朴素贝叶斯实验报告

千英卓 2017013622

1 Experiment Design

在这次实验中，我使用朴素贝叶斯方法进行垃圾邮件的识别。我对邮件的主体部分使用词袋模型进行建模，将每个词视为一维特征，分别计算一篇邮件为ham和spam的likelihood。除了主体的词袋表示，我还手动构造了5种特征，将在Section 4.3详细介绍。

所有的实验采用5-fold cross validation，即将数据集随机分为5份，依次以每一份作为测试集，其他4份作为训练集进行训练。最后对Accuracy, Precision, Recall, F1-score报告为5次实验结果的平均。

1.1 Dataset

实验使用了课程提供的英文邮件集作为数据集。共有37822邮件，去除UTF-8无法解码的后剩余32401封。每封邮件除了主体部分，还包含了Received from, Date, Subject, To, From等meta information，并有ham或spam的标签。我保留了每封邮件的Received from, Date, Subject和主体部分用于提取特征。

1.2 Bag-of-words

将单词定义为符合Python正则表达式*[a-zA-Z]+*的字符串。事先扫描所有邮件，将频数不小于3的单词作为全集(这样虽然会引入测试集的信息，但由于单词集本身为可从外部获取的信息，故忽略这个问题)。统计每封邮件主体中出现的单词，记录每个单词频数或仅记录是否出现，分别记为*bow*和*bow_dedup*。

1.3 Naive Bayes

对于有特征$x_1, x_2, ..., x_n$和类别y的样本,由贝叶斯公式有

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y)P(x_1, x_2, ..., x_n|y)}{P(x_1, x_2, ..., x_n)}$$

若假设$x_1, x_2, ..., x_n$独立，则有

$$P(x_1, x_2, ..., x_n|y) = \prod_{i=1}^{n} P(x_i|y)$$

所以

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, x_2, ..., x_n)}$$

对于某个样本，$P(x_1, x_2, ..., x_n)$是固定的，所以y的预测值可如下计算

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i|y) \tag{1}$$

实际计算时，对式子右边取对数防止数值过小带来精度问题

在垃圾邮件识别中，类别y的取值为$ham, spam$。而根据词袋模型，所有的单词即为$x_1, x_2, ..., x_n$

根据训练集中单词在ham和spam中的频数来计算概率，即

$$P(x_i|y_j) = \frac{\#(x = x_i, y = y_j)}{\#(y = y_j)} \tag{2}$$

对于词袋模型以外的一些特征(Receive, Subject, Capital)，其条件概率计算类似。

1.4 Metric

对模型表现的评估采用以下四项指标：Accuracy, Precision, Recall and F1-score. 将预测为ham记为postive，则这四项指标计算方式如下

$$\begin{aligned} Accuracy &= \frac{tp + tn}{tp + tn + fp + fn} \\ Precision &= \frac{tp}{tp + fp} \\ Recall &= \frac{tp}{tp + fn} \\ F1\text{-}score &= 2 * \frac{Accuracy * Precision}{Accuracy + Precision} \end{aligned} \tag{3}$$

其中tp, tn, fp, fn分别代表真阳性，真阴性，假阳性，假阴性。

2 Results

以下是多次实验的结果，其中模型的具体设置将会在Section 3和Section 4说明。

Style	Accuracy	Precision	Recall	F1-score	Time
standard	0.9729	0.9569	0.9726	0.9647	49.31
no_deduplicate	0.9606	0.9252	0.9755	0.9497	103.83
case_insensitive	0.9720	0.9568	0.9704	0.9636	44.84
filtered	0.9715	0.9419	0.9859	0.9634	29.91

表 1: 数据预处理方式实验

表1中*standard*模型的处理方式为将频数不小于3的词收入词表，单词对大小写敏感，每封邮件记录单词去重。*no_deduplicate*对每封邮件出现的词不进行去重处理；*case_insensitive*匹

配单词时对大小写不敏感； *filtered*在词表中去除出现超过5000次的高频词。

表2中分别使用训练集的1%, 5%, 50%和100%进行训练。

Size	Accuracy	Precision	Recall	F1-score
1%	0.9152	0.8565	0.9378	0.8946
5%	0.9553	0.9303	0.9547	0.9423
50%	0.9699	0.9519	0.9700	0.9608
100%	0.9729	0.9569	0.9726	0.9647

