Lab1 朴素贝叶斯分类器

实验设计

基本原理

样本：提供的数据集包括n封邮件原文和他们对应的真实类别（"spam" or "ham"），我们不妨形式化的定义为 (x_k, l_k) ，其中 x_k 表示邮件原文，l_k 表示类别，由于本次任务是一个二分类模型，且考虑到大部分实际应用中分类器的目的都是分辨出垃圾邮件，因此本次实验中定义 "spam" 为正类

先验分布：某个邮件是垃圾邮件或非垃圾邮件的概率，即 $P(l_k = \text{“}spam\text{”})P(l_k = \text{“}ham\text{”})$，在贝叶斯框架下这是基于历史数据或专家经验得到的，本次实验中我们通过统计数据中二者的频率来估计先验分布

采样（似然）分布：在给定模型参数下数据的概率分布。在垃圾邮件分类器中，这对应于在已知邮件类别的情况下，观察到特定单词出现的概率。朴素贝叶斯分类器依赖于一个简化假设，即特征之间相互独立。假设通过原样本提取到 m 个特征 $(x_k^1, x_k^2, \cdots, x_k^m)$ ，在本实验的基础部分中，特征即邮件正文经过分词后得到的单词构成的词袋模型，$x_k^i = x$ 表示第k个样本中词表中第 i 个词出现了了 x 次

后验分布 ：在观察到数据之后，对模型参数的概率分布的更新。根据贝叶斯定理，后验分布是先验分布和似然的乘积标准化后得到的。即 $P(l_k = l|x_k^1, x_k^2, \cdots, x_k^m) = \frac{P(l_k=l)P(x_k^1, x_k^2, \cdots, x_k^m|l_k=l)}{P(l_k=l, x_k^1, x_k^2, \cdots, x_k^m)}$

由于测试阶段只需要输出分类结果 l_k，而不是计算具体的后验概率值，因此分母并不需要真正被计算，而只需要求解 $\hat{l} = argmax_l P(l)P(x_k^1, \cdots x_k^m|l)$ 即可

数据预处理

数据集一共有37822个样本，其中 spam 和 ham 的比例大致为 2:1，对于这些数据我进行了以下处理。

1. **筛选编码**：通过后面的实验可以发现训练增加样本量会对模型性能带来较大的提升，因此在筛选那些不可编码的样本量时需要格外谨慎，这里我采用了 'utf-8', 'gbk', 'iso-8859-1', 'windows-1252′ 这四种常见编码对数据进行编码，并剔除那些无法被编码的字符。
2. **去除邮件头**：通过观察发现邮件除了正文以外有很多邮件头等信息，考虑到这些特征不应该作为正文信息进行分词处理，因此我在基础实验中筛选掉了这些信息。
3. **去除停用词**：通过检查我发现经过分词后有很多无意义的词汇，如 "the"，"a" 等，这些词出现的频率极高但并不能提供很多信息，出于减少文本数据集的大小和复杂性，让算法能够更加专注于那些具有实际意义和分析价值的词汇的考虑，去掉这些常见停用词

训练集与测试集划分

本次实验中我采用五折交叉验证的方法，即每次从预处理后的数据集中取 1/5 作为测试集（约 7000），剩余部分作为训练集训练，将五次结果取均值作为最终指标

实验结果与分析

在推理时我采用的模型假设为：$P(x_k^1, x_k^2, \cdots, x_k^m | l_k = l) = \Pi_{i=1, x_k^i > 0}^m P(x_k^i | l_k = l)$，这里增加了额外的限制条件，仅统计在该样本中词频大于零的特征。增加这一限制条件的初衷在于考虑到词表中非零值相当稀疏，大量值为零的特征并不会带来的足够的信息。此外，仅统计非零值也可以大大加快推理的速度。在下面的实验中我也将进一步证明这一假设并不会导致模型性能的损失。

而在实际算法中，我利用对数的保序性，对于原模型做了对数似然的等价变化，$\hat{l} = argmax_l P(l) P(x_k^1, \cdots x_k^m | l) = argmax_l log(P(l)) + \sum_i^m I(x_k^i > 0) logp(x_k^i | l)$，这一变换的等价性是显然的，在后面的实验中将说明这一做法对实际算法执行的优势。

