朴素贝叶斯分类器实验报告

实验设计

实验原理

实验基本原理是贝叶斯公式，也就是$P(B|A) = P(B)\frac{P(A|B)}{P(A)}$，根据该公式我们可以根据先验概率估计事件，

实验的原理基础是基于该公式的假设，也就是$P(y|x_1, x_2, \ldots, x_n)$正比于$P(y)\prod_{i=1}^{n} P(x_i|y)$。

在垃圾邮件分类这个特定的任务下，我们将是否为垃圾邮件记为y，至于x_i则基于邮件中出现的某个因素，于是$P(x_i|y_{spam})$的含义便是在所有垃圾邮件中，该因素出现的概率。为此，我们首先需要在目前已知邮件中提取并统计出我们需要的信息，最基础的设计是仅基于正文内容的词袋模型，但对每个词并不记录频数，而仅记录是否出现。

Tips: 如果希望利用到频数的信息，我们也可以定义类似于“出现次数在0～3次”，“出现次数在3～10次”，“出现次数在10次以上”这样的因素，只不过会占用更多的存储空间，并且粒度越细占用得越大，为了简洁，并没有采用这种方法。

实验流程

大致的实验流程如下:

1. 先处理文本，例如提取出正文等需要信息，去掉停用词
2. 在训练集中统计得到需要的概率
3. 在测试集中根据训练集中得到的概率进行测试与标签进行比对，根据评价指标计算结果

结果评价

采用准确率（accuracy），精准率（precision），召回率（recall），f1 socre作为评价的指标。

准确率计算公式为$\frac{TruePositive}{TruePositive+TrueNegative}$，是衡量模型好坏主要的指标

精准率计算公式为$\frac{TruePositive}{TruePositive+FalsePositive}$

召回率计算公式为$\frac{TruePositive}{TruePositive+FalseNegative}$

f1 score计算公式为$2 * \frac{precision*recall}{precision+recall}$

训练时采用5-fold方法，最后取平均值，

为了便于计算，将概率取统一log10，可将概率的连乘转变为求和，并且一定程度上提高精度。

关于实验的涉及到的另外一些细节会在后续报告中说明。

issue1 训练集大小对训练结果的影响

分别在训练集中采$5\%, 25\%50\%, 75\%100\%$进行训练（在已经进行5-fold五等分后得到的训练集上），并且每次要随机得采五次取平均值以消去偶然影响。

在零概率问题上 采取$\alpha = 1$的平滑方法。

5%

	accuracy	precision	recall	f1 score
min	0.9684	0.9672	0.9586	0.9733
max	0.9889	0.9939	0.9908	0.9907
avg	0.982	0.987	0.9839	0.9854

25%

	accuracy	precision	recall	f1 score
min	0.981	0.9723	0.9981	0.9854
max	0.999	0.9983	1.0	0.9992
avg	0.9944	0.9923	0.9991	0.9956

50%

	accuracy	precision	recall	f1 score
min	0.9802	0.9721	0.9911	0.9848
max	0.9983	0.9978	0.9994	0.9986
avg	0.9925	0.9917	0.9965	0.9941

75%

	accuracy	precision	recall	f1 score
min	0.9805	0.9721	0.9981	0.985
max	0.999	0.9983	1.0	0.9992
avg	0.9941	0.9921	0.9987	0.9953

100%

	accuracy	precision	recall	f1 score
min	0.981	0.9723	0.9981	0.9854
max	0.999	0.9983	1.0	0.9992
avg	0.9944	0.9922	0.9991	0.9956

在训练集规模增大的过程中，可以清楚地看到准确率，精准率，召回率，f1 score四个指标都在增加，意味着性能的提高，这是意料之中的，因为更多的训练样本意味着更多数据更多信息，可以获得与真实情况更加接近的统计结果。

issue2 零概率问题

实验过程中，会遇到两种零概率情况。

第一种是在训练集统计时中出现的，只在一种标签中出现的项，由于这种项的存在，会使得在连乘（实际上是连加，但$log(0)$趋于负无穷）时没出现过的那一个标签的概率变成 0，直观上来说这样导致的结果是分类器会过于武断，也就是仅仅因为一个词在训练集中没有在该标签中出现过就一票否决了该标签的可能性，虽然我们必须承认这样的词项为我们的判断帮助是比较大的。

第二种是在测试时测试集中出现了训练集中没有出现过的项。

这两种情况出现的原因应该都是属于训练量不足。

对此我采取了两种措施，一种是使用类似于拉普拉斯的平滑$\frac{\#\{y=y_i,x_i=k\}\alpha}{\#\{y=y_i\}+M\alpha}$，M是所有词项的个数，一种是直接赋予这样的项一个较小的概率。

类拉普拉斯平滑

$\alpha = 0.1$

	accuracy	precision	recall	f1 score
min	0.9817	0.9726	0.9981	0.986
max	0.999	0.9986	1.0	0.9992
avg	0.9947	0.9927	0.9992	0.9959

