**朴素贝叶斯**

有时候分类器会产生错误的结果，所以要求分类器给出一个最优的类别猜测结果，同时给出这个猜测的概率估计值。

朴素贝叶斯算法是有监督的学习算法，解决的是分类问题，如客户是否流失等多分类问题.

该算法的优点在于简单易懂、学习效率高、在数据较少的情况下，仍然有效，可以处理多类别问题.

缺点是：对于输入数据的准备方式较为敏感。

适用的数据类型：标称型数据。

该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，会导致算法精度在某种程度上受影响。

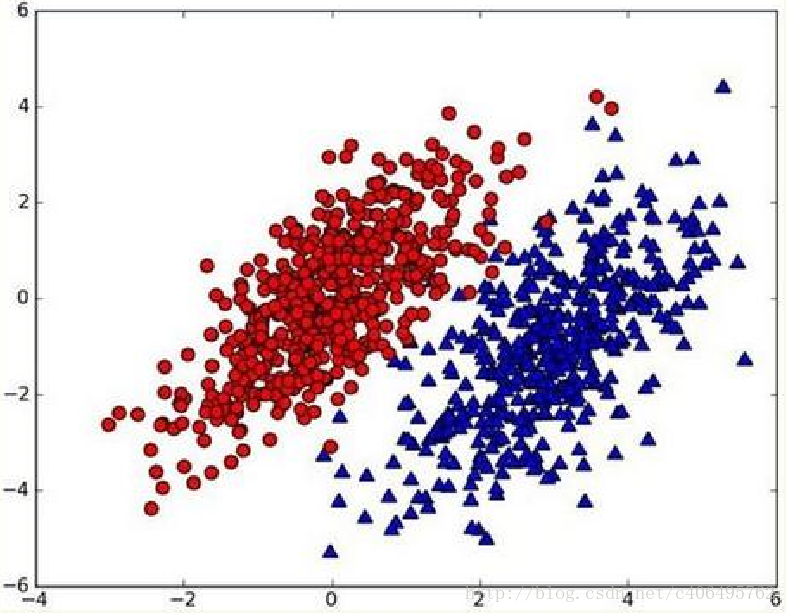
**朴素贝叶斯理论**

 朴素贝叶斯是贝叶斯决策理论的一部分，所以，首先了解一下贝叶斯决策理论。

**贝叶斯决策理论**

假设现在有一个数据集，由两类数据组成，数据分布如下图所示：

现在用p1(x,y)表示数据点(x,y)属于类别1的概率(图中红色圆点表示的类别)，用p2(x,y)表示数据点(x,y)属于类别2的概率(图中蓝色三角形表示的类别)，那么对于一个新数据点(x,y)，可以用下面的规则来判断它的类别：



如果p1(x,y) > p2(x,y)，那么类别为1

如果p1(x,y) < p2(x,y)，那么类别为2

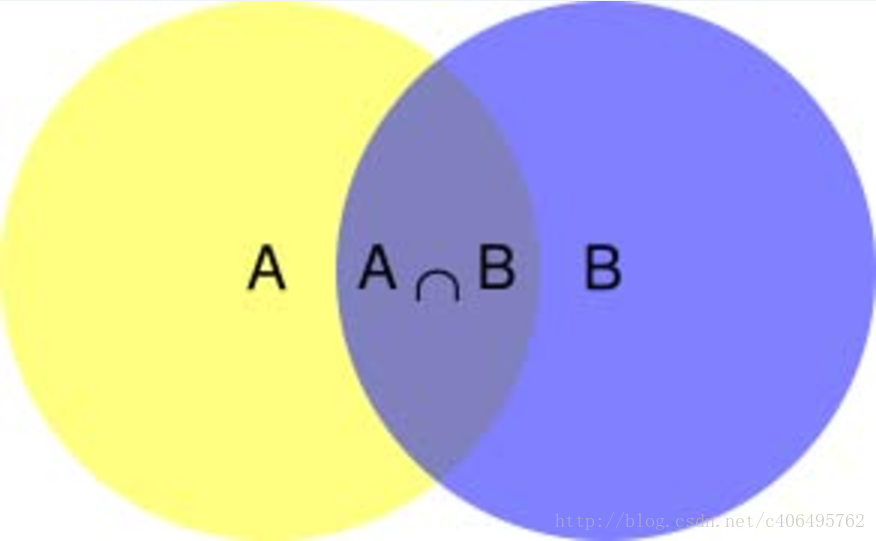
也就是说，我们会选择高概率对应的类别。

这就是贝叶斯决策理论的核心思想，即选择具有最高概率的决策。

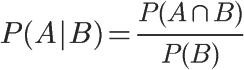
**那么如何计算p1和p2概率？**

**条件概率**

是指在事件B发生的情况下，事件A发生的概率，用P(A|B)来表示。



根据文氏图，看到在事件B发生的情况下，事件A发生的概率就是P(A∩B)除以P(B)。



因此，

IMG_257

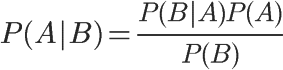
同理可得，

IMG_258

所以，

IMG_259

即

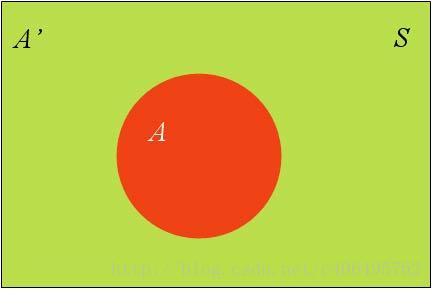


这就是条件概率的计算公式。

**全概率公式**

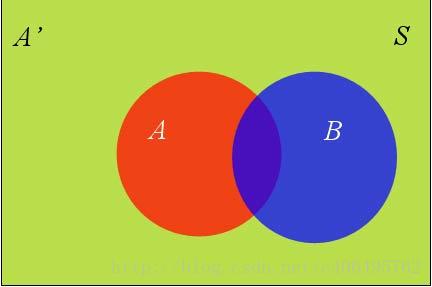
在计算p1和p2的时候，还要用到全概率公式，因此，这里继续推导全概率公式。

假定样本空间S，是两个事件A与A’的和



上图中，红色部分是事件A，绿色部分是事件A’，它们共同构成了样本空间S。

在这种情况下，事件B可以划分成两个部分。



即

IMG_257

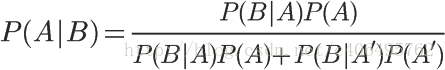
所以，

IMG_259

这就是全概率公式。

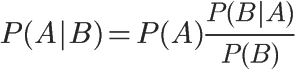
它的含义是，如果A和A’构成样本空间的一个划分，那么事件B的概率，就等于A和A’的概率分别乘以B对这两个事件的条件概率之和。