本次实验中我选取了如下指标衡量模型效果

- **Accuracy**：分类正确的概率

  $Accuary = \frac{TP+TN}{TP+FN+FP+TN}$
- **Precision**：所有预测为正例的结果中真阳性的比例（本次实验中定义 "spam" 为正类）

  $Precision = \frac{TP}{TP+FP}$
- **Recall**：所有真实标签为正例的结果中真阳性的比例

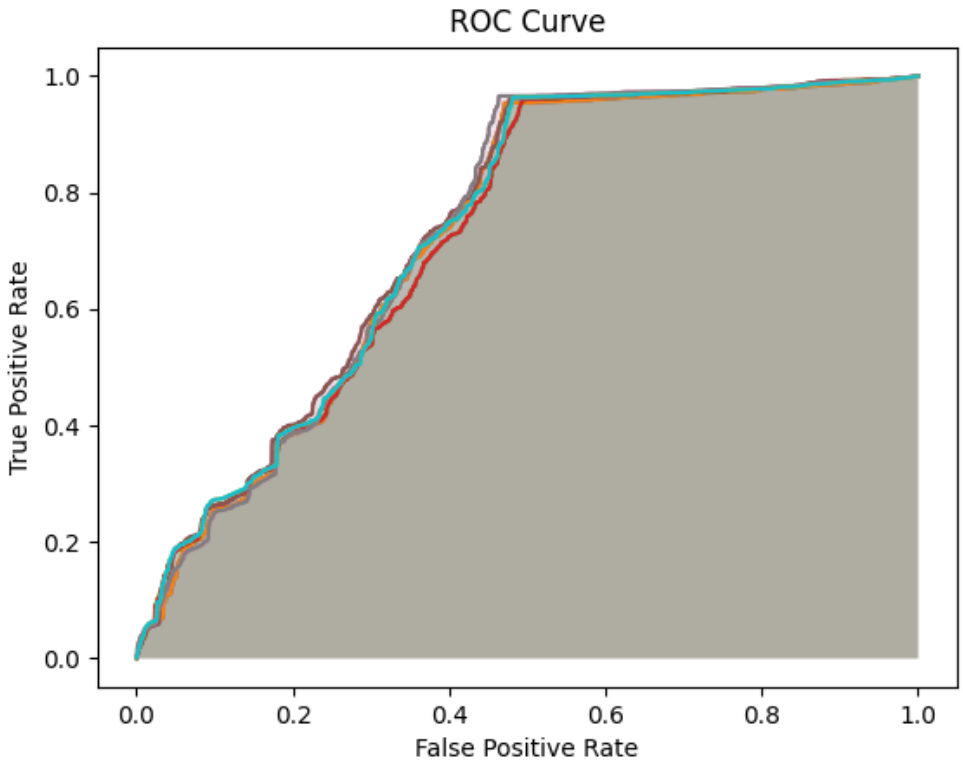
  $Recall = \frac{TP}{TP+FN}$
- **F1**：精准率和召回率的加权调和平均 ，公式和课件一致，取 $\beta = 1$

  $F_1 = \frac{2TP}{2TP+FP+FN}$
- **AUC**：ROC下方的面积，也可以解释为随机选取一正负样本对，正样本比负样本预测概率高的概率

下表为五折交叉验证中每一折的结果以及平均结果:

实验	Accuracy	Precision	Recall	F1	AUC
1	0.951	0.979	0.944	0.961	0.728
2	0.955	0.985	0.945	0.965	0.726
3	0.955	0.983	0.947	0.965	0.738
4	0.963	0.985	0.956	0.970	0.733
5	0.959	0.984	0.952	0.968	0.733
平均值	**0.957**	**0.983**	**0.949**	**0.966**	**0.732**

