1.朴素贝叶斯简介

朴素贝叶斯(Naive Bayesian)算法能够根据数据加先验概率来估计后验概率，在垃圾邮件分类、文本分类、信用等级评定等多分类问题中得到广泛应用。对于多数的分类算法，比如决策树、KNN等，他们都是判别方法，也就是直接学习出特征输出Y和特征X之间的关系。但朴素贝叶斯和多数分类算法都不同，朴素贝叶斯是生成算法，也就是先找出特征输出Y和特征X的联合分布$P(X,Y)$，然后用$P(Y|X)=\frac{P(X,Y)}{P(X)}$得出。

朴素贝叶斯算法的优点在于简单易懂、学习效率高，在某些领域的分类问题中能够与决策树相媲美。但朴素贝叶斯算法以自变量之间的独立性和连续变量的正态性假设为前提，会导致算法精度在一定程度上受到影响。

2.朴素贝叶斯算法模型

2.1统计知识回顾

深入算法原理之前，我们先来回顾下统计学的相关知识。

- **条件概率公式**

$$P(X,Y)=P(X)P(Y)\ X、Y_{相互独立}$$

- **条件概率公式**

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

- **全概率公式**

$$P(X)=\sum_j P(X|Y=Y_j)P(Y_j)_{\ 其中}\ \sum_j P(Y_j)=1$$

经过上面统计学知识，我们能够得出贝叶斯公式。

$$P(Y_k|X)=\frac{P(X,Y_k)}{P(X)}=\frac{P(X|Y_k)P(Y_k)}{\sum_{k=1}^{K}P(X|Y=Y_k)P(Y_k)}$$

2.2朴素贝叶斯模型

假设我们已经有部分数据，并且能从数据中得到先验概率，那么如何得到后验概率$P(Y_k|X)$呢？下面我们通过构建朴素贝叶斯分类模型来解决后验概率问题，假设分类模型数据有m个样本，每个样本有n个特征，特征输出有K个类别，定义为$C1,C2,\dots,Ck$。

$\{(x_1^{(1)},x_2^{(1)},\dots,x_n^{(1)},y_1),(x_1^{(2)},x_2^{(2)},\dots,x_n^{(2)},y_2),\dots,(x_1^{(m)},x_2^{(m)},\dots,x_n^{(m)},y_m)\}$

从样本中我们能够学习得到朴素贝叶斯的先验分布概率

$$P(Y=C_k)(k=1,2,\dots,K)$$

然后学习得到条件概率分布

$$P(X|Y=C_k)=P(X_1=x^{(1)},X_2=x^{(2)},\ldots,X_n=x^{(n)}|Y=C_k)$$

最后结合贝叶斯公式便可得到X和Y的联合分布$P(X,Y=C_k)$

$$P(X,Y=C_k)=P(Y=C_k)P(X|Y=C_k)\quad(1)$$

$$=P(Y=C_k)P(X_1=x^{(1)},X_2=x^{(2)},\ldots,X_n=x^{(n)}|Y=C_k)\quad(2)$$

上式中$P(Y=C_k)$可以通过极大似然法求出，得到的$P(Y=C_k)$就是类别C_k在训练集里面出现的频数。但是$P(X_1=x^{(1)},X_2=x^{(2)},\ldots,X_n=x^{(n)}|Y=C_k)$很难求出，因此朴素贝叶斯模型做一个大胆的假设，即X的n个维度之间相互独立，这样就可以得出

$$P(X_1=x^{(1)},X_2=x^{(2)},\ldots,X_n=x^{(n)}|Y=C_k)$$

$$=P(X_1=x^{(1)}|Y=C_k)P(X_2=x^{(2)}|Y=C_k),\ldots,P(X_n=x^{(n)}|Y=C_k)$$

这样我们便得到X和Y的联合分布，最后的问题是给定测试集$(x_1^{(test)},x_2^{(test)},\ldots,x_n^{(test)})$，如何判断它属于哪个类型?

