机器学习实验 9. 朴素贝叶斯分类器

贝叶斯定理

设 A，B 是两个事件，且 P(A)>0 则称 $P(A|B)=\frac{P(AB)}{P(B)}$为在事件 B 发生的条件下事件 A 发生的条件概率。

贝叶斯定理是关于随机事件 A 和 B 的条件概率（或边缘概率）的一则定理。贝叶斯定理告诉我们如何交换条件概率中的条件与结果，即如果已知$P(B|A)$，要求$P(A|B)$，那么可以使用贝叶斯公式:

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$$

在贝叶斯法则中，每个名词都有约定俗成的名称：$P(A)$是 A 的先验概率或边缘概率。之所以称为"先验"是因为它不考虑任何 B 方面的因素。$P(A|B)$是已知 B 发生后 A 发生的条件概率,也由于得自 B 的取值而被称作 A 的后验概率。$P(B|A)$是已知 A 发生后 B 发生的条件概率，也由于得自 A 的取值而被称作 B 的后验概率。$P(B)$是 B 的先验概率或边缘概率，也作标准化常量（normalized constant）。按这些术语，贝叶斯法则可表述为：后验概率 = (似然度×先验概率)/标准化常量。

贝叶斯决策论

贝叶斯决策论是概率框架下实施决策的基本方法。对分类任务来说，在所有相关概率都已知的理想情形下，贝叶斯决策论考虑如何基于这些概率和误判损失来选择最优的类别标记。λ_{ij} 表示将一个真实标记为c_j的样本误分类为c_i时产生的损失。后验概率$p(c_i|x)$表示将样本 x 分类给c_i的概率，那么将样本 x 分类成c_i产生的条件风险（conditional risk）为:

$$R(c_i|x) = \sum_{j=1}^{N} \lambda_{ij} p(c_j|x)$$

我们的目标是寻找一个判定标准，以最小化总体风险。这个判定准则也叫做贝叶斯判定准则(Bayes decision rule)：为最小化总体风险，只需要在每个样本上选择那个能使条件风险$R(c|x)$最小的类别标记，即

$$h^*(x) = \underset{c\in y}{argmin}\, R(c|x)$$

此时，h^*称为贝叶斯最优分类器(Bayes optional classifier)，与之对应的总体风险$R(h^*)$称之为贝叶斯风险(Bayes risk)，$1 - R(h^*)$反映了分类器所能达到的最好性能，即通过机器学习所能产生的模型精度的理论上限。具体来说，若目标是最小化分类错误率，则误判损失λ_{ij}可写为

$$\lambda_{ij} = \begin{cases} 0, & if\ i=j; \\ 1, & otherwise, \end{cases}$$

此时，条件风险可写为

$$R(c|x) = 1 - P(c|x)$$

于是，最小化分类错误率的贝叶斯最优分类器为：

$$h^*(x) = \underset{c\in y}{argmax}\, P(c|x)$$

朴素贝叶斯算法原理概述

假设有一个已标记的数据集$[x^{(i)}, y^{(i)}]$，其中$y = \{c_1, c_2, \dots, c_k\}$表示 k 种可能的类别标记，$x^{(i)} = [x_1, x_2, \dots, x_n]$表示含有 n 维属性的数据对象。针对一个新的样本 x，我们要预测 x 所属类别，即计算c_i：

$$c_i = \underset{c_i\in y}{argmax}(p(c_i|x))$$

上式表示，已知待分类数据对象 x 的情况下，分别计算 x 属于c_1、c_2、…、c_k的概率，选取其中概率的最大值，此时所对应的c_i，即为 x 所属类别。根据贝叶斯

定理，由证据因子$p(x)$、先验概率$p(c_i)$和类条件概率$p(x|c_i)$计算出后验概率 P $p(c_i|x)$，计算方式如下：

$$p(c_i|x) = \frac{p(x|c_i)p(c_i)}{p(x)}$$

基于贝叶斯公式来估计后验概率的难点在于类条件概率$p(x|c_i)$是所有属性上的联合概率，难以从有限的训练样本直接估计。因此，朴素贝叶斯分类器采用了“属性条件独立性假设”，即样本的各个属性之间相互独立。则$p(c_i|x)$计算方式如下：

$$p(c_i|x) = \frac{p(c_i)}{p(x)}\prod_{i=1}^{n} p(x_i|c_i)$$

其中，n 为属性数，x_i为 x 在第 i 个属性上的取值。

由于对于所有类别来说$p(x)$相同，因此朴素贝叶斯判定准则为

$$c_i = \underset{c_i \in y}{argmax}\ p(c_i)\prod_{i=1}^{n} p(x_i|c_i)$$

朴素贝叶斯模型的多种形式

①多项式模型：

特征：单词，值：k 类单词出现频次。

在多项分布朴素贝叶斯模型中，特征向量 x 的特征通常为离散型变量，并且假定所有特征的取值是符合多项分布的，可用于文本分类。在多项式模型中，设某文档 d=$(t_1, t_2, \ldots, t_k)$，t_i是该文档中出现过的单词，允许重复，则先验概率：

$p(c_i)$=类c_i下单词总数/整个训练样本的单词总数

类条件概率：

$p(t_k|c_i)$=(类c_i下单词t_i在各个文档中出现过的次数之和+1)/(类c_i下单词总数+|V|)

