# 山东大学计算机科学与技术学院

# 数据挖掘实验报告

| 实验题目： Homework 1: 朴素贝叶斯分类器 | | 学号： 201834882 |
|---|---|---|
| 日期：2018.11.23 | 班级： 18 级专硕班 | 姓名： 谢升军 |
| Email：1640893020@qq.com | | |

实验目的：实现朴素贝叶斯分类器，测试其在 20 Newsgroups 数据集上的效果

硬件环境：win10 系统的 8g 内存计算机

软件环境：Anaconda3，Spyder

实验步骤与内容：

## 一、 整体概述

### 1. 贝叶斯定理

如果有两个事件，事件 A 和事件 B。已知事件 A 发生的概率为 p(A)，事件 B 发生的概率为 P(B)，事件 A 发生的前提下。事件 B 发生的概率为 p(B|A)，事件 B 发生的前提下。事件 A 发生的概率为 p(A|B)，事件 A 和事件 B 同一时候发生的概率是 p(AB)。则有

$$p(AB)=p(A)p(B|A)=p(B)p(A|B)$$

依据式(1)能够推出贝叶斯定理为

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

给定一个全集{B1,B1,…,Bn}，当中 Bi 与 Bj 是不相交的，即 BiBj=∅。则依据全概率公式。对于一个事件 A。会有

$$p(A) = \sum_{i=1}^{n} p(B_i)p(A|B_i)$$

则广义的贝叶斯定理有

$$p(B_i|A) = \frac{p(B_i)p(A|B_i)}{\sum_{i=1}^{n} p(B_i)p(A|B_i)}$$

## 2. 朴素贝叶斯基本原理

给定一组训练数据集{(X1,y1),(X2,y2),(X3,y3),…,(Xm,ym)}。当中，m 是样本的个数。每个数据集包括着 n 个特征，即 Xi=(xi1,xi2,…,xin)。类标记集合为{y1,y2,…,yk}。设 p(y=yi|X=x)表示输入的 X 样本为 x 时，输出的 y 为 yk 的概率。

如果如今给定一个新的样本 x。要推断其属于哪一类，可分别求解 p(y=y1|x)，p(y=y2|x)，p(y=y3|x)，...，p(y=yk|x)的值。哪一个值最大，就属于那一类。即，求解最大的后验概率 argmaxp(y|x)。

那怎样求解出这些后验概率呢？依据贝叶斯定理。有

$$p(y = y_i|x) = \frac{p(y_i)p(x|y_i)}{p(x)}$$

一般地，朴素贝叶斯方法如果各个特征之间是相互独立的，则式(5)能够写成：

$$p(y = y_i|x) = \frac{p(y_i)p(x|y_i)}{p(x)} = \frac{p(y_i)\prod_{j=1}^{n} p(x_j|y_i)}{\prod_{j=1}^{n} p(x_j)}$$

由于(6)式的分母。对于每个 p(y=yi|x)求解都是一样的。所以，在实际操作中。能够省略掉。终于。朴素贝叶斯分类器的判别公式变成例如以下的形式：

$$y = \arg \max_{y_i} p(y_i)p(x|y_i) = \arg \max_{y_i} p(y_i)\prod_{j=1}^{n} p(x_j|y_i)$$

以下，是怎样通过样本对 p(y) 和 p(x|y) 进行概率预计。

**3.贝叶斯分类器的多项式模型**

- 多项式模型：
  - 重复的词语我们视为其出现多次

$$P(\text{“代开”},\text{“发票”},\text{“增值税”},\text{“发票”},\text{“正规”},\text{“发票”})|S)$$
$$= P(\text{“代开”}"|S)P(\text{“发票”}|S)P(\text{“增值税”}|S)P(\text{“发票”}|S)P(\text{“正规”}|S)P(\text{“发票”}|S)$$
$$= P(\text{“代开”}"|S)P^3(\text{“发票”}|S)P(\text{“增值税”}|S)P(\text{“正规”}|S)$$

