**《机器学习课程设计》实验报告**

**垃圾邮件的识别**

**一、研究现状分析**

自互联网普及以来，电子邮件逐渐成为人们生活中便捷的通信手段之一，然而，随之产生的垃圾邮件如图文艺一样蔓延，造成极大的危害，对现实社会造成危害，使反垃圾邮件行动刻不容缓。因此，对反垃圾邮件技术的研究已经成为影响互联网发展的重要课题之一。

国内外常用的反垃圾邮件技术，基本上都是基于黑名单、规则库、关键字等工作原理。主要发展思路可以分为基于特征过滤、基于改进的SMTP协议、应用知识系统三大类。

一、基于特征过滤技术从过滤的依据划分，可以分为基于IP地址过滤、基于信封过滤、基于信头过滤、基于内容过滤和基于垃圾邮件行为过滤。

二、基于改进的SMTP协议反邮件技术在设计SMTP协议时并没有过多考虑安全性问题，导致传统的SMTP通信机制对使用者没有身份认证功能，为了抵制垃圾邮件，需要改进SMTP协议，加入安全认证机制。

三、应用知识系统是指能够通过样本学习，自动调整系统对于新的样本的适应能力的系统。通过知识的补充来适应更多的样本空间，根据样本变化自平衡调整学习方向，学习经过系统处理的结果，进而克服知识分布不均衡等。

现有技术上，对垃圾邮件的拦截和阻隔基本原理是让计算机匹配垃圾邮件的模式达到对垃圾邮件的识别，翻垃圾邮件是一种模式的识别行为。随着模式识别领域和人工智能领域不断涌现出优秀的研究成果，利用这些优秀成果设计的反邮件技术，搭建性能优秀的反垃圾邮件系统，实现垃圾邮件过滤，阻断垃圾邮件的传播，是目前反垃圾邮件处理技术的发展趋势。

模式识别有结构模式识别和统计模式识别，结构模式识别只适合结构特征比较明显的地方。而垃圾邮件大多属于人为干扰，其结构特征变化多端，因而不适合采用结构模式、统计模式识别是以贝叶斯分类器为主的识别方法，主要内容是分类器的设计以及评价函数以及模式空间的非线性映射。基于贝叶斯分类器的分析算法可以根据用户认为的模式进行统计计算，具有学习渐进的功能，是一种自适应、具有智能统计的方法。**本实验就将基于贝叶斯分类器实现反邮件识别程序。**

**二、数据分析与处理**

**1.数据集特点：**

邮件内容不统一，有些缺少详细信息，比如发件人，收件人等，文件的编码格式也不一致，部分邮件含有非法字符。

**2.格式：**

全是txt格式文件储存，便于读写。

**3.中英文：**

内容都是英文，方便分词。

**4.来源：**

互联网上下载的专门用于垃圾邮件分类的测试数据

**5.规模：**

训练集和测试集都有4000封左右

**6.数据集处理：**

对所有的测试集和训练集的邮件按行进行预读，对于无法按行读取的邮件（在预处理过程中，所有的邮件均可正常打开，但是部分邮件按行读取会出现错误，应当是包含python无法处理的非法字符），手动从测试集和训练集中删除，以免后续的运行出错。

三、**实验方法与分析**

**1.基本原理**

(1)对训练集中的正常邮件和垃圾邮件分别建立词典，词典中的每一项记录词和该词出现的邮件数。对于那些只在正常邮件或者只在垃圾邮件中出现的词，依然把词收入两个词典中只是把没有出现的那个词典的词的出现数记为0。同时舍弃掉那些没有价值的出现总数过多或者过少的词。

（2）定义probSPAM = probHAM = 1

输入一封邮件，抽取其单词，对上述词典中的每一个单词进行处理：

1）在垃圾邮件词典中，若单词A出现在当前邮件中，那么probSPAM \*= （垃圾 邮件中该单词出现次数）/（垃圾邮件数量）；

2）在垃圾邮件词典中，若单词A没有出现在当前邮件中，那么probSPAM \*= [1- （垃圾邮件中该单词出现次数）/（垃圾邮件数量）]；

3）在非垃圾邮件词典中，若单词A出现在当前邮件中，那么probHAM \*= （正常 邮件中该单词出现次数）/（正常邮件数量）；

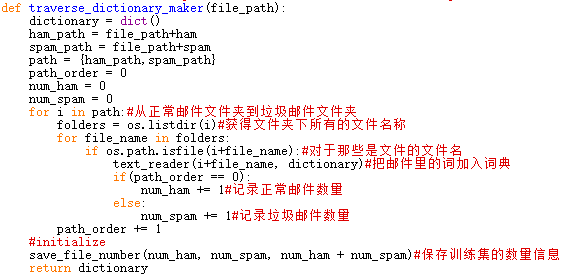
4）在非垃圾邮件词典中，若单词A没有出现在当前邮件中，那么probHAM \*= [1- （正常邮件中该单词出现次数）/（正常邮件数量）]；

（3）比较probSPAM以及probHAM，哪个相对较大就判定为对应空间。

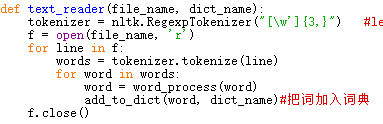
**2.具体实现步骤及代码截图**

(1)构造训练集的总词典

虽然在最终对测试机进行测试时，只需要用到正常邮件词典和垃圾邮件词典，但是在构建它们之前需要首先构造一个囊括了所有训练集的词典，目的是为了后面查找无价值词和正确生成正常邮件词典和垃圾邮件词典。

