实验1 - 朴素贝叶斯实验报告

陈意扬 计96 2019011341

实验目的

1）实现一个基于贝叶斯算法的垃圾电子邮件分类器

2）在真实数据集上评估分类器性能

3）分析实验结果

实验原理

本次实验所基于的实验原理为贝叶斯学习中的朴素贝叶斯方法。模型中把每封电子邮件视为一个样本点`<xn,yn>`，`xn`为抽取的特征向量，`yk`为`label`，在{spam，ham}中取值。贝叶斯公式为

$$P(y_k=\omega_i|x_k)=\frac{P(x_k|y_k=\omega_i)P(\omega_i)}{P(x_k)}=\frac{P(x_k|y_k=\omega_i)P(\omega_i)}{\Sigma_j P(x_k|y_k=\omega_j)P(\omega_j)}$$

$P(\omega_i)$表示该样本点属于第i类别的先验概率。本问题中i只有两类，$spam$和pam。

朴素贝叶斯模型对特征有独立性假设，则对于n维特征向量`xk`有

$$P(x_k|y_k=\omega_i)=\Pi_{j=1}^{n}P(x_k^{(j)}|y_k=\omega_i)$$

$$P(y_k=\omega_i|x_k)=\frac{P(\omega_i)\Pi_{j=1}^{n}P(x_k^{(j)}|y_k=\omega_i)}{\Sigma_m(P(\omega_m)\Pi_{j=1}^{n}P(x_k^{(j)}|y_k=\omega_m))}$$

贝叶斯分类器把后验概率最大的一类作为测试样本的分类，当label只有两个时，则取大于（等于）0.5概率的类别为分类。然而实际上，由于假阴性和假阳性错误分类的代价不同，当spam贝叶斯概率大于0.7时我才将测试样本邮件判定为spam。

实验步骤

1.切分数据集并提取特征（parse_email.py）

实验数据集来自课程提供的The English E-mail Data set。总共37822封email被分装在127个文件夹中。实验要求采用交叉验证，所以我将数据集切分为5份（按文件夹名排序000到025、026到050、051到075、076到100和101到126）,轮流取其中一份为测试集，其余四份为训练集。在实际parse过程中部分邮件无法解码，实际使用到的邮件有32401封，其中垃圾邮件有20030封。

```
➜  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word_sender.py 0 0
spam: 20030
total: 32401
```

通过对邮件格式的观察我发现在邮件头和邮件内容之间有一个空行，通过识别空行我将每封邮件分为header和content两个部分。在header中通过对'Subject:'的识别提取出title。我认为相同的词在title中和在content中属于不同的特征，所以在抽取bag of words特征的时候将content中的word和title中的word通过加上后缀分开。
同时我将所有的数字都归类为“number”特征、所有的链接归类为“`url`”特征、所有的`email addr`归类为“email”特征，并根据它们出现在title还是content中加上后缀。对于字母和数字混杂的词，我使用`eliminate_num_from_str`方法将字母词干提取出来作为一个word并同时记录一个“number”特征，所有words都去除了首尾的标点符号。
参考文本分类的general做法我选择去掉words中的停用词。我使用了之前的人工智能导论课实验作业的停用词列表（stopwords.txt），将已提取出来的words小写化之后和停用词表比对来去除停用词。
经过上述抽取后每个训练集中term的数量在50w左右（使用交叉验证方法循环选取训练集，总共有5个训练集，图中命令行两个参数为测试集的文件夹范围，其余文件作为训练集），存储在`exist_dict{}_{}.npy`（所有邮件中

的words）和`exist_spam_dict{}_{}.npy`（spam邮件中的words）中，这里我抽取的特征是每个word在邮件中的存在情况，即记录一个词在多少封邮件中出现。我没有选择word出现的频率，因为我认为记录频率存在较大bias（同一个word在一封邮件中反复出现，这样的word频率大然而对邮件分类提供的信息很少）。