将这个公式代入上一节的条件概率公式，就得到了条件概率的另一种写法：



**贝叶斯推断**

对条件概率公式进行变形，可以得到如下形式：



P(A)称为"先验概率"，即在B事件发生之前，我们对A事件概率的一个判断。

P(A|B)称为"后验概率"，即在B事件发生之后，我们对A事件概率的重新评估。

P(B|A)/P(B)称为"可能性函数"，这是一个调整因子，使得预估概率更接近真实概率。

**所以，条件概率可以理解成下面的式子：**

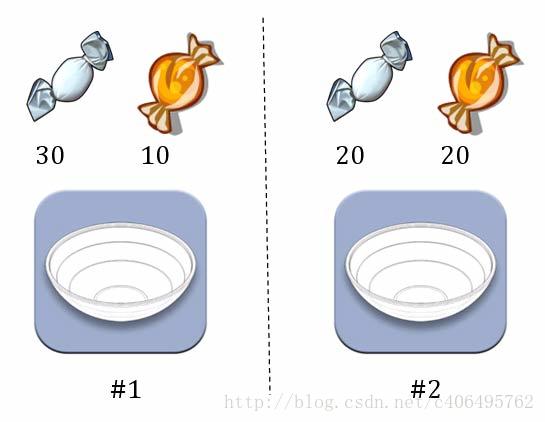
后验概率　＝　先验概率 ｘ 调整因子

这就是贝叶斯推断的含义。

我们先预估一个"先验概率"，然后加入实验结果，看这个实验到底是增强还是削弱了"先验概率"，由此得到更接近事实的"后验概率"。

如果"可能性函数"P(B|A)/P(B)>1，意味着"先验概率"被增强，事件A的发生的可能性变大；如果"可能性函数"=1，意味着B事件无助于判断事件A的可能性；如果"可能性函数"<1，意味着"先验概率"被削弱，事件A的可能性变小。

举个例子

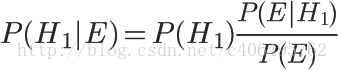


两个一模一样的碗，一号碗有30颗水果糖和10颗巧克力糖，二号碗有水果糖和巧克力糖各20颗。现在随机选择一个碗，从中摸出一颗糖，发现是水果糖。请问这颗水果糖来自一号碗的概率有多大？

H1表示一号碗，H2表示二号碗。由于这两个碗是一样的，所以P(H1)=P(H2)，P(H1)=0.5，把这个概率就叫做"先验概率"，即没有做实验之前，来自一号碗的概率是0.5。

再假定，E表示水果糖，所以问题就变成了在已知E的情况下，来自一号碗的概率有多大，即求P(H1|E)。我们把这个概率叫做"后验概率"，即在E事件发生之后，对P(H1)的修正。

根据条件概率公式，得到



P(E|H1)为一号碗中取出水果糖的概率，等于30÷(30+10)=0.75

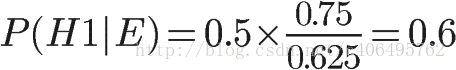
根据全概率公式，

IMG_264

所以，

IMG_265

将数字代入原方程，得到



这表明，来自一号碗的概率是0.6。也就是说，取出水果糖之后，H1事件的可能性得到了增强。

有时候为了减少计算量，全概率公式在实际编程中可以不使用。

两者的分母都是相同的，那我们只需要比较分子即可。即比较P(E|H1)P(H1)和P(E|H2)P(H2)的大小。

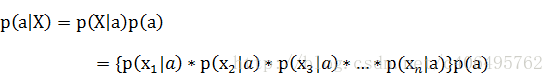
**朴素贝叶斯推断**

贝叶斯和朴素贝叶斯的概念是不同的，区别就在于“朴素”二字，朴素贝叶斯对条件个概率分布做了条件独立性的假设。

比如下面的公式，假设有n个特征：

IMG_256

由于每个特征都是独立的，我们可以进一步拆分公式



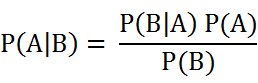
**举个例子**

某个医院早上来了六个门诊的病人，他们的情况如下表所示：

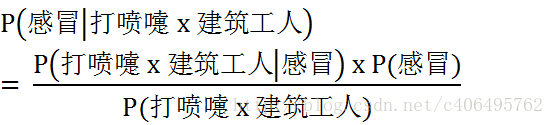


现在又来了第七个病人，是一个打喷嚏的建筑工人。请问他患上感冒的概率有多大？

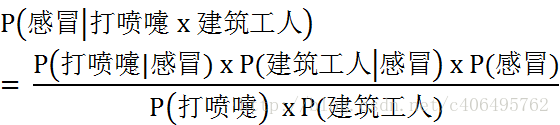
根据贝叶斯定理：



可得：



根据朴素贝叶斯条件独立性的假设可知，"打喷嚏"和"建筑工人"这两个特征是独立的，因此，上面的等式就变成了



计算：

IMG_259

因此，这个打喷嚏的建筑工人，有66%的概率是得了感冒。

同理，可以计算这个病人患上过敏或脑震荡的概率。比较这几个概率，就可以知道他最可能得什么病。

这就是贝叶斯分类器的基本方法：在统计资料的基础上，依据某些特征，计算各个类别的概率，从而实现分类。

**NBM.Py**

**动手实战--在线社区留言为例**

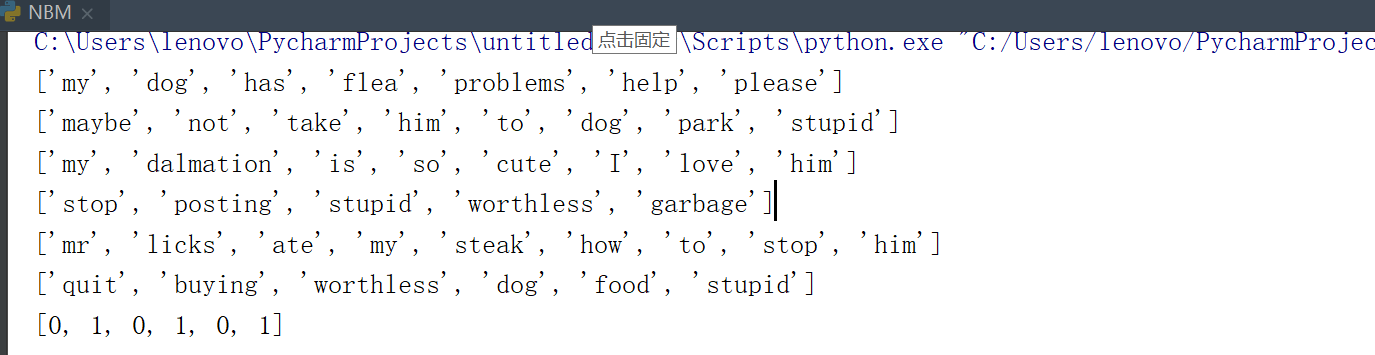
为了不影响社区的发展，要屏蔽侮辱性的言论，要构建一个快速过滤器，如果某条留言使用了负面或者侮辱性的语言，那么就将该留言标志为内容不当。