下图为五次训练的ROC曲线，其下方所围成面积即为AUC



实验分析：

1. **从准确率来看**，95.7%的邮件都能被正确分类，且在五组实验中保持了较小的波动，这表明基于词袋模型设计的朴素贝叶斯分类器在垃圾邮件检测任务上表现良好且稳定。
2. **观察精确率，召回率和 F1 分数**，模型在这三项指标上都能表现良好，这说明垃圾邮件识别中的假阴性和假阳性占比较少。特别的，精确率相比召回率效果更好，这也和课上讨论的垃圾邮件识别任务中对假阳性和假阴性容忍度差异相符合：在垃圾邮件识别任务中，避免将正常的邮件误判为垃圾邮件相比未成功过滤掉部分垃圾邮件是更为严重的问题，因此**精确率较高说明模型在避免假阳性上表现很好，符合垃圾邮件识别任务的场景。**F1的结果综合考虑了精确率和召回率，F1较高说明模型综合性能优秀
3. **从 ROC 曲线和 AUC 值来看**，在评分上，模型在更为细化的评价尺度上表现较好，即更有可能给垃圾邮件更高的评分，分数靠后的一半基本都是负例。
4. 为了验证零特征信息是否对模型性能提升有帮助，我对比了在推断时统计完整词表和仅统计非零特征（基础设定）两种推断模型性能结果，可以看到，除了 AUC 上前者有较大优势外，仅统计非零向量表现基本优于统计完整词表或与之持平，这证明了限制条件的增加是合理的。

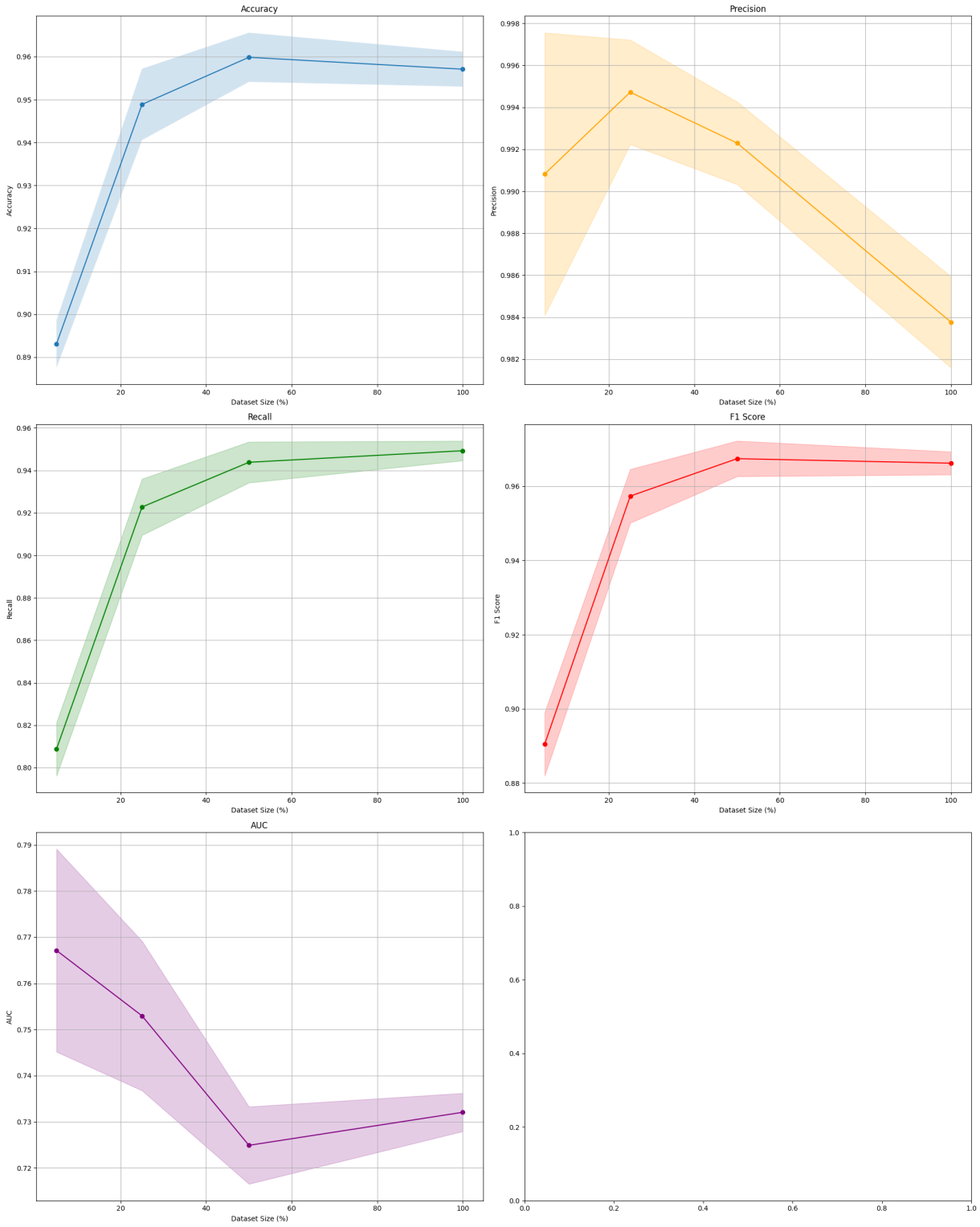
	Accuracy	Precision	Recall	F1	AUC
基础设定	**0.957**	**0.983**	0.949	**0.966**	0.732
考虑完整词表	0.950	0.969	**0.953**	0.961	**0.748**

