基于朴素贝叶斯的垃圾邮件分类

2017011469 邱翊君

朴素贝叶斯原理

给定包含$x_1, x_2, \ldots, x_n$等n个特征的样本x（x为特征向量，$x = (x_1, x_2, \ldots, x_n)$），朴素贝叶斯分类的目标是确定该样本属于K个可能类别$y_1, y_2, \ldots, y_K$的概率$P(y_k|x)$，其中$k = 1, 2, \ldots K$。根据贝叶斯定理可得

$$P(y_k|x) = \frac{P(x|y_k)P(y_k)}{P(x)}$$

其中$P(y_k)$描述的是在不提供观测样本等相关知识的情况下y_k成立的概率，即先验概率，我们可以通过训练集得到。而$P(y_k|x)$称为后验概率，表示在已经观测到特征x的情况下，样本属于类别k的概率。$P(x)$称为证据，只依赖于特征的整体分布，而不依赖于特定的类别，因此是一个常数。

朴素贝叶斯假设n个特征相互独立，从而

$$P(x|y_k) = P(x_1|y_k)P(x_2|y_k)\ldots P(x_n|y_k)$$

因此朴素贝叶斯分类器得出，最有可能的假设

$$\begin{aligned} y_{NB} &= \underset{y_k}{argmax} P(y_k) \prod_i P(x_i|y_k) \\ &= \underset{y_k}{argmax} \{logP(y_k) + \sum_i logP(x_i|y_k)\} \end{aligned}$$

在本实验中，y只有两种分类，spam和ham；而特征向量x用邮件中所有单词出现的频率来描述。

朴素贝叶斯的实现

1. 数据集的划分

由于要实现五折交叉验证，所以首先将所有数据随机均分成5份，之后的5轮实验过程中，可以每轮取一折作为测试集，余下四折作为训练集。

`src/build.py`实现了这一功能，生成的五份数据集保存在`dataFolds/`目录下的`fold0`~`fold4`中.

2. 基于训练集的学习

首先，从训练集对应的4个文件中读取数据。对于每一封邮件，如果是非utf-8编码，即视为噪声，这里选择直接剔除。

然后，用python中的`email`模块对邮件内容进行解析，得到只包含邮件内容的字符串。再用`nltk`库中的英语分词将其分割为一个一个的单词，统计单词总数、单词出现的频率等相关数据。实验过程中，发现有一部分邮件中包含html格式的内容，于是用`BeautifulSoup`库将这些html内容解析为纯文本。

将统计得到的结果保存在`/trainingData/rate=?/`目录下（其中? 对应抽样比例，将不同抽样比例得到的数据放到不同的文件夹。例如，存储有单词-频数的字典，用python中的`json`库，排序之后直接存储成.json文件，便于之后测试时使用。

这部分功能由`src/train.py`实现。

3. 在测试集上验证

首先，读取对应的训练结果文件，构建好“单词-频数”字典等变量。

然后，对于测试集中的每一封邮件，按照之前同样的办法得到包含正文的单词列表。遍历每一个单词，作为一个x_i，从而根据前述算法作出分类。例如，$P(x_i|spam)$ = 在训练集中x_i在spam类出现的次数 / 训练集中spam类词语的总数

将得到的分类与实际分类进行比较。所有样本测试完之后，对结果作出评估。每一个样本有四种情况：

	实际为spam	**实际为ham**
被判为spam	真阳性	假阳性
被判为ham	假阴性	真阴性

对于结果，有四种评估的方法：Accuracy，Precision，Recall，F1，他们的定义如下：

1. Accuracy：准确率，即分类正确的样本所占的比例

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

2. Precision：精确率，被判为阳性的样本中正确的比例

$$precision = \frac{TP}{TP + FP}$$

3. Recall：召回率，实际阳性的样本中被判为阳性的比例

$$recall = \frac{TP}{TP + FN}$$

4. F1：综合考虑Precision和Recall，F-Measure是将Precision和Recall加权调和平均，当参数a取1时即得到F1

$$F = \frac{(a^2 + 1)PR}{a^2(P + R)}$$

$$F1 = \frac{2PR}{P + R}$$

在本实验的五折交叉验证中，每一轮都计算出上述四个指标，最后将无论的结果取算术平均值就得到最终的结果。

4. 分类器性能评估

完成以上步骤后得到的实验结果存储在`/res/rate=1/result.txt`中（不考虑邮件header，平滑策略alpha=1e-6，sampleRate=1）