2.精炼特征（select_word.py）

50w的term规模太大，我在select_word.py对前一阶段得到的`npy`文件进行words的精炼。我首先做了频次截断，去掉所有出现在小于5封邮件的words。之后，我在`ens2(word, dict, dict2)`方法中计算经每个word分类后（有这个word的邮件和没有这个word的邮件）的信息增益，按信息增益的大小筛掉后2%的words。
经过特征精炼后，每个训练集中term的数量减少到4w左右。

```
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word.py 0 26
spam: 16078
total: 25574
42329.14
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word.py 26 51
spam: 16122
total: 25893
43178.799999999996
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word.py 51 76
spam: 15996
total: 25900
43660.96
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word.py 76 101
spam: 15943
total: 26131
40080.04
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word.py 101 127
spam: 15981
total: 26106
39374.44
```

3.对测试集进行测试（test.py）

首先对测试集进行特征提取（方法和对训练集的特征提取类似）。提取之后根据贝叶斯公式计算概率（由频率计算概率）。

$$其中P(\omega_i)可以根据训练集中垃圾邮件数和总邮件数计算。$$

$$P(x_k^{(j)}|y_k=\omega_i)可以根据垃圾邮件中出现该word的邮件数和所有邮件中出现该word的邮件数计算。$$

在垃圾邮件分类场景中，将一封正常的邮件错归类于垃圾邮件的代价是远高于把一封垃圾邮件错分类为正常邮件的。后者可以被用户人为审查筛掉，而前者则可能让用户错失重要邮件。所以在最终分类时，我将垃圾邮件贝叶斯概率>0.5改成>0.7，这个策略在接下来的模型evaluation中有明显的效果。

模型的评价

我使用了Accuracy，Precision，Recall，False Positive rate，False Negative rate这些指标来评估各分类器。

	分类为spam	**分类为ham**
真实类别spam	True Positive	False Negative
真实类别ham	False Positive	True Negative

accuracy：所有样本正确分类的占比

precision：分类为spam中真实类别为spam的占比

recall：真实类别spam中分类为spam的占比

实验结果（平滑系数alpha=0.00001）

指标	best	worst	average
accuracy	0.98543	0.98286	0.98397
precision	0.99624	0.99499	0.99562
recall	0.98226	0.97786	0.97937
false positive rate	0.00706	0.00888	0.00774
false negative rate	0.01774	0.02214	0.02063

实验采用交叉验证，训练集的选取不同得到5个模型，best、worst和average是在这5个模型中比较取得。以上为实验提交的最终模型的性能。该模型主要考虑以上五个指标，而对分类速度的考虑较少，所以在测试环节（test.py）的耗时较长。接下来评价不同模型参数对性能的影响。

不同截断频率对性能的影响（信息增益丢弃=2%，平滑系数=0.00001，spam分类概率要求0.7）

指标（average）	5	10	20
accuracy	0.98397	0.96104	0.94109
precision	0.99562	0.99652	0.99659
recall	0.97937	0.94370	0.91224
false positive rate	0.00774	0.00622	0.00575
false negative rate	0.02063	0.05630	0.08776

可以看出随着截断频率的提升，accuracy和recall明显下降，false negative rate明显上升。结合对邮件的人为观察，我推测可能的原因是ham邮件的特征词多样性高，单个词出现的频率普遍较低，随着截断频率的提升主要丢失了ham邮件的特征词的信息，导致false negative rate明显上升；spam邮件特征词多样性较低，且单个词出现的频率较高，频率截断截掉较少的spam邮件的特征词，所以precision和false positive rate影响较小甚至有轻微上升。

不同信息增益丢弃对性能的影响（截断频率为5，平滑系数=0.00001，spam分类概率要求0.7）

指标（average）	0	0.02	0.1	0.5
accuracy	0.98401	0.98397	0.97451	0.94652
precision	0.99560	0.99562	0.99595	0.99598
recall	0.97932	0.97937	0.96497	0.92069
false positive rate	0.00988	0.00774	0.00743	0.00672
false negative rate	0.01995	0.02063	0.03503	0.07931