对此问题建立两个类型：侮辱类和非侮辱类，使用1和0分别表示。

把文本看成单词向量或者词条向量，也就是说将句子转换为向量。

先假设已经将本文切分完毕，存放到列表中，并对词汇向量进行分类标注。

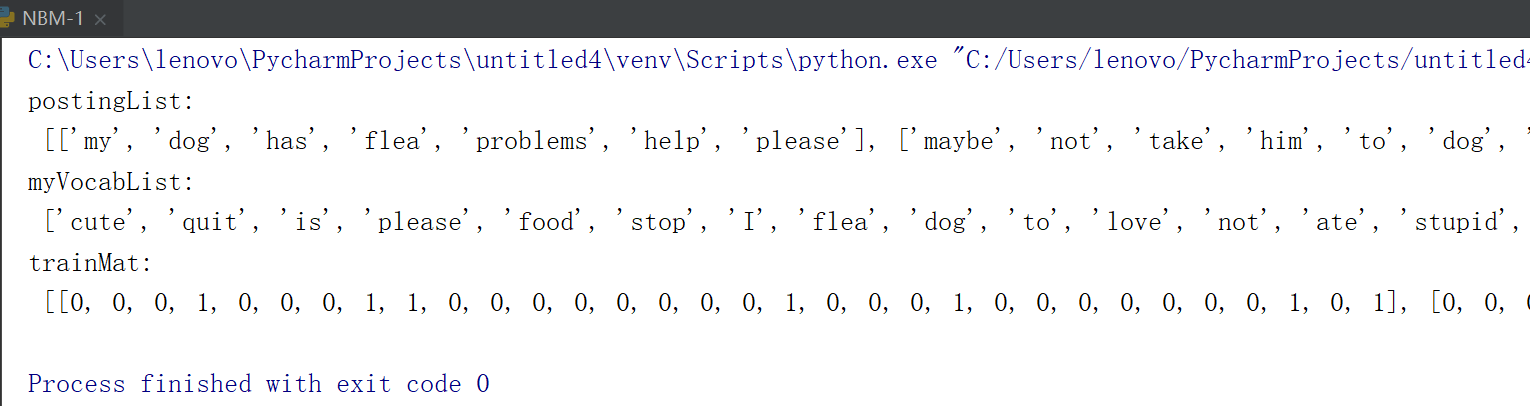
代码输出如下：



从运行结果可以看出，postingList是存放词条列表中，classVec是存放每个词条的所属类别，1代表侮辱类 ，0代表非侮辱类。

**NBM-1.Py**

继续编写代码，前面我们已经说过我们要先创建一个词汇表，并将切分好的词条转换为词条向量。



从运行结果可以看出，postingList是原始的词条列表，myVocabList是词汇表。myVocabList是所有单词出现的集合，没有重复的元素。

词汇表是用来将词条向量化的，一个单词在词汇表中出现过一次，那么就在相应位置记作1，如果没有出现就在相应位置记作0。

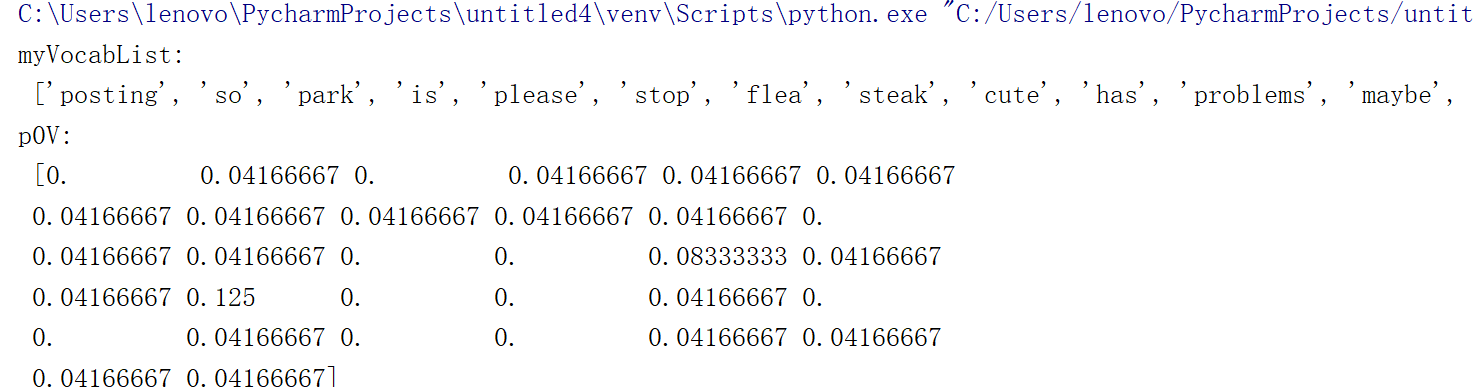
trainMat是所有的词条向量组成的列表。它里面存放的是根据myVocabList向量化的词条向量。

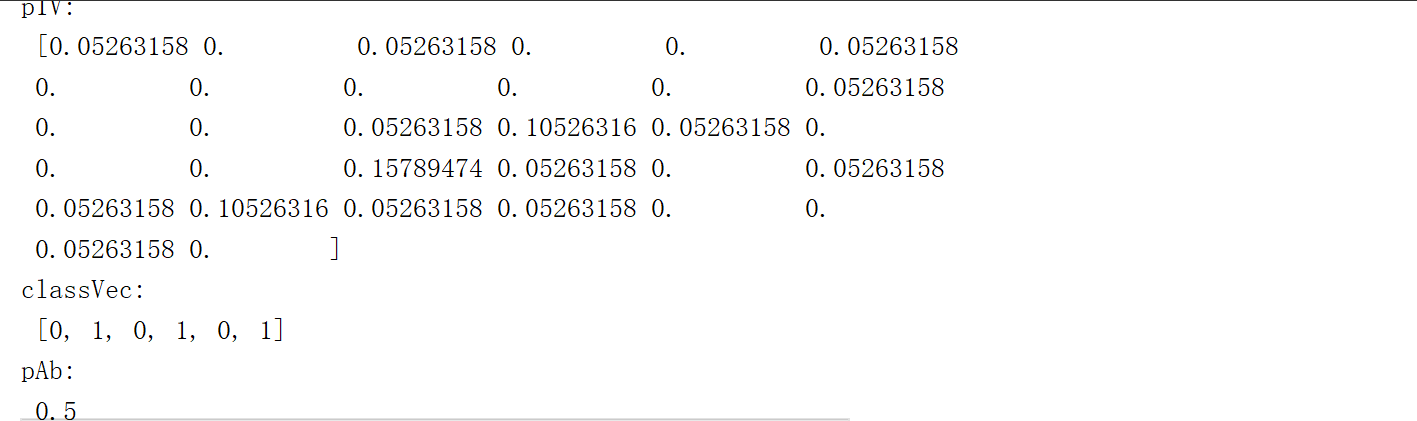
**NBM-3.Py**

我们已经得到了词条向量。

接下来，我们就可以通过词条向量训练朴素贝叶斯分类器。

代码输出如下：





运行结果如下：

p0V存放的是每个单词属于类别0，也就是非侮辱类词汇的概率。比如p0V的倒数第6个概率，就是stupid这个单词属于非侮辱类的概率为0。同理，p1V的倒数第6个概率，就是stupid这个单词属于侮辱类的概率为0.15789474，也就是约等于15.79%的概率。

pAb是所有侮辱类的样本占所有样本的概率，从classVec中可以看出，一用有3个侮辱类，3个非侮辱类。所以侮辱类的概率是0.5。

因此p0V存放的就是P(him|非侮辱类) = 0.0833、P(is|非侮辱类) = 0.0417，一直到P(dog|非侮辱类) = 0.0417，这些单词的条件概率。

