# Lab\_1 基于朴素贝叶斯的垃圾邮件检测

## 一、实习目的与要求

1、结合实际应用理解分类的分类过程；

2、深入理解特征提取、数据处理、训练、测试、分类评估等过程；

3、理论结合实践，采用朴素贝叶斯方法实现垃圾邮件的检测；

4、通过进一步查阅文献，了解相关研究方向的最新研究进展。

## 二、实习题目

利用基于概率论的分类方法——朴素贝叶斯方法，实现垃圾邮件的检测。

#### 【实验数据】

50封包含纯文本内容的电子邮件，其中50% SPAM，50% HAM。

#### 【分类过程描述】

主控制函数为：spamFilter

1. **数据预处理**

**读取文件，并进行单词提取，**

classList=[];fullData=[];docData=[];  
**for** i **in** range(1,26):  
 f=open(**"email/ham/"**+str(i)+**".txt"**,**"r"**); *#好的邮件* data=by.textParge(f.read());  
 classList.append(0);  
 fullData.extend(data);  
 docData.append(data);  
  
 f=open(**"email/spam/"**+str(i)+**".txt"**,**"r"**) *#垃圾邮件* data = by.textParge(f.read());  
 classList.append(1);  
 fullData.extend(data);  
 docData.append(data);

**利用python正则表达式，从邮件文本中提取单词**

**def** textParge(bigString):  
 listOfTokens=re.split(**r'\W'**,bigString) *#以非字母数字的字符来划分  
 #regEX=re.compile(r'\W\*');  
 #listOfTokekens=regEX.split(bigString);* **return** [tok **for** tok **in** listOfTokens **if** len(tok)>2];

**将单词转换成向量：采用了两种转换方法，一种为词集统计（只统计单词是否出现），另一种为词袋统计（统计单词出现的个数）**

**词集统计放文本到向量的转换：**

*#词集统计inputSet在词汇表中词频***def** SetsetWordToVec(vocabList,inputSet):  
 retVocab=[0]\*len(vocabList);  
 **for** item **in** inputSet:  
 **if** item **in** vocabList:  
 retVocab[vocabList.index(item)]=1;  
 **else**:  
 print(item+**"is not in vacabulary"**);  
 **return** retVocab;

**词袋统计放文本到向量的转换：**

*#词带统计inputSet在词汇表中词频***def** setWordToVec(vocabList,inputSet):  
 retVocab=[0]\*len(vocabList);  
 **for** item **in** inputSet:  
 **if** item **in** vocabList:  
 retVocab[vocabList.index(item)]+=1;  
 **else**:  
 print(item+**"is not in vacabulary"**);  
 **return** retVocab;

1. **特征选择**

**所有有效的单词，去除标点符号（因为此时邮件中单词种类较少，故将所有单词都置为分析特征）**

*#创建词汇表***def** createVocabList(dataSet):  
 vocabSet=set([]);  
 **for** item **in** dataSet:  
 vocabSet=vocabSet|set(item);  
 **return** list(vocabSet);

**（3）模型训练**

**使用80%的邮件做训练集，剩余20%做测试集，随机选择。**

**生成训练集和测试集**

*#测试集生成*traingSet=list(range(50));testSet=[];testClass=[];  
**for** i **in** range(10):  
 *#randomIndex=int(random.uniform( 0,len(traingSet))); #随机数生成包含上下限* randomIndex=random.randint(0,len(traingSet)-1)  
 testSet.append(traingSet[randomIndex]); *#traingSet[randomIndex]与randomIndex的区别* testClass.append(classList[randomIndex]);  
 **del**(traingSet[randomIndex]); *#  
  
#训练集训练提取条件概率（训练集生成）*trainMat=[];  
trainClass=[];  
setTrainMat=[];  
**for** item **in** traingSet:  
 trainMat.append(by.setWordToVec(vocabList,docData[item]));  
 trainClass.append(classList[item]);  
 setTrainMat.append(by.SetsetWordToVec(vocabList,docData[item]));  
  
proList0,proList1,probolity=by.trainNB0(trainMat,trainClass); *#计算个字的条件概率*setproList0,setproList1,setprobolity=by.trainNB0(setTrainMat,trainClass);

**结合贝叶斯公式，分析好的邮件中各单词出现的概率和垃圾邮件中各单词出现的概率（即为条件概率）**

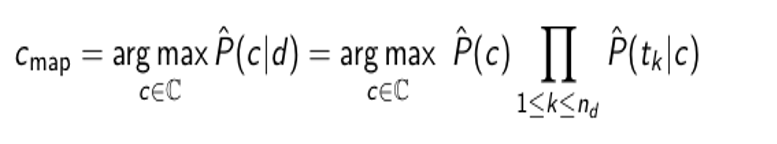


*#计算训练集中各词汇的出现概率（分别在好的邮件和坏的邮件）条件概率***def** trainNB0(trainMarix,trainCategory):  
 line=len(trainMarix);  
 column=len(trainMarix[0]);  
 *#print(sum(trainCategory),line)* probality=sum(trainCategory)/float(line); *#训练集中垃圾邮件的概率* pList0=ny.ones(column);pList1=ny.ones(column);  
 pSum0=pSum1=float(column);  
 **for** i **in** range(len(trainCategory)):  
 **if** trainCategory[i]==0:  
 pList0+=trainMarix[i];  
 pSum0+=sum(trainMarix[i]);  
 **elif** trainCategory[i]==1:  
 pList1+=trainMarix[i];  
 pSum1+=sum(trainMarix[i]);