可以看出当对信息增益较小的word适量丢弃时（0.02），accuracy、precision、recall等指标几乎不变，而false positive rate明显下降。结合对邮件的人为观察，spam邮件特征词信息增益都较大，很多特征如`url`存在性，ham邮件很少存在，这些词的信息增益很大。而信息增益小的一般是ham和spam邮件中普遍存在的词，这些词可能因为训练集的选取而存在较大bias，对那些处在分类概率要求0.7附近的邮件有较大影响。对信息增益较小的word适量丢弃可以去掉这些有较大bias的特征词，降低false positive rate。

然而当丢弃过多时，主要丢弃的是信息增益普遍较小的ham邮件特征词，导致accuracy、recall和false negative rate的大幅下降。

不同分类判断标准对False Positive rate和False Negative rate的影响（信息增益丢弃=2%，截断频率为5，平滑系数=0.00001）

指标（average）	0.5	0.7	0.9
false positive rate	0.01132	0.00774	0.00730
false negative rate	0.01776	0.02063	0.03456

分析可知贝叶斯概率大于0.5的ham邮件主要集中在0.5到0.7的区间上，若继续提升分类概率要求false positive rate下降微弱而false negative rate激增，得不偿失。

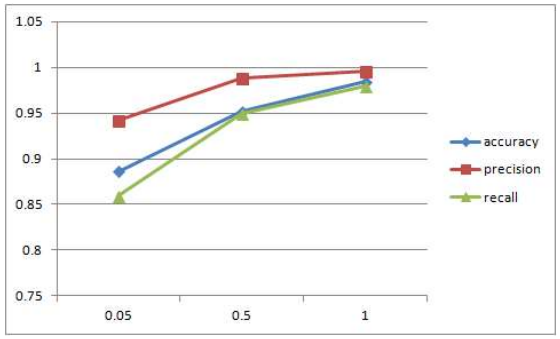
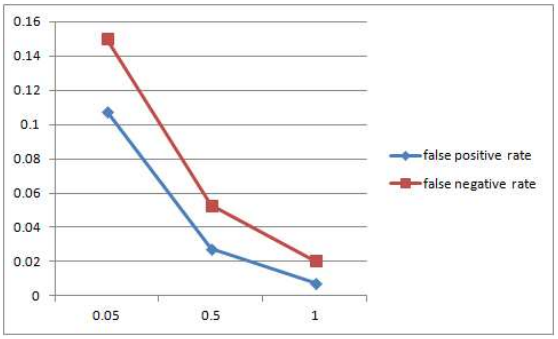
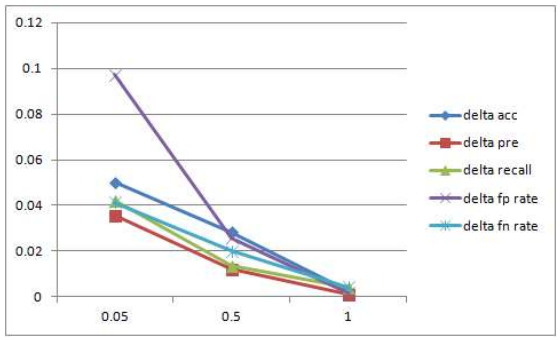
ISSUE 1: THE SIZE OF TRAINING SET

同样用交叉验证的方法，我取原训练集规模的5%（5个文件夹），50%（51个文件夹）和100%（101个文件夹）作为训练集，训练得到5%、50%、100%级别模型，剩下的数据集作为测试集进行测试，实验数据如下（信息增益丢弃=2%，截断频率为5，平滑系数=0.00001，spam分类概率要求0.7）

指标（5%级别）	best	worst	average
accuracy	0.91253	0.86245	0.88591
precision	0.96757	0.93203	0.94142
recall	0.88162	0.83930	0.85879
false positive rate	0.04196	0.13904	0.10774
false negative rate	0.13201	0.17348	0.14996

指标（50%级别）	best	worst	average
accuracy	0.96989	0.94147	0.95148
precision	0.99231	0.98007	0.98762
recall	0.95470	0.94105	0.94913
false positive rate	0.01862	0.04443	0.02794
false negative rate	0.04071	0.06070	0.05286