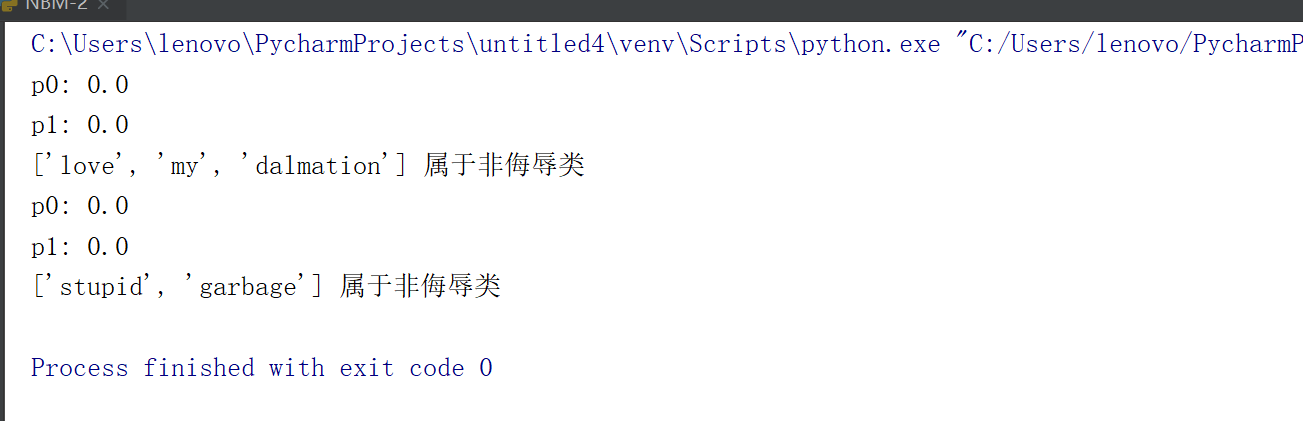
同理，p1V存放的就是各个单词属于侮辱类的条件概率。pAb就是先验概率。

**NBM-2.Py**

已经训练好分类器，接下来，使用分类器进行分类

我们测试了两个词条，在使用分类器前，也需要对词条向量化，然后使用classifyNB()函数，用朴素贝叶斯公式，计算词条向量属于侮辱类和非侮辱类的概率。

代码输出如下：



发现，这样写的算法无法进行分类，p0和p1的计算结果都是0，这里显然存在问题。

**总结**

朴素贝叶斯推断的一些优点：

生成式模型，通过计算概率来进行分类，可以用来处理多分类问题。

对小规模的数据表现很好，适合多分类任务，适合增量式训练，算法也比较简单。

朴素贝叶斯推断的一些缺点：

对输入数据的表达形式很敏感。

由于朴素贝叶斯的“朴素”特点，所以会带来一些准确率上的损失。

需要计算先验概率，分类决策存在错误率。

现在有一个问题：

有时候系统会以pytest的形式运行，想换回普通模式。

那么就点击Pytest in xxx.py上面的白色三角---点击Edit configurations--点击左上角的“-”---点击左上角的“+”--选择目录，添加文件名--点击“ok”就可以了。

**朴素贝叶斯改进之拉普拉斯平滑**

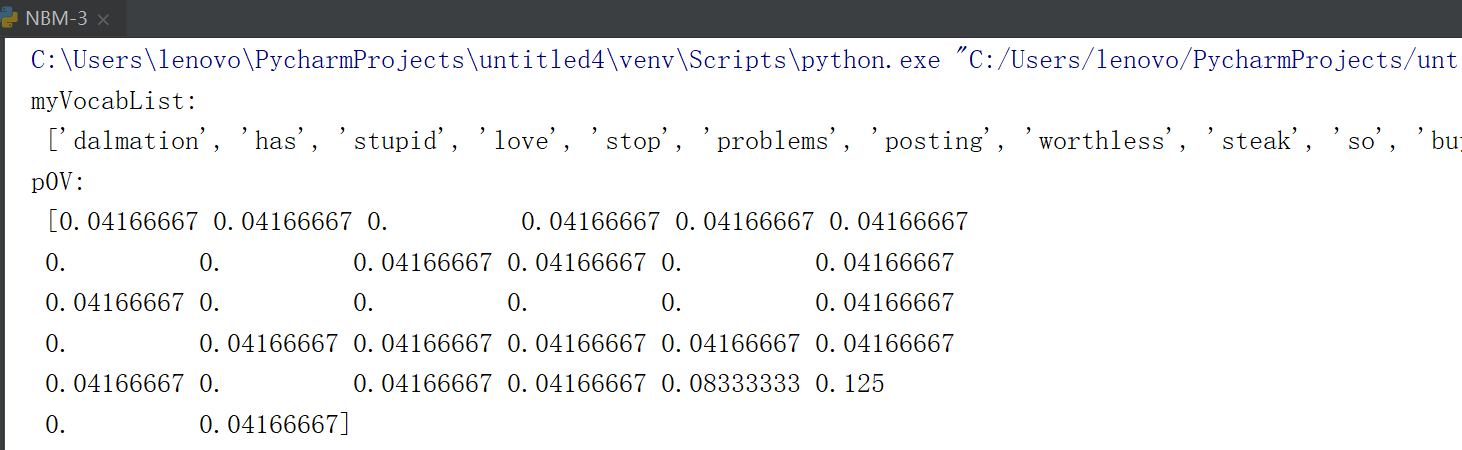
**NBM-4.Py**

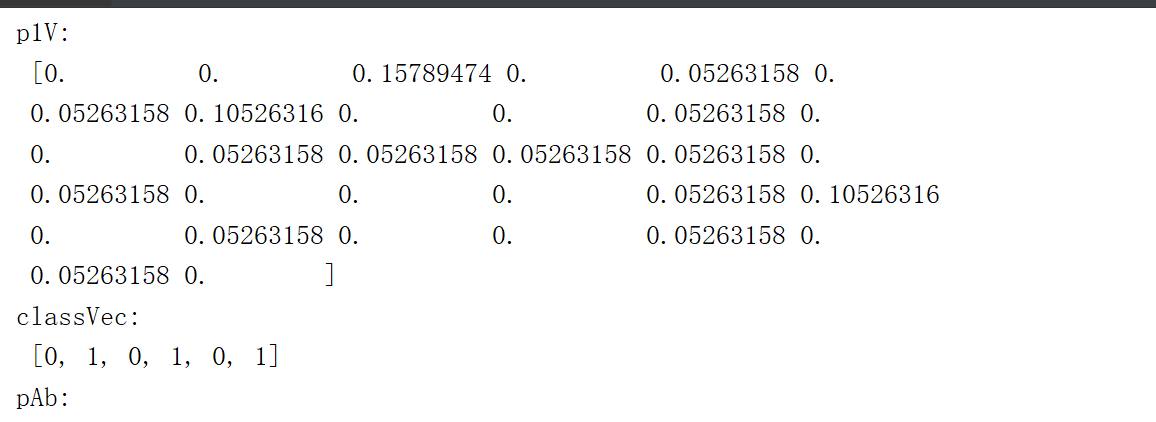
前面的算法存在一定的问题，需要进行改进。

利用贝叶斯分类器对文档进行分类时，要计算多个概率的乘积以获得文档属于某个类别的概率，即计算p(w0|1)p(w1|1)p(w2|1)。

如果其中有一个概率值为0，那么最后的成绩也为0。

这是上面程序运行的截图。





例如第三个Stupid，p0V中为0，p1V中为0.15789474.