5. 为了证明对数似然使用的必要性，我对比了采用连乘方式计算联合分布的计算方法，实验结果如下，可以看到，对数似然相比直接计算联合分布有明显的优势，其原因很可能在于贝叶斯后验分布中分子项很小，连乘易导致计算超出精度变为0，因而失去关键信息。这一点可以结合下面的零概率问题进一步证明：在使用连乘方法计算联合分布时，约有25%的联合概率给出了0的结果，而即使通过引入平滑项 α 解决零概率问题后，仍有10%的联合概率为0，这就是计算超出精度的结果。而在对数似然中由于通过取对数将乘法运算转换成加法运算，因此不存在这种问题，更多细节的信息能够被保留下来。

	Accuracy	Precision	Recall	F1	AUC
对数似然	**0.957**	**0.983**	**0.949**	**0.966**	**0.732**
直接算联合分布	0.926	0.982	0.902	0.941	0.719

问题1：训练集大小对分类器的性能有着怎样的影响

本节中，我选取了数据集的 5%，25%，50%，100% ，并保持其它实验设计参数不变的情况下，测量了模型的各项指标随数据量的变化趋势，其中折线为五折一实验中均值的变化趋势，阴影表示每组实验五次结果的方差



- **准确率**： 准确率存在明显的趋势，整体来看，**随着数据集规模的增大，准确率也在提高，在50%训练数据集时达到最大**。从50%到100%，模型准确率略有所下降，但实验方差减小，说明实验逐渐趋于稳定

- **精确度：**在不同数据集大小的情况下，精度通常保持较高水平，但随着数据集大小从 50%增加到100%，有略微下降趋势，且方差略有所增加，这表明在数据集规模较大时，精确度可能对数据集或模型配置的变化更为敏感。其原因可能是因为精确度一直保持在一个很高的水平，而随着数据量增加，信息的复杂程度和噪声都随之增加，因此假阳性不可避免的有所增加，导致精确度不能再保持在极高的水平，并且组内实验波动也有所增加。
- **召回率**：随着数据集规模的扩大，召回率也呈上升趋势，这与准确率类似。而与精确度相比，单组实验中召回率的方差更大（坐标轴比例尺有区别，不能只看阴影面积），这意味着模型识别垃圾邮件的能力不如避免假阳性的能力那么一致。此外，和精确度相比，召回率对数据集大小更敏感，当数据量大小为5%时，召回率极低。
- **F1 分数**：作为兼顾了精确度和召回率的指标，F1 分数随数据集的大小而提高，在 50% 到 100% 的范围内，阴影方差区域较窄且平均值较大。这表明，当数据集至少达到全部数据集的二分之一时，F1 分数可以在各次试验中保持稳定。
- **AUC：**AUC 曲线图显示出更大的波动性，在数据集规模达到 50%时出现明显下降。这里较大的差异可能的解释是，模型的预测排序能力可能受数据的特殊性或模型训练过程中其他随机因素的影响较大。