1. **分类结果评价**

**根据训练模型和贝叶斯公式，分析训练集中每一份邮件属于垃圾邮件属于垃圾邮件的概率和好的邮件的概率。**

**分类器:**



1. **分类器修改：**
2. **加一平滑：防止p(w0|A)\* p(w1|A)\* p(w2|A)\*P(A)中某一项P(wi|A)为0，采用加一平滑，分子初始化为1，分母初始化为词汇表的长度（即单词的种类数）。**
3. **对数计算：防止p(w0|A)\* p(w1|A)\* p(w2|A)\*P(A)太多太小的数相乘从而造成下溢出。**

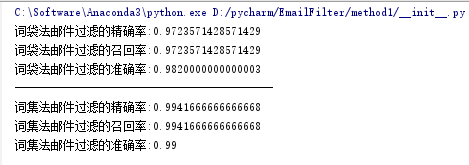


*#测试集代入模型分析过程***def** classifyNB(testData,proList0,proList1,probality):  
 p0=p1=0.0;  
 **for** i **in** range(len(proList1)):  
 p0+=(testData[i]\*mt.log(proList0[i]));  
 p1+=(testData[i]\*mt.log(proList1[i]));  
  
 p1+=mt.log(probality);  
 p0+=mt.log(1-probality);  
 **if** p1>p0:**return** 1;  
 **else**: **return** 0;

**测试数据分别代入词集分析和词袋分析：使用的分析过程都是上面函数classifyNB**

*#训练集训练并推断出错误率*errorCount=0;setErrorCount=0;  
**for** item **in** testSet:  
 docVec=by.setWordToVec(vocabList,docData[item]);  
 *#print(docVec)  
 #print(by.classifyNB(docVec, proList0, proList1, probolity), classList[item])* **if** by.classifyNB(docVec,proList0,proList1,probolity)!=classList[item]:  
 *#print(by.classifyNB(docVec,proList0,proList1,probolity),classList[item])* errorCount+=1;  
 setDocVec=by.SetsetWordToVec(vocabList,docData[item]); *#词集方法计算邮件过滤的错误率* **if** by.classifyNB(setDocVec,setproList0,setproList1,setprobolity)!=classList[item]:  
 setErrorCount+=1;

**结果：**



#### 【源代码】

（附上源代码）

1. 采用多次验证的方式，将数据测试1000次，每一次都随机选取10份邮件作为测试集，其余作为训练集。总后对每次的结果求平均。

rates=[];setRates=[];  
**for** i **in** range(1000):  
 rate,setRate=spamFilter();  
 rates.append(rate);  
 setRates.append(setRate);  
  
print(**"词袋法邮件过滤的错误率:"** + str(sum(rates)/len(rates)));  
print(**"词集法邮件过滤的错误率:"** + str(sum(setRates)/len(setRates)));

采用SKLearn中伯努利朴素贝叶斯分析垃圾邮件（其余代码与之前所述过程相同）

glf=BernoulliNB();  
glf.fit(trainMat,trainClass);  
error=0;  
**for** item **in** testSet:  
 docVec=[by.setWordToVec(vocabList,docData[item])];  
 result=glf.predict(docVec);  
 **if** result[0]!=classList[item]:  
 error+=1;  
  
setError=0;  
glf = BernoulliNB();  
glf.fit(setTrainMat, trainClass);  
**for** item **in** testSet:  
 setDocVec=[by.SetsetWordToVec(vocabList,docData[item])];  
 result=glf.predict(setDocVec);  
 print(result[0],classList[i])  
 **if** result[0]!=classList[i]:  
 setError+=1;

#### 【改进设想】

（或查阅文献提出改进设想；或考虑与第三方机器学习库分类结果的比较；或与采用其他分类算法实现的结果比较等）。

1:训练集采用**5-折交叉验证**，而不是直接训练集简单训练。充分利用了训练集的数据，优化了训练模型。

2.采用SKLearn中的贝叶斯方程进行分析和自己编写的贝叶斯公式分析垃圾邮件做对比。如GaussianNB(高斯朴素贝叶斯)、MultinomialNB(多项式朴素贝叶斯)、BernoulliNB(伯努利朴素贝叶斯)。综合分析。

## 三、本次实习小结

通过本次实习，学习了基于贝叶斯的分析过程，对应于本次实习，主要也就是单词提取并转化为向量，生成词汇表并构建被贝叶斯公式要求的条件概率的反式，再根据贝叶斯公式求出求出所要求解的概率，即某一邮件属于好的邮件的概率以及坏的邮件的概率。后期尝试了一下基于SKlearn的贝叶斯分析，再词集方法统计上和自己的差别很大，SKLearn中的贝叶斯公式计算的错误率可达0.5，且往往都是将好的邮件分成了垃圾邮件。感觉自己要做的还有很多。不懂的地方也还有很多。继续努力。

=======================================================

我的联系方式：

电话：13349830890

邮箱：liuyy@cug.edu.cn