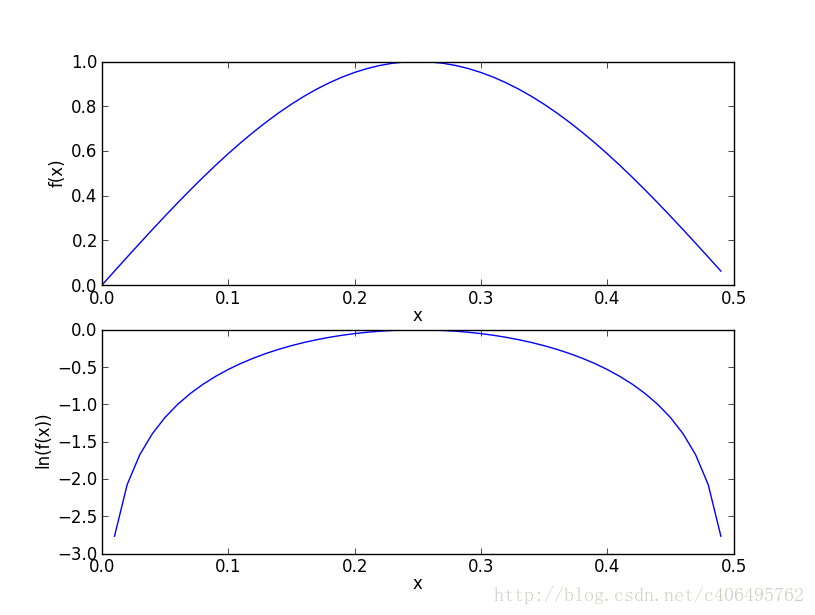
在计算的时候已经出现了概率为0的情况。如果新实例文本，包含这种概率为0的分词，那么最终的文本属于某个类别的概率也就是0了。显然，这样是不合理的，为了降低这种影响，可以将所有词的出现数初始化为1，并将分母初始化为2。