2.3朴素贝叶斯推断

假如我们预测的类别结果为C_{result}，其中C_{result}是使$P(Y=C_k|X=X^{(test)})$最大的类别，数学表达式为

$$C_{result}=\underbrace{\arg\max}_{C_k}P(Y=C_k|X=X^{(test)})$$

$$=\underbrace{\arg\max}_{C_k}\frac{P(X=X^{(test)}|Y=C_k)P(Y=C_k)}{P(X=X^{(test)})}$$

由于对所有类别计算$P(Y=C_k|X=X^{(test)})$时，分母都是$P(X=X^{(test)})$，因此我们的预测公式可以简化为

$$C_{result}=\underbrace{\arg\max}_{C_k}P(X=X^{(test)}|Y=C_k)P(Y=C_k)$$

然后利用朴素贝叶斯的独立性假设，就可以得到朴素贝叶斯推断公式

$$C_{result}=\underbrace{\arg\max}_{C_k}P(Y=C_k)\prod_{j=1}^{n}P(X_j=X_j^{(test)}|Y=C_k)$$

2.4朴素贝叶斯参数估计

对于**2.3**中朴素贝叶斯推断公式，我们只要求出$P(Y=C_k)$和
$P(X_j=X_j^{(test)}|Y=C_k)(j=1,2,\ldots,n)$，就可以得到预测结果。对于$P(Y=C_k)$比较简单，通过极大似然估计能够得到$C_k$出现的概率，也就是样本类别$C_k$出现的次数$m_k$除以样本总数m。而对于$P(X_j=X_j^{(test)}|Y=C_k)(j=1,2,\ldots,n)$则取决于我们的先验条件。

- 如果X_j是**离散值**，那么可以假设X_j服从多项式分布，这样得到的$P(X_j = X_j^{(test)}|Y = C_k)$是在样本类别中$X_j^{test}$出现的频率，公式如下所示。其中$m_k$为样本类别$C_k$出现的次数，而$m_{kj}^{(test)}$为类别是$C_k$的样本中，第j维特征$X_j^{(test)}$出现的次数。

$$P(X_j = X_j^{(test)}|Y = C_k) = \frac{m_{kj}^{(test)}}{m_k}$$

  某些时候可能一些类别在样本中从来没有出现，这样可能导致$P(X_j = X_j^{(test)}|Y = C_k)$为0，因此会影响后验概率的估计。为了解决此类情况，我们引入拉普拉斯平滑，公式如下所示。其中λ为大于0的常数，常取为1，O_j为第j个特征的取值个数。

$$P(X_j = X_j^{(test)}|Y = C_k) = \frac{m_{kj}^{(test)} + \lambda}{m_k + O_j\lambda}$$

- 如果X_j是**稀疏的离散值**，即各个特征的出现概率很低，那么可以假设X_j服从伯努利分布，即特征X_j出现记为1，不出现记为0。即我们只关注X_j是否出现，不关注X_j出现的次数，这样得到的$P(X_j = X_j^{(test)}|Y = C_k)$是在样本类别$C_k$中$X_j^{(test)}$出现的频率，公式如下所示。其中$X_j^{(test)}$取值为0和1。

$$P(X_j = X_j^{(test)}|Y = C_k) = P(X_j|Y = C_k)X_j^{(test)} + (1 - P(X_j|Y = C_k))(1 - X_j^{(test)})$$

- 如果X_j是**连续值**，那么假设X_j的先验概率为高斯分布(正态分布)，这样假设$P(X_j = X_j^{(test)}|Y = C_k)$的概率分布公式如下所示。其中$\mu_k$和$\sigma_k^2$是正态分布的期望和方差，$\mu_k$为样本类别$C_k$中，所有$X_j$的平均值，$\sigma_k^2$为样本类别$C_k$中，所有$X_j$的方差，$\mu_k$和$\sigma_k^2$可以通过极大似然估计求得。

$$P(X_j = X_j^{(test)}|Y = C_k) = \frac{1}{\sqrt{2\pi\sigma^2}}exp(-\frac{(X_j^{(test)} - \mu_k)^2}{2\sigma_k^2})$$

3.朴素贝叶斯算法流程

我们总结下朴素贝叶斯算法流程，假设训练集有m个样本n个维度，训练样本有K个特征输出类别，分别为$C1, C2, \ldots, C_K$，每个特征输出类别的样本个数为$m_1, m_2, \ldots, m_K$。第K个类别中，如果是离散特征，则特征X_j各个类别取值为m_{jl}，其中l的取值为$1, 2, \ldots, S_j$，S_j为特征j不同的取值数。

$$\{(x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}, y_1), (x_1^{(2)}, x_2^{(2)}, \ldots, x_n^{(2)}, y_2), \ldots, (x_1^{(m)}, x_2^{(m)}, \ldots, x_n^{(m)}, y_m)\}$$