总体来说，**随着数据量的增加，除精确率和AUC外，模型的各项性能指标和稳定性都会带来提升**，但由于数据量增加也可能带来一些噪声数据，因此对于精确率这种起点很高的指标来说数据量的增加带来的增益并不明显。AUC指标相比其他指标波动较大且没有明显规律，这可能说明在这项任务中，AUC并不是一个很好的评价指标。

问题 2: 零概率问题

问题阐述：

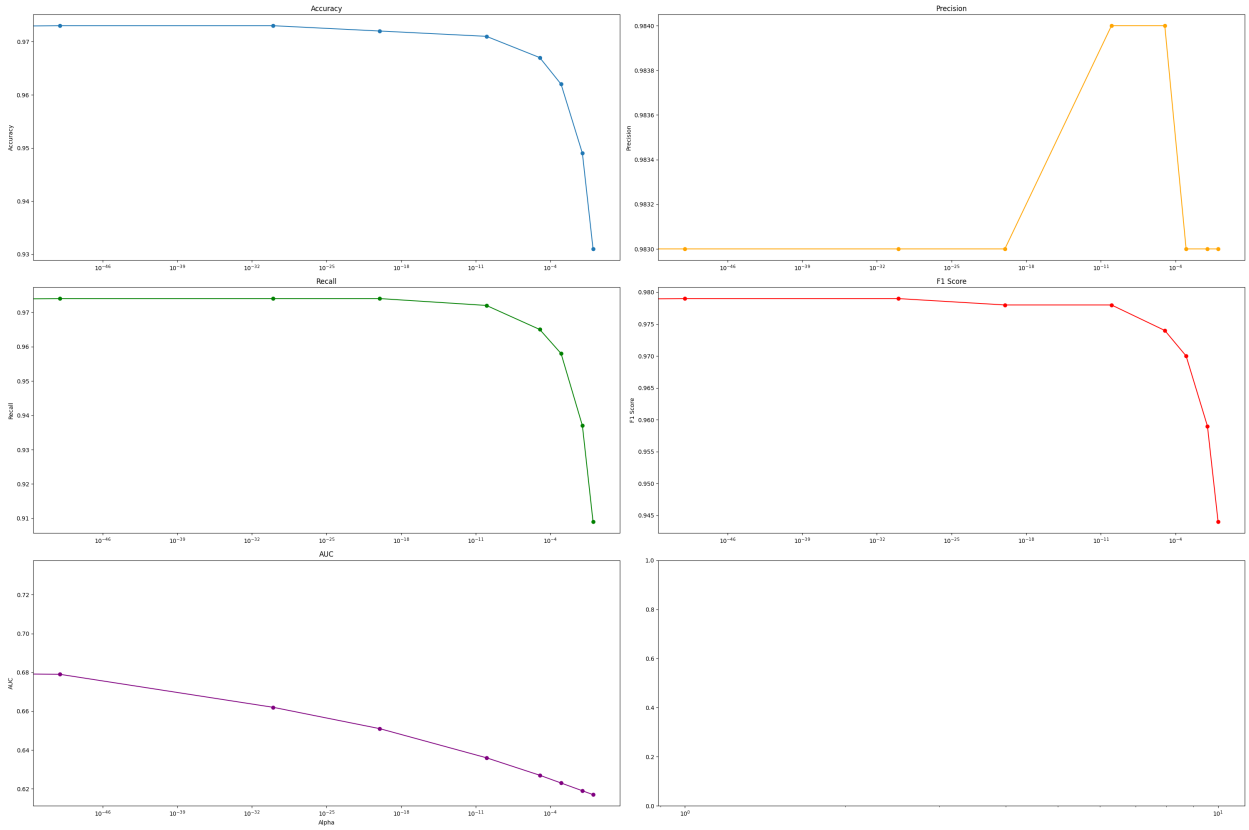
假设在训练集中没有样本出现 $x_k^i = j_i, l_k = l$，那么在统计特征时会导致$P(x_k^i = j_i | l_k = l) = 0$，而由于之后的计算是有边缘分布连乘积计算联合分布，那么会导致估计的联合概率 $P(l_k = l | x_k^1 = j_1 \cdots x_k^i = j_i \cdots x_k^m = j_m) = 0$，这意味着某一两个词的异常频率会对整体的联合概率产生极大的影响，使得模型鲁棒性差，无法对于垃圾邮件中的波动做出合理的反应。