指标	best	worst	average
accuracy	0.98543	0.98286	0.98397
precision	0.99624	0.99499	0.99562
recall	0.98226	0.97786	0.97937
false positive rate	0.00706	0.00888	0.00774
false negative rate	0.01774	0.02214	0.02063

性能随训练集规模的变化	
	
性能波动幅度（max-min）随规模的变化	
	

观察实验结果可以发现训练集规模对性能有很大的影响。当训练集规模较小的时候（5%），所有指标都很差，并且best和worst之间极差很大，性能随训练集选取不同而有很大的波动；当训练集选取中等规模（50%）时，性能比100%稍差但已经十分接近，并且波动幅度有明显减小。我推测当训练集规模较小（5%）的时候bag of words能提取的特征word太少，而且随训练集选取不同提取的特征words在整体上有明显的差别，测试集中的很多的词都不在模型的特征中而只能用零概率平滑去计算，导致准确率低下。同时结合对邮件的人为观察我发现邮件中有极少量的“噪声”邮件，即words比较反常的邮件（ham邮件中出现url或者spam邮件粗看很正常），训练集规模大能很好地稀释这样的“噪声”邮件，而当训练集规模较小时，这些“噪声”邮件就会影响性能，且随着“噪声”邮件的分布不同导致性能随训练集选取不同而有很大的波动。

ISSUE 2: ZERO-PROBABILITIES

由于测试集中很大概率存在着贝叶斯概率为零的特征（模型特征words中不存在的word），如果直接计算会导致贝叶斯概率直接归0而丢失其他所有信息。为了避免这种情况的诞生，我们拟采用平滑处理的方法，利用下式对于某维特征上的概率进行平滑处理。

$$P(x_k^{(i)}|y_k=\omega_j)=\frac{\alpha+\Sigma_{l=1}^{M}I(x_l^{(i)}=x_k^{(i)},y_l=\omega_j)}{M\alpha+\Sigma_{l=1}^{M}I(y_l=\omega_j)}$$

其中I(condition)为示性函数，满足condition函数值为1否则为0。

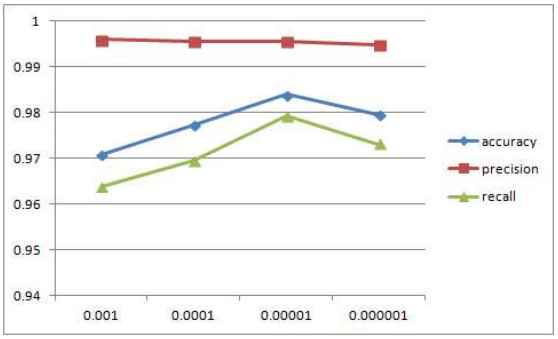
$\Sigma_{l=1}^{M}I(x_l^{(i)}=x_k^{(i)},y_l=\omega_j)$表示所有$label$为$j\in\{spam,ham\}$的训练集邮件中出现$x_k^{(i)}$对应的特征$word$的邮件数。

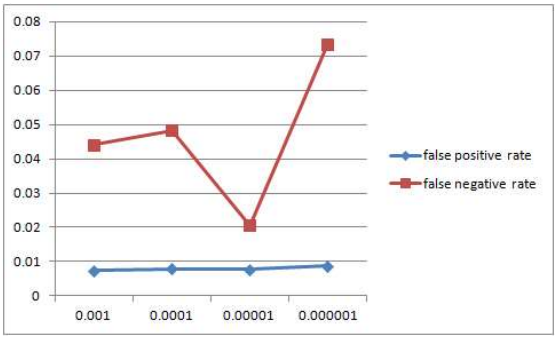
$\Sigma_{l=1}^{M}I(y_l=\omega_j)$表示所有$label$为$j\in\{spam,ham\}$的训练集邮件数

M为训练集邮件的总数

指标	0.001	0.0001	0.00001	0.000001
accuracy	0.97091	0.97742	0.98397	0.97966
precision	0.99603	0.99555	0.99562	0.99484
recall	0.96378	0.96952	0.97937	0.97312
false positive rate	0.00751	0.00793	0.00774	0.00876
false negative rate	0.04409	0.04828	0.02063	0.07342

