种用于平滑分类数据的技术。引入拉普拉斯平滑法来解决零概率问题,通过应用此方法,先验概率和条件概率可以写为

$$P_{\lambda}(C_k) = P_{\lambda}(Y = C_k) = \frac{\sum_{i=1}^N I(y_i = C_k) + \lambda}{N + K\lambda}$$

$$P_{\lambda}(x_i = a_j | y = C_k) = \frac{\sum_{i=1}^N I(x_i = a_j, y_i = C_k) + \lambda}{\sum_{i=1}^N I(y_i = C_k) + A\lambda}$$

其中 K 表示类别数量, A 表示 a_j 中不同值的数量通常 $\lambda = 1$

加入拉普拉斯平滑之后, 避免了出现概率为0的情况, 又保证了每个值都在0到1的范围内, 又保证了最终和为1的概率性质。

3.朴素贝叶斯案例

25

| Word | P (word Sports) | P (word Not Sports) |
|--------------|--------------------------|-------------------------|
| a | $(2 + 1) \div (11 + 14)$ | $(1 + 1) \div (9 + 14)$ |
| very | $(1 + 1) \div (11 + 14)$ | $(0 + 1) \div (9 + 14)$ |
| close | $(0 + 1) \div (11 + 14)$ | $(1 + 1) \div (9 + 14)$ |
| game | $(2 + 1) \div (11 + 14)$ | $(0 + 1) \div (9 + 14)$ |

$$P(a | \text{Sports}) \times P(\text{very} | \text{Sports}) \times P(\text{close} | \text{Sports}) \times P(\text{game} | \text{Sports}) \times P(\text{Sports}) \\ = 2.76 \times 10^{-5} = 0.0000276$$

$$P(a | \text{Not Sports}) \times P(\text{very} | \text{Not Sports}) \times P(\text{close} | \text{Not Sports}) \\ \times P(\text{game} | \text{Not Sports}) \times P(\text{Not Sports}) \\ = 0.572 \times 10^{-5} = 0.00000572$$

4.朴素贝叶斯代码实现

26

01 贝叶斯方法

02 朴素贝叶斯原理

03 朴素贝叶斯案例

04 朴素贝叶斯代码实现

4.朴素贝叶斯代码实现

27

最常用的GaussianNB是高斯贝叶斯分类器。它假设特征的条件概率分布满足高

斯分布:
$$P(X^{(j)}|y = c_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(X^{(j)} - \mu_k)^2}{2\sigma_k^2}\right)$$

数学期望(mean): μ 方差: $\sigma^2 = \frac{\sum(X - \mu)^2}{N}$

其他贝叶斯分类器:

MultinomialNB是多项式贝叶斯分类器, 它假设特征的条件概率分布满足多项式分布;

BernoulliNB是伯努利贝叶斯分类器。它假设特征的条件概率分布满足二项分布。

4.朴素贝叶斯代码实现

28

最常用的GaussianNB是高斯朴素贝叶斯分类器的scikit-learn实现。

```
import numpy as np
X = np.array([[ -1, -1], [ -2, -1], [ -3, -2], [ 1, 1], [ 2, 1], [ 3, 2]])
Y = np.array([1, 1, 1, 2, 2, 2])
#引入高斯朴素贝叶斯
from sklearn.naive_bayes import GaussianNB
#实例化
clf = GaussianNB()
#训练数据 fit相当于train
clf.fit(X, Y)
#输出单个预测结果
print "==Predict result by predict=="
print(clf.predict([[ -0.8, -1]]))
print "==Predict result by predict_proba=="
print(clf.predict_proba([[ -0.8, -1]]))
print "==Predict result by predict_log_proba=="
print(clf.predict_log_proba([[ -0.8, -1]]))
```

4.朴素贝叶斯代码实现

29

最常用的GaussianNB是高斯朴素贝叶斯分类器的Numpy实现。
详细过程见代码。

```
class NaiveBayes:
    def __init__(self):
        print('Gaussian naive bayes model!')

    def gaussian_pdf(self, x_test, x):
        """
        计算高斯正态分布下的条件概率
        Params:
            x_test(array):
            x(array): 同属一个类别的x数据
        """
        temp1 = (x_test - x.mean(0)) * (x_test - x.mean(0))
        temp2 = x.std(0) * x.std(0)
        return np.exp(-temp1 / (2 * temp2)) / np.sqrt(2 * np.pi * temp2)

    def fit(self, x_train, y_train):
        self.x_train = x_train
        self.y_train = y_train

    def predict(self, x_test, y_test=None):
        assert len(x_test.shape) == 2
        self.classes = np.unique(np.concatenate([y_train, y_test], 0))
        pred_probs = []

        # 对于每个输入, 计算其处于每个类别的概率
        for i in self.classes:
            idx_i = self.y_train == i
            # 计算P(y)
            p_y = len(idx_i) / len(self.y_train)

            # 利用高斯概率密度函数计算P(x|y)
            p_x_y = np.prod(self.gaussian_pdf(x_test, self.x_train[idx_i]), 1)

            # 计算x, y的联合概率, P(x|y)P(y)
            prob_i = p_y * p_x_y

            pred_probs.append(prob_i)

        pred_probs = np.vstack(pred_probs)

        # 取具有最高概率的类别
        label_idx = pred_probs.argmax(0)
        y_pred = self.classes[label_idx]
        if y_test is not None:
            self._score(y_test, y_pred)
        return y_pred

    def _score(self, y_test, y_pred):
        self.score = np.count_nonzero(y_test == y_pred) / len(y_test)

model = NaiveBayes()
model.fit(X_train, y_train)
print('GaussianNB train done!')
print(model.predict([4.4, 3.2, 1.3, 0.2]))

model.score(X_test, y_test)
```

1. Prof. Andrew Ng. Machine Learning. Stanford University
2. 《统计学习方法》，清华大学出版社，李航著，2019年出版
3. 《机器学习》，清华大学出版社，周志华著，2016年出版
4. Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2006

谢 谢!



课件、视频、代码地址

32

下载地址：

<https://github.com/fengdu78/WZU-machine-learning-course>

最新更新公布在公众号“机器学习初学者”