epoch	accuracy	precision	recall	F1
0	0.899875	0.983951	0.851730	0.913079
1	0.905017	0.984021	0.860518	0.918135
2	0.904401	0.987997	0.859736	0.919415
3	0.905882	0.990685	0.856364	0.918640
4	0.909176	0.985320	0.865706	0.921648
average	0.904870	0.986395	0.858811	0.918183

issues

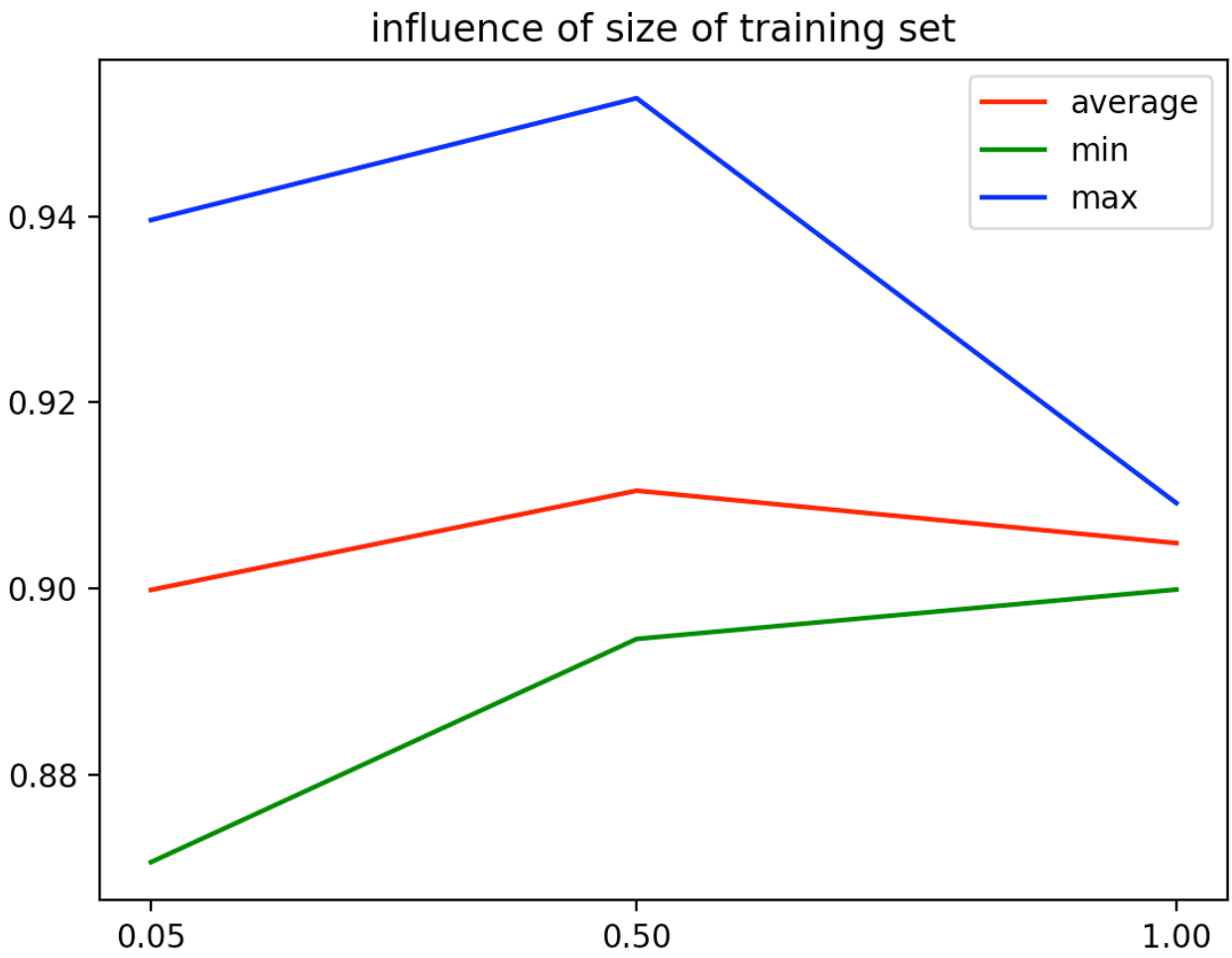
1. 训练集大小对实验结果的影响

理论上，由于朴素贝叶斯方法是基于特征出现频率的统计来进行分类决策的，以频率来估计概率，所以很自然的结果就是，当训练集规模越大时，某个有特定性质的词汇在某一个特定类别的邮件中出现的频率也就越大，也就更接近实际概率，进而分类结果也会更准确。

控制变量进行实验，均不考虑邮件header，平滑策略alpha=1e-6，分别选取训练集的5%、50%、100%进行实验。只需要修改`\src\train.py`中的`sampleRate`变量即可。评估指标选择accuray。

	0.05	0.5	1
0	0.897224	0.894573	0.899875
1	0.892279	0.900613	0.905017
2	0.939580	0.952674	0.904401
3	0.870588	0.902820	0.905882
4	0.899501	0.901841	0.909176
average	0.899834	0.910504	0.904870
min	0.870588	0.894573	0.899875
max	0.939580	0.952674	0.909176

可以看出，随着训练集规模的增加，模型准确率也基本是增加的趋势。但是也可以观察到，采用100%训练集时得到的平均准确率并不比50%的更高，非常符合过拟合现象。但是贝叶斯方法的特点是结合先验概率和后验概率，避免了只使用先验概率的主观偏见，也避免了单独使用样本信息的过拟合现象。于是进一步分析上述数据。



随着训练规模的增大，得到的准确率的方差越来越小，而且对最小值来说，准确率是严格递增的。因此可以得出结论，训练集较小时，模型的效果不稳定，在测试集上出现更大的准确率是正常的波动现象。总的来说，随着训练规模的增加，模型准确率越来越高且稳定。

2. 零概率

当测试样本中某一个单词在训练集的任一类中从未出现过时，$P(someword|spam)=0$，从而导致$P(x_i|spam)=0$，直接将该邮件错误的分为另一类。