这种做法就叫做拉普拉斯平滑，又被称为加1平滑，是比较常用的平滑方法，它就是为了解决0概率问题。

除此之外，另外一个遇到的问题就是下溢出，这是由于太多很小的数相乘造成的。在程序中，在相应小数位置进行四舍五入，计算结果可能就变成0了。

为了解决这个问题，对乘积结果取自然对数。通过求对数可以避免下溢出或者浮点数舍入导致的错误。同时，采用自然对数进行处理不会有任何损失。

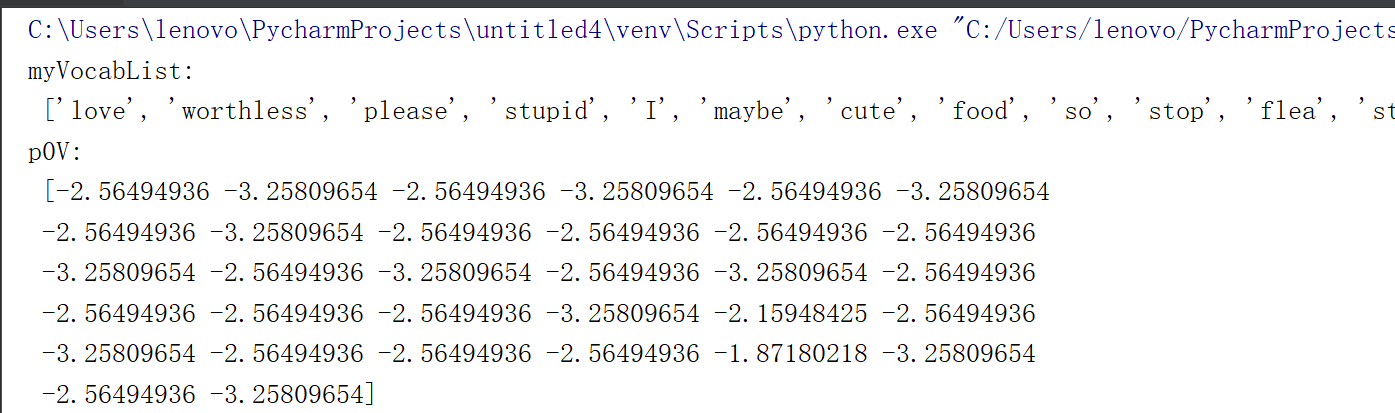
下图是函数f(x)和ln(f(x))的曲线。

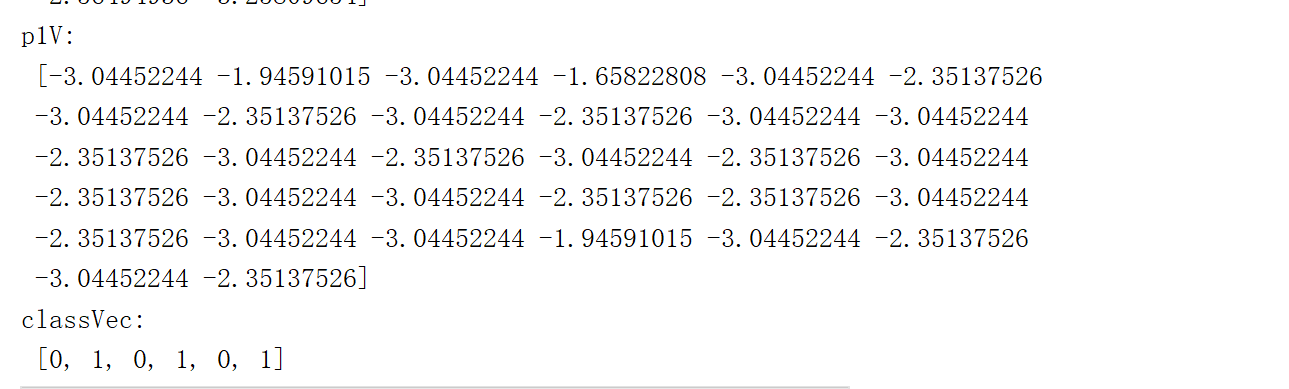


观察这两条曲线，发现它们在相同区域内同时增加或者减少，并且在相同点上取到极值。

它们的取值虽然不同，但不影响最终结果。因此对上面的的trainNB0函数进行更改。

更改后，代码输出如下：





这样得到的结果就没有问题了，不存在0概率。

这样，我们的朴素贝叶斯分类器就改进完毕了。

# **朴素贝叶斯之过滤垃圾邮件**

使用朴素贝叶斯解决一些现实生活中的问题时，需要先从文本内容得到字符串列表，然后生成词向量。

下面这个例子：电子邮件垃圾过滤。

首先看一下使用朴素贝叶斯对电子邮件进行分类的步骤：

收集数据：提供文本文件。

准备数据：将文本文件解析成词条向量。

分析数据：检查词条确保解析的正确性。

训练算法：使用建立的trainNB0()函数。

测试算法：使用classifyNB()，并构建一个新的测试函数来计算文档集的错误率。

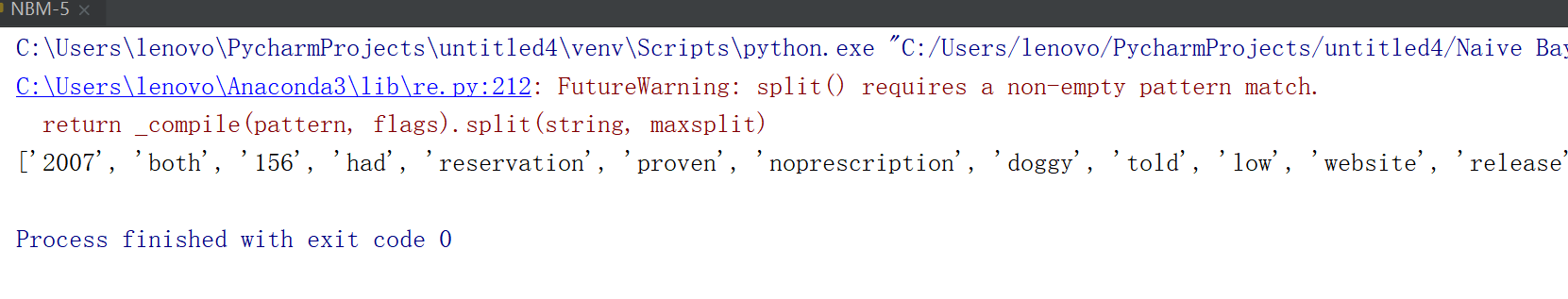
使用算法：构建一个完整的程序对一组文档进行分类，将错分的文档输出到屏幕上。

**NBM-5.Py**

**准备数据**

有两个文件夹ham和spam，spam文件下的txt文件为垃圾邮件。

对于英文文本，我们可以以非字母、非数字作为符号进行切分，使用split函数即可。

代码输出如下：

Error:UnicodeDecodeError: 'gbk' codec can't decode byte 0xae in position 199: illegal multibyte sequence

发现是文档里的“？”符号出错。根据decode byte 0xae in position 199看出来好像是文件中某个字节不能解码。

解决方法：

打开email\ham\23.txt，第三行，找到SciFinance ?，把?替换成空格即可。

Error:

发现上面也报错了，所以百度了一下：

因为使用\s\*模式时，要求在零个或多个空白的子字符串上拆分。

但是空字符串与该模式匹配，因为其中没有空格。

re.split决定放弃空的子字符串选项，而是在一个或多个空白上拆分。

你可以用+代替\*来表示你自己：

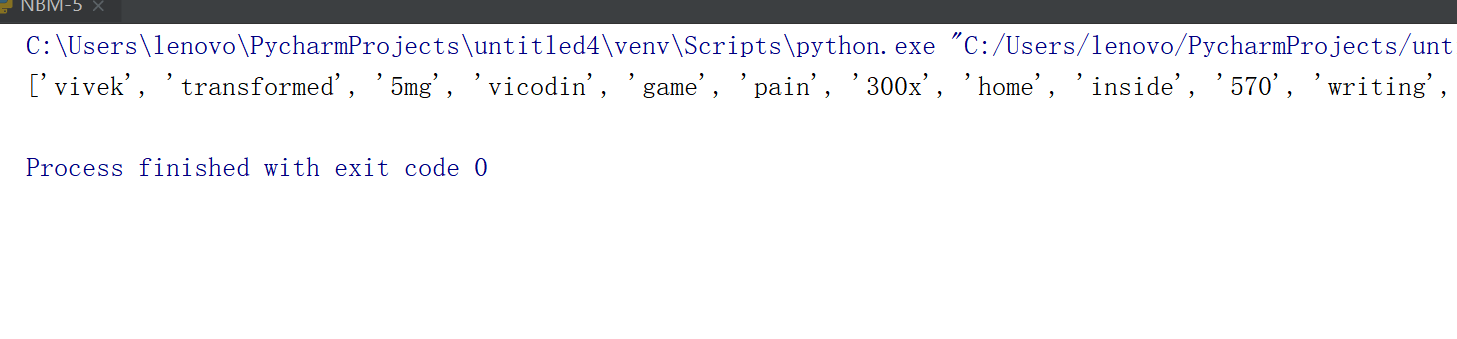
$ python3.6 -c "import re; print(re.split('\s\*', 'I am going to school'))"

更改为：

$ python3.6 -c "import re; print(re.split('\s+', 'I am going to school'))"

代码输出如下：

没有报错了

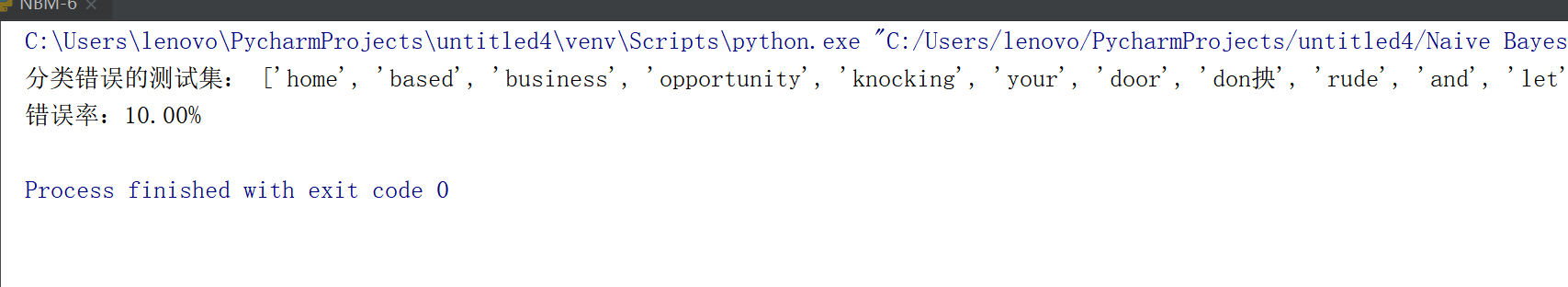


**NBM-6.Py**

根据词汇表，我们就可以将每个文本向量化。

将数据集分为训练集和测试集，使用交叉验证的方式测试朴素贝叶斯分类器的准确性。

代码输出如下：



函数spamTest()会输出在10封随机选择的电子邮件上的分类错误概率。

既然这些电子邮件是随机选择的，所以每次的输出结果可能有些差别。

如果发现错误的话，函数会输出错误的文档的此表，这样就可以了解到底是哪篇文档发生了错误。

如果想要更好地估计错误率，那么就应该将上述过程重复多次，比如说10次，然后求平均值。为了避免错误，可以修正分类器。

# **朴素贝叶斯之新浪新闻分类(Sklearn)**

## **中文语句切分**

**NBM-7.Py**

英文的语句可以通过非字母和非数字进行切分，但是汉语句子呢？

这部分的工作不需要自己做了，可以直接使用第三方分词组件，即jieba。

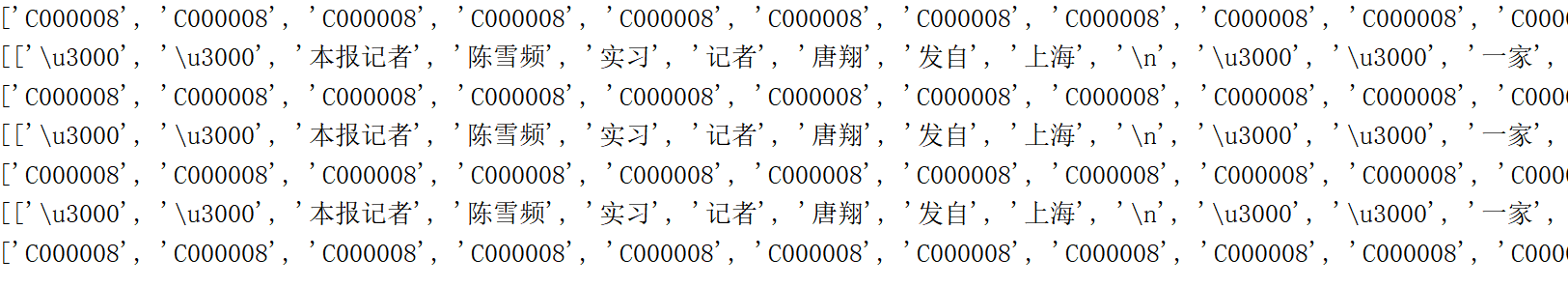
**新闻分类数据集**

数据集已经做好分类，分文件夹保存，分类结果如下：



代码运行结果如下所示:

可以看到，已经顺利将每个文本进行切分，并进行了类别标记。



**文本特征选择**

**NBM-8.Py**

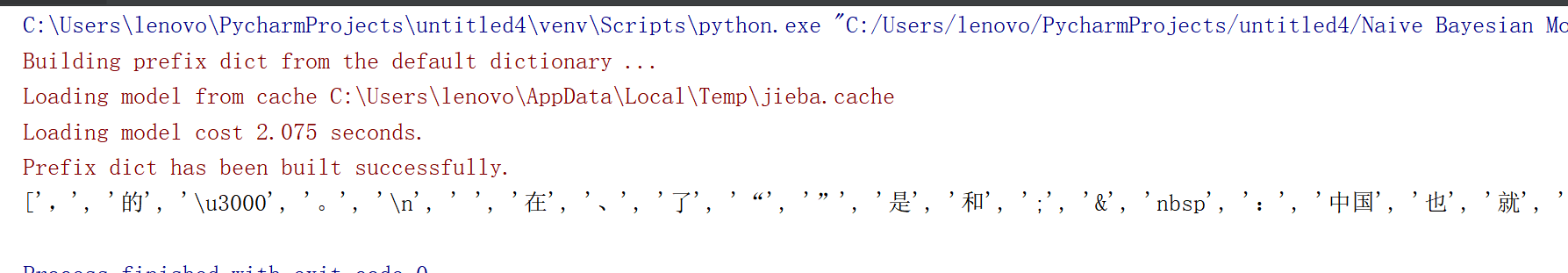
将所有文本分成训练集和测试集，并对训练集中的所有单词进行词频统计，并按降序排序。

将出现次数多的词语在前，出现次数少的词语在后进行排序。

all\_words\_list就是将所有训练集的切分结果通过词频降序排列构成的单词合集。

观察一下结果，发现，这里包含了很多标点符号，很显然，这些标点符号是不能作为新闻分类的特征的。

代码输出如下：



制定的规则：

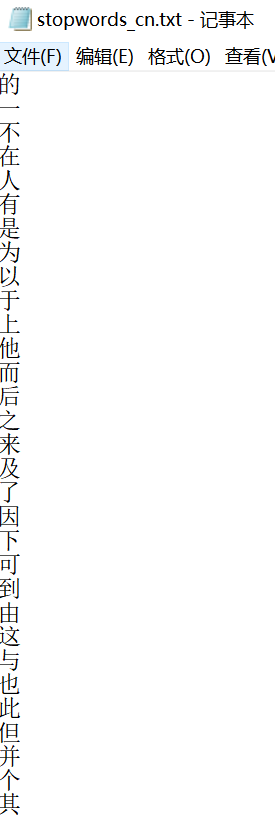
首先去掉高频词，至于去掉多少个高频词，可以通过观察去掉高频词个数和最终检测准确率的关系来确定。

除此之外，去除数字，不把数字作为分类特征。

同时，去除一些特定的词语，比如：”的”，”一”，”在”，”不”，”当然”,”怎么”这类的对新闻分类无影响的介词、代词、连词。

可以使用已经整理好的stopwords\_cn.txt文本。

这个文件打开是这个样子的:

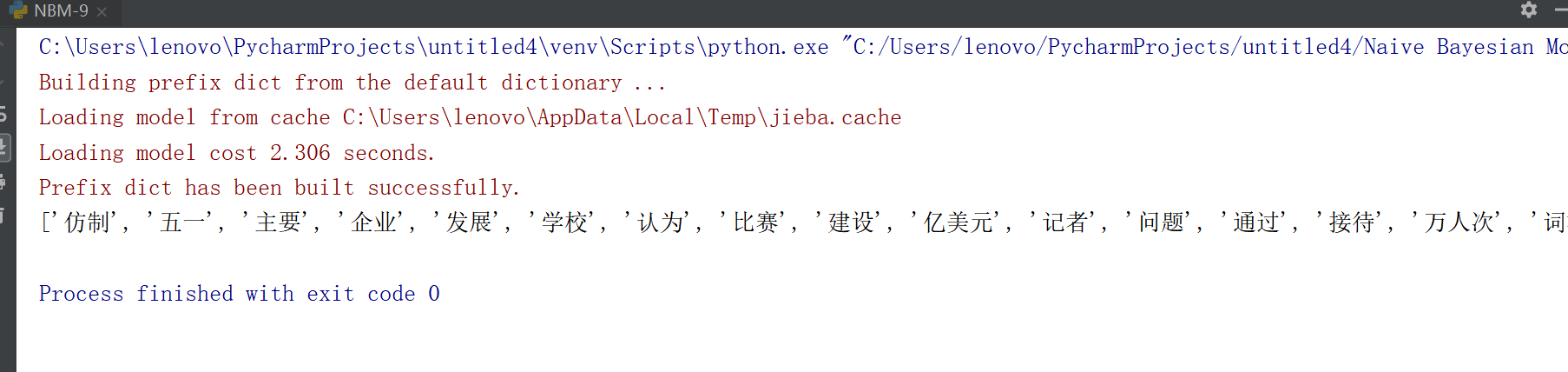


**NBM-9.Py**

可以根据这个文档，将这些单词去除，不作为分类的特征。

我们先去除前100个高频词汇。

代码输出如下：



可以看到，我们已经滤除了那些没有用的词组，这个feature\_words就是我们最终选出的用于新闻分类的特征。

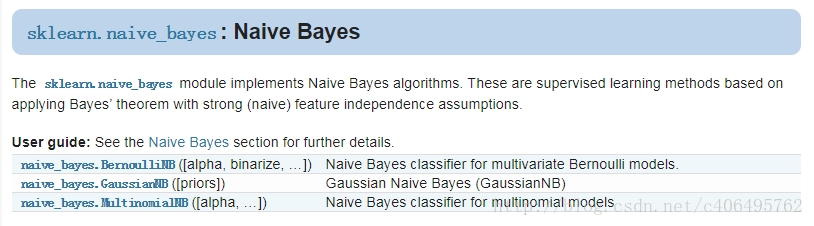
随后，我们就可以根据feature\_words，将文本向量化，然后用于训练朴素贝叶斯分类器。