通过对原模型预测得分可以看到，由于零概率现象的存在，有10%-25%的测试样本的后验概率为0，这一结果说明了零概率问题存在的广泛性。

基于课件所提出的解决方法，我们给出平滑概率公式 $P(x_k^i = j_i | l_k = l) = \frac{\#(x_k^i = j_i, l_k = l) + \alpha}{\#(l_k = l) + M\alpha}$，其中 $M = 2$ 表示类别数量。下面是尝试不同 α 下模型性能的比较

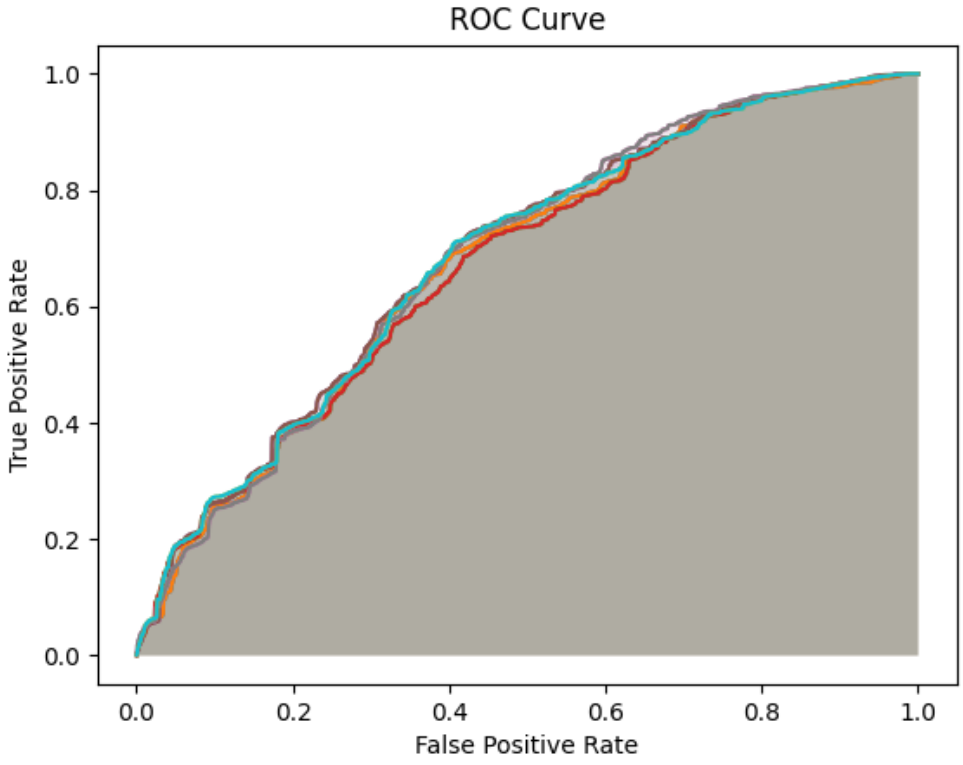
α	Accuracy	Precision	Recall	F1	AUC
0	0.957	0.983	0.949	0.966	0.732
1e-50	**0.973**	**0.983**	**0.974**	**0.979**	**0.679**
1e-30	0.973	0.983	0.974	0.979	0.662

α	Accuracy	Precision	Recall	F1	AUC
1e-20	0.972	0.983	0.974	0.978	0.651
1e-10	0.971	0.984	0.972	0.978	0.636
1e-5	0.967	0.984	0.965	0.974	0.627
1e-3	0.962	0.983	0.958	0.970	0.623
1e-1	0.949	0.983	0.937	0.959	0.619
1	0.931	0.983	0.909	0.944	0.617



可以看到 α 在0到1之间时平滑概率对性能的提升最为明显，且在 α 较小$(1e-50, 1e-30)$ 时对模型提升较为明显。

通过ROC曲线可以看到，和基础实验相比，ROC曲线相对平滑，说明原来得分较低的那部分负例得分被更均匀的分散到了正例之中，这使得ROC曲线更加平滑但也导致AUC值有所下降