预测结果为$X^{(test)}$的分类，算法流程如下所示

- 如果没有Y的先验概率，则计算Y的K个先验概率$P(Y = C_k) = \frac{m_k}{m}$。
- 分别计算第k个类别的第j为特征的第l个取值条件概率$P(X_j = x_{jl}|Y = C_k)$。
  - 如果是离散值，则公式如下所示，其中λ为大于0的常数，常常取为1，O_j为第j个特征的取值个数。

$$P(X_j = x_{jl}|Y = C_k) = \frac{x_{jl} + \lambda}{m_k + O_j\lambda}$$

- 如果是稀疏离散值，则公式如下所示，此时l取值为0或1。

$$P(X_j = x_{jl}|Y = C_k) = P(X_j|Y = C_k)x_{jl} + (1 - P(X_j|Y = C_k))(1 - x_{jl})$$

- 如果是连续值，则不需要计算各个l个取值概率，直接求正态分布的参数，公式如下所示。μ_k为样本类别C_k中所有X_j的平均值，σ_k^2为样本类别C_k中所有X_j的方差。

$$P(X_j = x_j|Y = C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} exp(-\frac{(x_j - \mu_k)^2}{2\sigma_k^2})$$

- 对于实例$X^{(test)}$，分别计算

$$P(Y = C_k) \prod_{j=1}^{n} P(X_j = X_j^{(test)}|Y = C_k)$$

- 确定实例$X^{(test)}$的分类C_{result}

$$C_{result} = \underbrace{\arg\max}_{C_k} P(Y = C_k) \prod_{j=1}^{n} P(X_j = X_j^{(test)}|Y = C_k)$$

从上面计算可以看出，朴素贝叶斯没有复杂的求导和矩阵运算，因此效率很高。但朴素贝叶斯假设数据特征之间相互独立，如果数据特征之间关联性比较强的话，我们尽量不要使用朴素贝叶斯算法，考虑其他分类方法比较好。

4.Sklearn实现朴素贝叶斯

利用sklearn自带的iris数据集进行训练，选取70%的数据当作训练集，30%的数据当作测试集。因iris数据集为连续值，所以采用GaussianNB模型，训练后模型得分为0.933333333333。更多关于sklearn.naive_bayes的使用技巧可以访问官方教程。

```
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

#load data
iris=load_iris()
X=iris.data
y=iris.target
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_sta
te=1)

mnb=GaussianNB()
mnb.fit(X_train,y_train)

print(mnb.predict(X_test))
# [0 1 1 0 2 2 2 0 0 2 1 0 2 1 1 0 1 1 0 0 1 1 2 0 2 1 0 0 1 2 1 2 1 2 2 0
1
#  0 1 2 2 0 1 2 1]
print(y_test)
```

```
# [0 1 1 0 2 1 2 0 0 2 1 0 2 1 1 0 1 1 0 0 1 1 1 0 2 1 0 0 1 2 1 2 1 2 2 0
1
#  0 1 2 2 0 2 2 1]
print(mnb.score(X_test,y_test))
# 0.933333333333
```

5.朴素贝叶斯优缺点

5.1优点

- 具有稳定的分类效率。
- 对缺失数据不敏感，算法也比较简单。
- 对小规模数据表现良好，能处理多分类任务，适合增量式训练。尤其是数据量超出内存后，我们可以一批批的去增量训练。

5.2缺点

- 对输入数据的表达形式比较敏感，需针对不同类型数据采用不同模型。
- 由于我们是使用数据加先验概率预测后验概率，所以分类决策存在一定的错误率。
- 假设各特征之间相互独立，但实际生活中往往不成立，因此对特征个数比较多或特征之间相关性比较大的数据来说，分类效果可能不是太好。

6.推广

更多内容请关注公众号**谓之小一**，若有疑问可在公众号后台提问，随时回答，欢迎关注，内容转载请注明出处。

长按关注微信公众号

由锤子便签发送 via Smartisan Notes

参考

> 刘建平_Pinard-朴素贝叶斯算法原理小结