注意这一项:$P^3(\text{“发票”}|S)$。

  - 统计与判断时，都关注重复次数。

**4.多项式模型的平滑方法**

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c)}{\sum_{w \in V} \left(count(w, c)\right)}$$

$$= \frac{count(w_i, c)+1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

## 二、 具体实现

由于贝叶斯分类器和上一次的 VSM 模型都需要分实验数据和测试数据、统计词频信息、读取测试文件，所以大部分的方法都可以由上一次实验的方法改写

### 1. 计算每个类的单词词频

```
def createdicount(sampleFilesDir): #计算每个类中的单词的词频
    n = 0
    wordMap = {}     #存储词频
    sampleList = listdir(sampleFilesDir)
    dfcount = len(sampleList)
    for j in range(len(sampleList)):
            sampleDir = sampleFilesDir + '/' + sampleList[j]
            temp = open(sampleDir).readlines()
            tempdic = Counter(temp)  #调用counter函数计算文件单词的词频
            for key,value in tempdic.items():
                key = key.strip('\n')   #去除空格
                wordMap[key] = wordMap.get(key,0) + value  #计算词频
                n = n + value
    return wordMap, n ,dfcount
```

此方法用于求某个类中的单词和此单词的词频放到 wordmap 中去，n 表示此类中的单词总数，dfcount 表示此类文件中文件总数。

## 2. 计算每个类中单词出现的概率

```
def creatall():
    diccount = []  #存20个类的词典和词频，大小为20
    count = []    #存每个类的单词总数
    lendic = []   #存每个类词典的单词数量
    dcount = []  # 存每个类的文件总数
    p = []  #存储计算好的概率
    nll=[]  #存储没有单词的概率

    fileDir = targettrain
    sampleFilesList = listdir(fileDir)
    for i in range(len(sampleFilesList)):
        sampleFilesDir = fileDir + '/' + sampleFilesList[i]
        a,b,c = creatediccount(sampleFilesDir)
        diccount.append(a)
        count.append(b)
        dcount.append(c)
        lendic.append(len(a))
    for i in range(len(count)):
        temp = { }
        for key,value in diccount[i].items():
            temp[key] = ((value+1)/(count[i]+lendic[i]))*10**4 #用的多项式模型，用的多
        p.append(temp)
        nll.append((1/(count[i]+lendic[i]))*10**4)

    return p,nll,dcount
```

此方法用贝叶斯分类器的多项式模型计算每个单词出现的概率，并用了多项式模型的平滑技术，由于计算出的概率太小，我在这给每个概率乘上了 10 的四次方。

## 3. Bays 算法

```
def bayes():
    p,nll,dcount = creatall()
    classname = listdir(targettrain)
    docusum = 0
    for i in range(len(dcount)):
        docusum = docusum + dcount[i]

    test,testfrom = opentest(targettest)
    #print(test[len(test)-1])
    result = []
    temp = []

    for i in range(len(test)):
        temp = []
        for j in range(len(p)):
            s = 1
            for k in range(len(test[i])):
               if test[i][k] in p[j].keys(): #如果此类中有
                        s = s*p[j][test[i][k]]
               else:
                    s =s*nll[j]

            s = s*(dcount[j]/docusum)
            temp.append(s)
        result.append(classname[temp.index(max(temp))])
```

此方法用于计算测试文件在每个类中的概率，并取概率最大的类作为它的分类。

## 三、 实验结果

此实验最后的正确率为

```
201834882xieshengjun/实验二朴素贝叶
斯算法/bayes.py', wdir='E:/GitHub/
201834882xieshengjun/实验二朴素贝叶
斯算法')
0.8468838975007908
```

**结论分析与体会：**

有了上一次实验 VSM 模型的基础，这次实验做起来并不困难，代码量也不大，方法基本是由 Vsm 模型的方法改写，最主要的是理解了贝叶斯分类器的原理，贝叶斯分类器的效率很高，运行时间也很快，最后的正确率和 Knn 方法相似。