为了避免这种情况，采用平滑策略处理：

$$Smoothing: P(x_i=k|y=c)=\frac{\#\{y=c,x_i=k\}+\alpha}{\#\{y=c\}+M\alpha}$$

其中，M是不同类别的数量，本实验中当然恒为2。α取0的时候，上式即为极大似然估计，因此应尽可能趋近于零，防止偏离模型太多。显然α的不同取值会对实验结果有影响，下面采用控制变量法对次进行分析。（不考虑邮件header，sampleRate=1，五折交叉验证）

α	**0.1**	**0.001**	**1e-6**	**1e-9**	**1e-12**
0	0.878197	0.893013	0.899875	0.901903	0.904866
1	0.883630	0.901714	0.905017	0.907061	0.909420
2	0.887995	0.901720	0.904401	0.907556	0.910238
3	0.887027	0.903787	0.905882	0.909428	0.910234
4	0.893258	0.906679	0.909176	0.912297	0.914014
average	0.886021	0.901383	0.904870	0.907649	0.909754

可以看出，随着α趋近于0，准确率越来越高。

3. 特殊特征

之前的实验过程中，只考虑了邮件正文部分，实际上数据集中还有很多有用的部分被丢弃掉了，比如邮件头中的一系列数据：

1. 发件人 / Received from...
2. 发送时间 / Time
3. 发送平台 / X-Mailer
4. 是否含有html / Content_Type

将它们纳入特征向量x中，其他实验条件不变（SampleRate=1，α=1e-6，五折交叉验证），最终得到实验结果如下：

epoch	**accuracy**	**precision**	**recall**	**F1**
0	0.982689	0.997672	0.974236	0.985815
1	0.983016	0.997919	0.974593	0.986118
2	0.986275	0.998732	0.979607	0.989077
3	0.982434	0.998401	0.973247	0.985664
4	0.981117	0.997147	0.972180	0.984505
average	0.983106	0.997974	0.974773	0.986236

和前面同样条件下不含这些额外特征数据的实验结果相比有显著的提升。