其中，V 是训练样本的单词表（即抽取单词，单词出现多次，只算一个），|V|则表示训练样本包含多少种单词。$p(t_i|c_i)$可以看作是单词t_i在证明 d 属于类c_i上提供了多大的证据，而$p(c_i)$则可以认为是类别c_i在整体上占多大比例(有多大可能性)。

②伯努利模型

特征：文本，值：k 类文本出现频次。

在伯努利朴素贝叶斯模型中，每个特征的取值是布尔型，或以 0 和 1 表示，所以伯努利模型中，每个特征值为 0 或者 1。计算方式:

$p(c_i)$= 类c_i下文件总数/整个训练样本的文件总数

$p(t_k|c_i)$=(类c_i下包含单词t_i的文件数+1)/(类c_i的文档总数+2)

③高斯模型

特点：只有它适用于连续变量预测（如身高预测）。

在高斯朴素贝叶斯模型中，特征向量 x 的特征通常为连续型变量，并且假定所有特征的取值是符合高斯分布的，即:

$$p(x_i|c=c_i)=\frac{1}{\sqrt{2\pi\sigma_{ik}}}\exp\left(-\frac{(x_i-\mu_{ik})^2}{2\sigma_{ik}{}^2}\right)$$

其中，μ_{ik}，σ_{ik}分别为每一维特征的均值和方差。

朴素贝叶斯算法描述

1）在已知标记的训练集$[x^{(i)}, y^{(i)}]$，其中$y=\{c_1, c_2, \dots, c_k\}$表示 k 种可能的类别标记，在训练集上求得类先验概率$p(c_i)$;

2）设$x=[x_1, x_2, \dots, x_n]$为一个待分类项统计，每个x_i为 x 的一个特征属性，求得在各类别下各个特征属性的条件概率$p(x_i|c_i)$

3）对每个类别c_i计算 $p(c_i)\prod_{i=1}^{n}p(x_i|c_i)$

4）确定 x 的分类结果$c_i = \underset{c_i \in y}{argmax}\ p(c_i)\prod_{i=1}^{n}p(x_i|c_i)$

朴素贝叶斯算法在模拟数据中的一般实现

1）生成已标记的数据集：

```
from sklearn.datasets import load_iris
iris=load_iris()
```

我们使用 sklearn 包下的 datasets 里 Iris 数据集。数据集里一共包括 150 行记录，其中前 4 列为花萼长度，花萼宽度，花瓣长度，花瓣宽度等 4 个用于识别鸢尾花的属性，第 5 列为鸢尾花的类别（包括 Setosa，Versicolour，Virginica 三类）。也即通过判定花萼长度，花萼宽度，花瓣长度，花瓣宽度的尺寸大小来识别鸢尾花的类别。

2）尝试使用 3 种不同类型的朴素贝叶斯模型来预测类别：

①高斯分布型

```
from sklearn.naive_bayes import GaussianNB #高斯分布型
gnb=GaussianNB() #构造
pred=gnb.fit(iris.data,iris.target) #拟合
y_pred=pred.predict(iris.data) #预测
```

②伯努利型

```
from sklearn.naive_bayes import BernoulliNB #伯努利型
gnb=BernoulliNB()
pred=gnb.fit(iris.data,iris.target)
y_pred=pred.predict(iris.data)
```

③多项式型

```
from sklearn.naive_bayes import MultinomialNB #多项式型
gnb=MultinomialNB()
pred=gnb.fit(iris.data,iris.target)
y_pred=pred.predict(iris.data)
```

iris.target 是科学家给出的分类，y_pred 是朴素贝叶斯模型产生的预测，最后用 print 将不同值的个数求出来，高斯分布型结果是 150，6；伯努利型结果是 150，100，说明此模型不合适用来预测 Iris 数据花的种类；多项式型结果是 150，7。

```
print(iris.data.shape[0],(iris.target !=y_pred).sum())
```

3）对模型进行验证：

使用 sklearn.model_selection.cross_val_score()，对模型进行验证：将数据集分为 10 份，其中 9 份作为训练模型，1 份用来做评估，score 是交叉验证的对象，结果返回准确率。

① 高斯分布型

```
from sklearn.naive_bayes import GaussianNB
gnb=GaussianNB()
scores=cross_val_score(gnb,iris.data,iris.target,cv=10)
print("Accuracy:%.3f"%scores.mean())
```

结果：Accuracy:0.953

② 伯努利型

```
from sklearn.naive_bayes import BernoulliNB
from sklearn.model_selection import cross_val_score
gnb=BernoulliNB()
scores=cross_val_score(gnb,iris.data,iris.target,cv=10) #将数据集分为10份，其中
                                                        #score是交叉验证的对象
                                                        #结果是返回准确率的概念
print("Accuracy:%.3f"%scores.mean())
```

结果：Accuracy:0.333

③ 多项式型

```
from sklearn.naive_bayes import MultinomialNB
gnb=MultinomialNB()
scores=cross_val_score(gnb,iris.data,iris.target,cv=10)
print("Accuracy:%.3f"%scores.mean())
```

结果：Accuracy:0.953