问题 3: 特征设计

本节中我尝试引入了四个特征：

- **发送时间：**包括发送时的小时（0~23）和发出的日期是星期几
- **Xmailer字段：**这一字段表示消息从哪个客户端发出出来的，通过对数据集的观察，我将客户端选择范围限定在了Outlook，devMail，Frobozz，Foxmail，Bat，其余用"None"代替
- **Received 字段：**这一字段表示消息的路由信息，记录了邮件传递过程，这里我构建特征的方式为：如果该字段有多个路由，仅选择一个，将其类似于词表一样整理
- **正文编码方式**：除了课上给出的提示外，我还尝试了将正文编码方式融入到模型中，这一尝试的想法源自于构建数据集时的观察：**凡是无法用utf-8编码打开的，往往都是垃圾邮件**，考虑到原模型的精确率已经相对较高，因此尝试提升其召回率是更好的选择，而基于数据集的观察又刚好可以帮助我们将那些其他特征不易区分出来的垃圾邮件辨别出来，也即提升召回率。

用公式表示，当引入特征 s_k 时，贝叶斯求解公式变为

$$\hat{l} = argmax_l P(l)P(x_k^1, \cdots x_k^m, s_k|l) = argmax_l log(P(l)) + \sum_i^m I(x_k^i > 0) logp(x_k^i|l) + log(P(s_k|l)) \cdot w$$

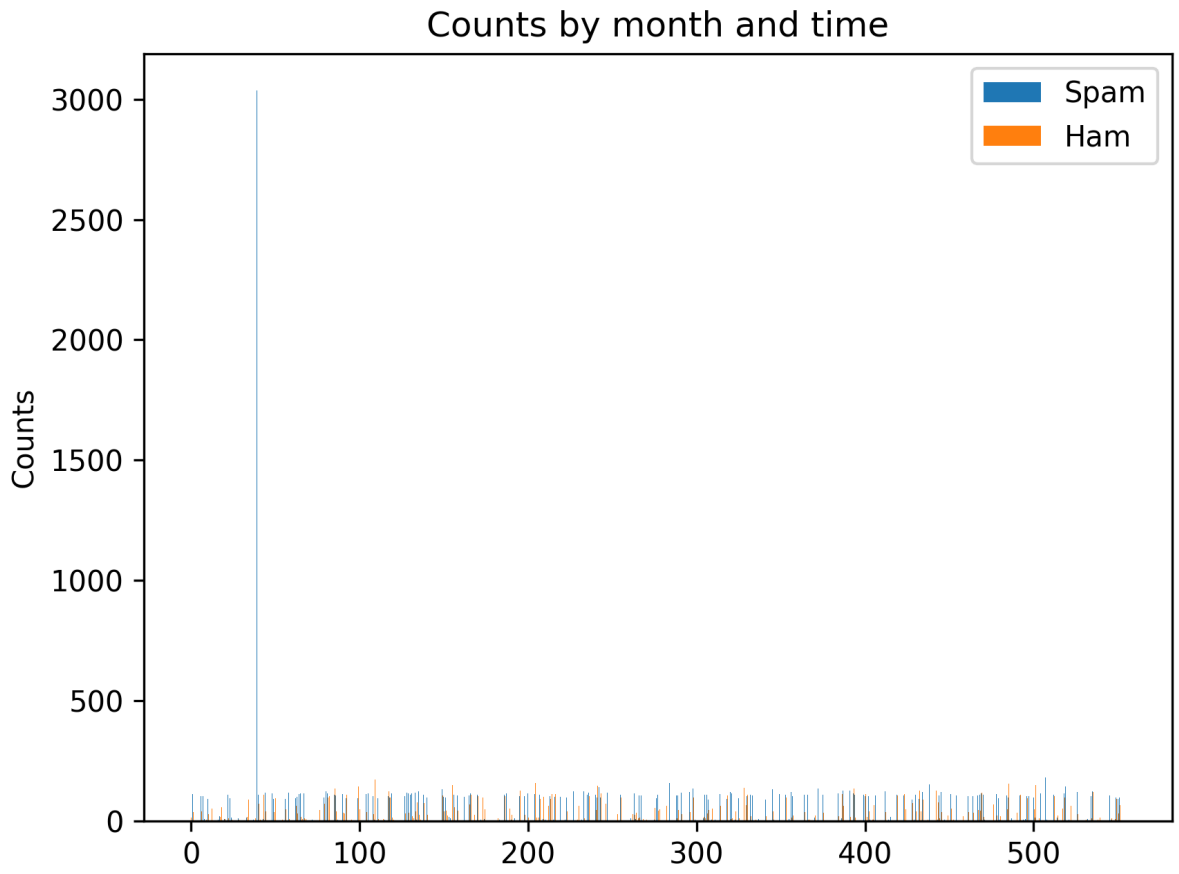
其中 w 为超参数，表示该特征的权重，下面的实验中我们统一选择，$w = 2$，这样做一方面可以突出加入特征后的分类效果变化（相比$w = 1$，前者更能凸显特征引入前后的变化），另外一方面也可以避免在测试集上调参的嫌疑