表 2: 训练集大小实验

Alpha	Accuracy	Precision	Recall	F1-score
1.0	0.9729	0.9569	0.9726	0.9647
0.8	0.9724	0.9546	0.9741	0.9642
0.6	0.9712	0.9503	0.9754	0.9627
0.4	0.9690	0.9438	0.9770	0.9601
0.2	0.9676	0.9378	0.9799	0.9584

表 3: 平滑系数实验

表3中Alpha为式中系数α。

Feature	Accuracy	Precision	Recall	F1-score
bow	0.9729	0.9569	0.9726	0.9647
bow+receive	0.9755	0.9579	0.9789	0.9683
bow+subject	0.9813	0.9690	0.9824	0.9756
bow+date	0.9728	0.9567	0.9727	0.9646
bow+capital	0.9688	0.9579	0.9603	0.9591
bow+html	0.9725	0.9565	0.9720	0.9642
bow+rec+sub	0.9822	0.9698	0.9841	0.9769

表 4: 不同特征实验

表4为词袋模型加上不同的特征的实验。其中*receive, subject, date, capital, html*分别代表使用了通过Receive from, Subject, Date, 大写单词和HTML tag的信息构造出的特征(详

见Section 4.3)。

3 Analysis

我在实验中主要在构造词表和记录bag-of-word时遇到了问题。

一开始，我将所有的单词转为小写后处理，但后来考虑到邮件文本中大小写会有语义上的不同，所以改为记录区分大小写的单词。同样去掉频数低于3(在一封邮件中出现多次频数计为1)的词后，词表大小由原来的110k增加至170k。如表1所示，改动后Accuracy, Precision, Recall, F1-score均升高，同时耗时也增加。这是在意料之中的，因为增大词表后特征粒度更细，模型可以学到更多的语义信息，分类能力自然更强。而更大的词表在创建，遍历时时间开销更大，因此时间略微增加。

考虑到高频词语义信息往往低，于是尝试去除词表中的高频词(频数超过5000)。词表大小仅从176725降至176414，而程序运行时间减少了约40%，即主体中单词数量大概减少了40%。如表所示，去除高频词后Accuracy, Precision和F1-socre都有所降低，但Recall则有明显的增加。这说明模型更倾向于将邮件分类为positive(ham)，我猜测可能是因为高频词中有部分在spam中频率明显高的词，去除后导致被分类为spam的可能降低。

起初，我按照词袋模型的标准实现对每封邮件记录每个词出现的频数。然而由于不同邮件长度差别很大，且并不能假设ham和spam的长度分布大致相同，我猜测这样处理会更容易受噪声影响。将每封邮件的单词去重后(即只记录有或没有)，模型果然在Accuracy, Precision, F1-socre和速度上都有了显著提升。 Recall则略微下降，猜测可能是因为某些在ham中大量出现的词的频数被压缩，所以模型更倾向于预测negative。

另外需要说明的是，有另一种朴素贝叶斯方法将特征的维数定为词表的大小，以一个词是否在某篇邮件出现作为特征的取值。这样做对于每篇邮件虽然利用了更多信息，但特征数将会是我的实现的数百至数千倍，训练和预测开销将大大增加。此外，考虑到目前的最佳Accuracy已达到98.22%，即使有所提升空间也很小，所以不采用这种模型。

4 Discussion

4.1 Size of Training Set

通过实验验证训练集大小对模型效果的影响。对于100%以外的每个size，均将训练集随机选取重复五次取均值。

由表2可见，四项指标均随着训练集增大而显著增加，符合预期。同时注意到，仅使用5%的训练数据就可以达到95%以上的正确率，这说明以单词作为特征提供的信息非常强，模型在少量数据下就可以学到很好的效果。

4.2 Zero-Probabilities

在我们目前对条件概率的计算中，若某个词x在某个类别y中没有出现过，则其概率会被统计为0. 即代表，若一封邮件出现了x，就必不可能被预测为y，这显然是不合理的。于是，需要对条件概率进行平滑化处理，避免出现概率为0的情况。

我采用了Lidstone's law进行平滑处理，将(2)式修改为

$$P(x_i|y_j) = \frac{\#(x = x_i, y = y_j) + \alpha}{\#(y = y_j) + M\alpha} \tag{4}$$