**使用Sklearn构建朴素贝叶斯分类器**

数据已经处理好了，接下来就可以使用sklearn构建朴素贝叶斯分类器了。

在scikit-learn中，一共有3个朴素贝叶斯的分类算法类。分别是GaussianNB，MultinomialNB和BernoulliNB。

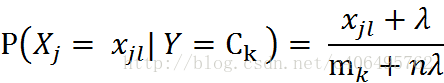
其中GaussianNB就是先验为高斯分布的朴素贝叶斯，MultinomialNB就是先验为多项式分布的朴素贝叶斯，而BernoulliNB就是先验为伯努利分布的朴素贝叶斯。



对于新闻分类，属于多分类问题。

我们可以使用MultinamialNB()完成我们的新闻分类问题。

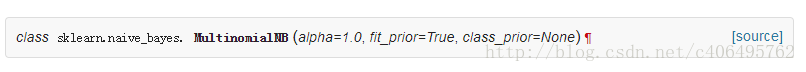
MultinomialNB假设特征的先验概率为多项式分布，即：



其中，P(Xj = Xjl | Y = Ck)是第k个类别的第j维特征的第l个取值条件概率。mk是训练集中输出为第k类的样本个数。

λ为一个大于0的常数，尝尝取值为1，即拉普拉斯平滑，也可以取其他值。

接下来，我们看下MultinamialNB这个函数，只有3个参数：



参数说明如下：

**alpha**：浮点型可选参数，默认为1.0，其实就是添加拉普拉斯平滑，即为上述公式中的λ ，如果这个参数设置为0，就是不添加平滑。

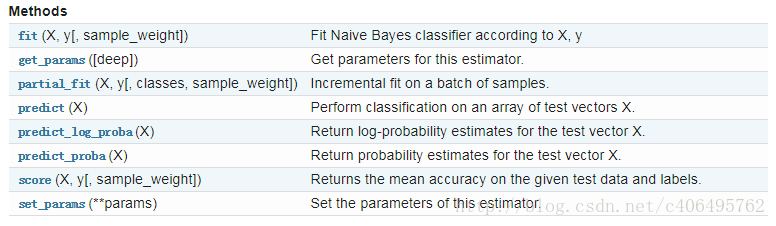
**fit\_prior：**布尔型可选参数，默认为True。

布尔参数fit\_prior表示是否要考虑先验概率，如果是false,则所有的样本类别输出都有相同的类别先验概率。

否则可以自己用第三个参数class\_prior输入先验概率，或者不输入第三个参数class\_prior让MultinomialNB自己从训练集样本来计算先验概率，此时的先验概率为P(Y=Ck)=mk/m。其中m为训练集样本总数量，mk为输出为第k类别的训练集样本数。

**class\_prior：**可选参数，默认为None。

除此之外，MultinamialNB也有一些方法供我们使用：



MultinomialNB一个重要的功能partial\_fit方法，这个方法的一般用在如果训练集数据量非常大，一次不能全部载入内存的时候。这时我们可以把训练集分成若干等分，重复调用partial\_fit来一步步的学习训练集，非常方便。

GaussianNB和BernoulliNB也有类似的功能。 在使用MultinomialNB的fit方法或者partial\_fit方法拟合数据后，我们可以进行预测。此时预测有三种方法，包括predict，predict\_log\_proba和predict\_proba。

predict方法就是我们最常用的预测方法，直接给出测试集的预测类别输出。

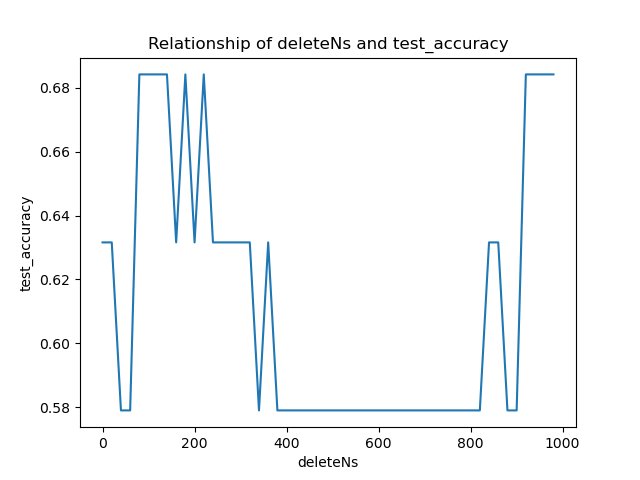
predict\_proba则不同，它会给出测试集样本在各个类别上预测的概率。容易理解，predict\_proba预测出的各个类别概率里的最大值对应的类别，也就是predict方法得到类别。

predict\_log\_proba和predict\_proba类似，它会给出测试集样本在各个类别上预测的概率的一个对数转化。转化后predict\_log\_proba预测出的各个类别对数概率里的最大值对应的类别，也就是predict方法得到类别。

通过观察取不同的去掉前deleteN个高频词的个数与最终检测准确率的关系，确定deleteN的取值：

**NBM-10.Py**

代码输出如下：



**NBM-11.Py**

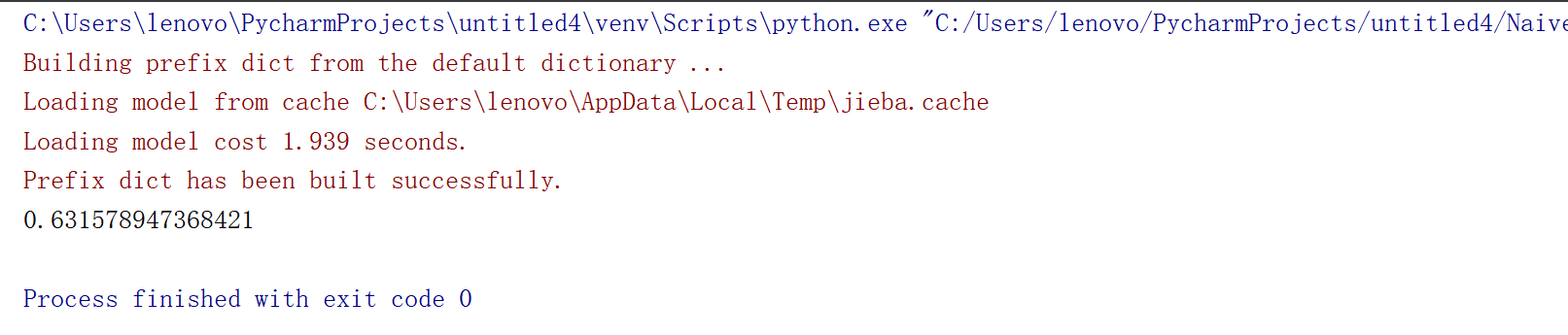
我们绘制出了deleteNs和test\_accuracy的关系，这样我们就可以大致确定去掉前多少的高频词汇了。

每次运行程序，绘制的图形可能不尽相同，我们可以通过多次测试，来决定这个deleteN的取值，然后确定这个参数，这样就可以顺利构建出用于新闻分类的朴素贝叶斯分类器了。

测试感觉450还不错，最差的分类准确率也可以达到百分之50以上。

可以将if \_\_name\_\_ == '\_\_main\_\_'的代码修改一下：

代码输出如下：



**总结**

在训练朴素贝叶斯分类器之前，要处理好训练集。

根据提取的分类特征将文本向量化，然后训练朴素贝叶斯分类器。

去高频词汇数量的不同，对结果也是有影响的的。

拉普拉斯平滑对于改善朴素贝叶斯分类器的分类效果有着积极的作用。