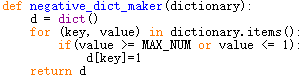


调用该函数构造总词典。需要注意的是，总词典中每个词项对应的不是出现该词的邮件数，而是该词出现的总数（比前者大）。



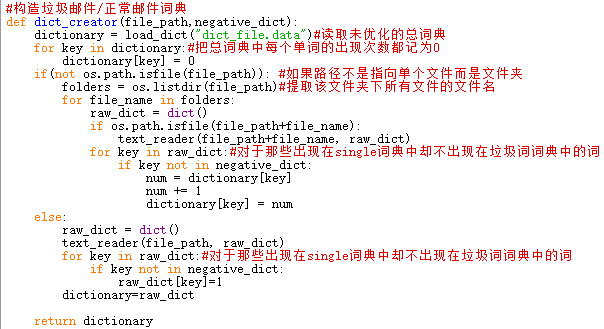
此函数的目的是读取file\_name对应的文件并用其完善词典dict\_name。通过python自带的包nltk，我们可以很容易地对邮件的每一行进行分词，然后把每一个词加入词典（如果词典中没有此词，就添加一项，否则把此词出现的次数加一，此函数非常简单，不再赘述）。这里nltk的分词系统有一些局限性，会出现无法识别引号，把多个词识别为1个，把数字识别为词之类的错误，不过并不会对我们的分类效果产生太大的影响，就忽视掉了。

（2）构建无价值词词典



如图，遍历总词典中的每一个词只把出现过多和过少的词加入新词典并返回新词典。

(3)构建垃圾邮件/正常邮件词典

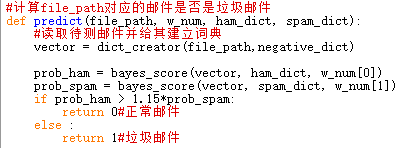


构建垃圾邮件/正常邮件词典的方法类似，都是用此函数（最外层的if语句用于判断路径是文件夹路径还是文件路径，如果是文件夹路径，则生成垃圾邮件/正常邮件词典，否则生成单封邮件词典，后者在测试测试集时会用到）。在函数的开头，先从文件中读取存好的总词典，赋给dictionary，然后把每个词出现的次数置为0。对于每一封邮件，都构造一个新的词典，调用text\_reader实现。这里并非多此一举，而是因为text\_reader构造的词典记录的是词出现的总次数，并非出现的邮件数，而且会丢失出现次数为0的词，所以需要每处理一封邮件都生成一个词典，然后把新生成的词典遍历，找出有价值的词，在dictionary的相应位置加一。

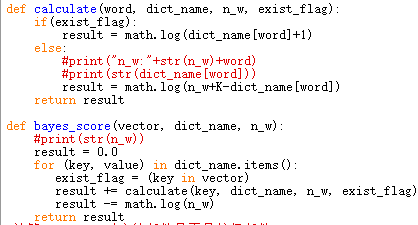
（4）对测试集进行测试



如图，predict函数对每一封邮件进行分类，0表示正常邮件，1表示垃圾邮件。Test函数的主要功能是调用函数测试每一封邮件，并对测试结果进行整理和输出。



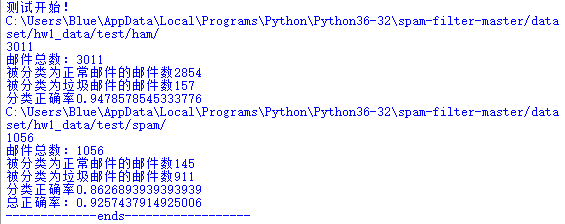
Vector调用dict\_creator给待测邮件建立词典，调用bayes\_score计算此邮件的正常邮件得分和垃圾邮件得分，哪个分数更高则属于哪一类邮件。这里因为正常邮件被分类为垃圾邮件造成的损失远远大于垃圾邮件被分类为正常邮件的损失，所以给prob\_spam乘以一个系数1.15（这里两个值都为负数，所以prob\_spam乘以一个大于1的数之后变小了）



此部分即为前面基本原理（2）的代码化，不再赘述。

**四、结果分析与讨论**

**1.实验结果**



**2.分析与讨论**

分类器的成功率超过了90%，我认为已经实现了分类目标。在实际的垃圾邮件分类器中，还会考虑邮件发送人，黑名单等信息，分类的效果还能够进一步提高。

**五、实验难点与创新点**

在最开始的的时候，基于前几次作业的经验，我们打算先把邮件集抽取一些重要的特征属性（比如是否包含附件，邮件长度等），转换为arff文件，然后使用之前学过的分类方法进行分析。但是这遇到了一些困难，比如数据集残次不齐，有些邮件涉及了我们想要的特征属性，有些根本没有提到这些特征属性。而且许多特征属性的提取难度很大，让我们无从下手。

最后我们在互联网上查找了资料，了解到python很适合进行垃圾邮件分析（因为有许多现成的包可以调用）。之前我们都没有学习过python语言，所以我们花了一些时间了解python(其实很简单，看看就懂了)，并参考了网上博客的资料，最终实现了这个垃圾邮件分类器。