其中M, α为参数。

不同α下的实验结果见表3。可见，α越大，模型效果越好。实际上从实验结果看来，在很大的范围内，模型效果均是与$M * \alpha$取值正相关的。

M * alpha	Accuracy	Precision	Recall	F1-score
2	0.9528	0.9023	0.9826	0.9407
10000	0.9667	0.9390	0.9760	0.9571
25920	0.9728	0.9569	0.9726	0.9647
50000	0.9757	0.9667	0.9699	0.9683
100000	0.9772	0.9726	0.9675	0.9700
200000	0.9773	0.9745	0.9658	0.9701
500000	0.9774	0.9759	0.9647	0.9703

表 5: 对$M * \alpha$取值的实验

根据助教的PPT，M的取值应为类别的数量，即为2。然而事实上M取2的效果远不如M取极大的值，下面尝试通过分析M取2与M取100000时概率的区别来进行解释(假设$\alpha = 1$)：
对于x_i，$lnP(y = ham|x_i)$的差值为$ln\frac{N_ham+100000}{N_ham+2}$，$lnP(y = spam|x_i)$的差值为$ln\frac{N_spam+100000}{N_spam+2}$。
其中N_ham, N_spam分别为ham，spam的数量。根据统计，spam占所有邮件的65.9%，约为ham的两倍，所以N_ham约为8000， N_spam约为16000.

$$\Delta lnP(y = ham|x_i) \approx ln\frac{N\_ham + 100000}{N\_ham} \approx ln13.5 \approx 2.6$$

$$\Delta lnP(y = spam|x_i) \approx ln\frac{N\_spam + 100000}{N\_spam} \approx ln13.5 \approx 1.98$$

即M取2比取100000时每个单词得到的ham的对数似然值比spam高0.6，相比之下越长的邮件越容易预测为ham。而ham本身在数据集中占比较少，且根据生活经验，spam中长邮件的比例会更高，所以M=2时表现更差。

4.3 Specific Features

在主体部分的词袋模型以外，我还采用了以下5种特征。

4.3.1 Receive

通过Receive from的meta信息构造特征。具体做法是使用正则表达式提取每条Receive from中from之后最近的域名(或unknown，或localhost)，对于提取不到的标记为UNK。预处理数据集，获取域名表。将每个域名当作一个特征，在训练和预测时使用和词袋中的单词同样的方法。

如表4所示，加入域名特征后四项指标均有所增加，其中Recall增加尤为显著。我猜测这是因为如"edu", "localhost"等域名值为ham的概率非常高，起到了白名单的作用。

4.3.2 Subject

通过Subject的meta信息构造特征。处理方法与主体部分完全相同，但另外构造了词表以节省空间。由实验结果可见，通过Subject构造的特征对分类效果有非常显著的提升。体现标题所含语义信息十分强，符合预期。

4.3.3 Date

通过Subject的meta信息构造特征。使用正则表达式提取发送时间，分为工作日凌晨(0:00-8:00)，工作日白天(8:00-16:00)，工作日晚上(16:00-24:00)，周末凌晨，周末白天，周末晚上六个时段。加上未成功提取的UNK，作为一个有七种取值的特征参与计算。考虑到可能有时差和网络延时等因素，很难认为发送时间与是否为垃圾邮件相关，而实验结果页证明Date特征对模型的表现几乎没有影响。

4.3.4 Capital

通过主体部分所有大写单词构造特征。使用正则表达式提取邮件主体的大写单词，构造词表后将每个单词作为一个特征计算。构造这个特征是考虑到spam中可能有较大量的大写单词，然而结果证明这个特征主要是负影响。猜测原因可能是区分大小写的词袋模型已经较好地提取了大写单词相关的信息，而少量的纯大写单词噪声较大，降低整体表现。

4.3.5 HTML

通过主体部分的HTML tag构造特征。使用正则表达式提取主体部分的HTML tag，若超过6个则将此邮件标记为带HTML，将是否带HTML作为一个特征参与计算。构造这个特征是因为发现有较多的spam均带有很多HTML tag，然而实验结果并不理想，猜测是因为是否带HTML与是否是spam本身并不相关。