$\alpha = 1$

	accuracy	precision	recall	f1 score
min	0.981	0.9723	0.9981	0.9854
max	0.999	0.9983	1.0	0.9992
avg	0.9944	0.9922	0.9991	0.9956

$\alpha = 10$

	accuracy	precision	recall	f1 score
min	0.979	0.972	0.9959	0.9838
max	0.9983	0.9978	0.9994	0.9986
avg	0.9928	0.9916	0.9971	0.9943

从结果上来看三个α取三个值的效果都相差不远，在实验中主要采用传统的拉普拉斯平滑也就是

固定小概率

$P = 0.1$

	accuracy	precision	recall	f1 score
min	0.6835	0.6835	0.6708	0.7338
max	0.7947	0.9339	0.7195	0.8063
avg	0.7511	0.8815	0.6913	0.7746

$P = 0.01$

	accuracy	precision	recall	f1 score
min	0.8901	0.8901	0.864	0.9111
max	0.9348	0.99	0.9014	0.9425
avg	0.9145	0.9772	0.8825	0.9274

$P = 1e - 5$

	accuracy	precision	recall	f1 score
min	0.9783	0.9725	0.9939	0.9833
max	0.9978	0.9989	0.9977	0.9982
avg	0.9925	0.9927	0.9955	0.9941

$P = 1e - 10$

	accuracy	precision	recall	f1 score
min	0.9819	0.9726	0.9991	0.9861
max	0.9992	0.9989	1.0	0.9993
avg	0.9951	0.9929	0.9996	0.9962

$P = 1e - 20$

	accuracy	precision	recall	f1 score
min	0.9819	0.9726	0.9994	0.9861
max	0.9992	0.9989	1.0	0.9993
avg	0.9953	0.9929	0.9998	0.9963

$P = 1e - 50$

	accuracy	precision	recall	f1 score
min	0.9819	0.9726	0.9997	0.9861
max	0.9993	0.9989	1.0	0.9994
avg	0.9953	0.9929	0.9999	0.9964

从以上数据中我们可以清楚得看到，当该概率越来越小的时候，实际上效果是越来越好的，甚至趋于正确率1了。而这个常数越来越小则意味着一个零概率的词否决一个标签的影响，当$P = 1e - 50$时，一个没有出现过的词项对最后的计算是直接-50的，当P越来越小，最后结果也越来越趋近于就是乘了一个零概率，而效果却是越来越好，这其实是超出预期的。但我们也可以从这个现象推断原因：垃圾邮件一般会包含一些正常邮件绝不会出现的词项，导致我们即使是根据这些词“一票否决是正常邮件”的可能性，最后分类器的正确率依然非常高。

根据上面的分析在这个场景下，我猜想直接找垃圾邮件和正常邮件的词项集合的差集就可以得到比较好的结果，但这可能是因为数据集的特性，并非通法，有局限性，这种方法可能是钻了牛角尖，正统的方法应该还是借助于朴素贝叶斯，并且在实验时，使用平滑而不是使用这种极小概率。

issue3 其他特征

除了正文，我觉得邮件中可以提供的另一重要信息是发件人。

一方面，发送垃圾邮件的邮箱应该是集中的，也就是说一般发送垃圾邮件的人是群发的，或者发送垃圾邮件的人是一个团体，或者一个人有很多账号，这些账号是趋于一致的，例如使用同一个邮箱服务器，所以我将发件人视为仅次于正文的第二重要的判别垃圾邮件的信息。

另一方面，如果邮箱地址中包含.edu等字样也大概率不会是垃圾邮件，这也是很重要的信息。

为利用上这一信息，类似于对正文的处理，对训练集中的邮箱进行提取，然后去掉前面的用户名，对后面的字符串以'.'进行分割，建立另一个表，就可以利用上邮箱地址的信息了。

还需要考虑的问题是，正文中的词项数是很大的，如果我们想要邮箱地址得到和正文持平的地位，就要在计算的时候给邮箱的概率贡献乘上一个权重系数。

$weight = 1$

	accuracy	precision	recall	f1 score
min	0.995	0.995	0.9957	0.9961
max	1.0	1.0	1.0	1.0
avg	0.9982	0.9987	0.9985	0.9986

$weight = 10$

	accuracy	precision	recall	f1 score
min	0.995	0.995	0.9957	0.9961
max	1.0	1.0	1.0	1.0
avg	0.9982	0.9987	0.9985	0.9986

从结果上来看，效果比不加发送者有些许提升，由于本来准确率就很高，所以提升并不是很显著，另外当权重为1和10的时候也没有变化，我也理解为有可能剩下的极少数的错判的例子并不是依据发送者就可以区别出来的，在如此高的准确率上还要获得提升难度过高。

运行方法

将数据放至data文件夹下

运行`python3 TextFilter.py`可观察结果

命令行参数如下

```
-a 指定alpha参数大小
-s 开启平滑，若不开启则在零概率问题上取固定小概率
-m 是否考虑发送方
-p 设定固定小概率大小，默认为1e-2
```