性能随平滑系数alpha的变化





观察实验数据可知刚开始随着alpha的减小，accuracy和recall指标均有小幅上升，precision、false positive rate和false negative rate则有不规则波动；当alpha的值小到一定程度（0.00001）后，accuracy和recall指标达到最好；继续减小alpha到0.000001时，accuracy和recall指标反而开始下降，false negative rate骤增。我推测刚开始alpha值较大时（0.001），对本身非零的概率有较大的影响，会降低性能；减小alpha到一定程度后（0.00001）达到最优；继续减小alpha，本身为零的概率在平滑后也变得过于小，如果一个testcase有较多模型特征中没有出现的words时，可能造成原本后验概率上占据绝对优势的类别概率变得过小，和不平滑没有区别，从而导致accuracy和recall指标下降，false negative rate骤增。

ISSUE 3: SPECIFIC FEATURES

通过对数据集中邮件的人为观察，我发现垃圾邮件和正常邮件的发送者邮箱的类型很有特点：正常邮件后缀很多为edu，而垃圾邮件后缀为com的居多。针对这个发现我决定对发送者邮箱的后缀进行特征提取。

同样采用交叉验证的方法，我将数据集切分为5份（按文件夹名排序000到025、026到050、051到075、076到100和101到126），轮流取其中一份为测试集，其余四份为训练集。按照之前parse邮件的方法将邮件分为header和content，在header中通过对'From:'的识别提取出`sender email addr`（这部分逻辑见parse_email_sender.py）。由于是邮箱后缀的特征提取，这部分我没有（也不应该）去掉停用词，而是根据“.”分词之后去掉带@的邮箱主体部分，留下后缀。

在select_word_sender.py中，我对频率3进行截断之后去除信息增益最小的10%的后缀，剩余的特征后缀在500个左右。

```
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word_sender.py 0 26
spam: 16078
total: 25574
501
450.90000000000003
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word_sender.py 26 51
spam: 16122
total: 25893
559
503.1
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word_sender.py 51 76
spam: 15996
total: 25900
574
516.6
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word_sender.py 76 101
spam: 15943
total: 26131
572
514.8000000000001
→  naive-bayes-for-mail-classification git:(main) ✗ python3 select_word_sender.py 101 127
spam: 15981
total: 26106
585
526.5
```

实验结果如下（平滑系数=1，spam分类概率要求0.7）

指标	best	worst	average
accuracy	0.81659	0.68608	0.74405
precision	0.91448	0.71491	0.80563
recall	0.80218	0.76439	0.79202
false positive rate	0.13880	0.47577	0.33692
false negative rate	0.19782	0.23561	0.21001

分析实验结果可知根据sender邮箱后缀提取特征训练的模型在性能上远不及bag of words特征的模型，并且随训练集选取不同性能有很大的波动幅度（10%级别）。根据issue1分析的经验我推测造成上述结果的主要原因还是在于特征数量太少，并且特征的信息增益普遍低于bag of words。可以注意到该模型的worse false positive rate竟然接近于50%，我想这是因为“噪声”后缀非常多而且分布极为不均匀，当选取作为测试集的数据集部分集中出现“噪声”后缀时，对于false positive rate影响极大。同时，我注意到平滑系数alpha的选择需要大于bag of words模型的alpha多个数量级才能取得较好的性能，我推测对于特征本身较少的模型，测试集大量后缀没有包括在模型特征中，这时零概率平滑系数如果选的太小会让贝叶斯概率普遍都接近于0，极大影响性能。

实验结论

对于垃圾邮件分类这个场景，基于朴素贝叶斯算法和“bag of words”特征的模型表现优异，若训练得当在accuracy上可以达到98%，false positive rate低于1%。在算法相同的情况下，对特征的选取不同、对所选特征的处理不同、训练集规模的不同、零概率平滑系数的选取不同都会影响性能。其中对特征的选取不同和训练集规模的不同会极大程度地影响模型表现（10%幅度），对所选特征的处理不同和零概率平滑系数的选取不同对模型表现的影响相对较小（1%幅度）。