实验结果：

特征	Accuracy	Precision	Recall	F1	AUC
词袋模型($\alpha = 1e-50$)	**0.973**	**0.983**	**0.974**	**0.979**	**0.679**
+ 时间	0.966	0.992	0.954	0.973	0.679
+ 编码方式	0.974	0.983	0.977	0.980	0.678
+ Xmailer	0.957	0.997	0.937	0.966	0.677
+ **Received**	0.975	0.989	0.972	0.980	0.680

分析：

- **时间特征**：引入时间特征除了对提升精确度有所帮助外，对于其它指标都没有明显帮助，究其原因，从常理上来讲，垃圾邮件的出现频率确实和时间没有必然联系，引入时间信息带来的信息增益并不明显。通过对数据分析的结果也可以得到相似的结论，除了“None”出现频次较高外，其它的时间在两种label中分布都十分均匀。不过引入时间特征以后，模型的精确度得以进一步提高，基于此前的讨论，高精确度对于垃圾邮件识别任务十分必要



- **编码方式**：正如前文所分析的，编码方式的引入提高了垃圾邮件的召回率，并且精确率没有下降，这使得模型整体的得以提升。
- **Xmailer**：和时间类似，Xmailer的引入可以提升模型的精确率，但也会导致召回率的下降，因此整体性能并未上升。
- **Received**：引入 Recieved 后模型除了召回率有轻微下降外整体性能都得以提升。

综上，编码方式和Recieved特征对于模型性能提升有正向的帮助，在同时加入这两个特征以后模型性能如下，在两个特征的帮助下，模型的准确率和精确度均提高了0.5%左右

特征	Accuracy	Precision	Recall	F1	AUC
词袋模型($\alpha = 1e-50$)	**0.973**	**0.983**	**0.974**	**0.979**	**0.679**
+ 时间 && Received	0.977	0.989	0.974	0.982	0.679

受问题一的启发，我还进一步探究**了不同样本量下引入这些新的特征对模型的影响**。实验结果如下

Accuracy：

样本量	5%	25%	50%	100%
词袋模型($\alpha = 1e-50$)	0.968	0.985	0.985	0.973
+ 时间 && Received	0.963	0.984	0.986	0.977

Precision:

样本量	5%	25%	50%	100%
词袋模型($\alpha = 1e-50$)	0.992	0.994	0.992	0.983
+ 时间 && Received	1.0	0.994	0.994	0.989

Recall:

样本量	5%	25%	50%	100%
词袋模型($\alpha = 1e-50$)	0.948	0.981	0.984	0.974
+ 时间 && Received	0.931	0.979	0.983	0.974

F1 Score:

样本量	5%	25%	50%	100%
词袋模型($\alpha = 1e-50$)	0.969	0.987	0.988	0.979
+ 时间 && Received	0.964	0.987	0.988	0.982

AUC:

样本量	5%	25%	50%	100%
词袋模型($\alpha = 1e-50$)	0.692	0.669	0.649	0.679
+ 时间 && **Received**	0.691	0.668	0.649	0.679

分析：样本量较小时，引入新的特征对于模型性能并没有正向的提升，这可能是因为词袋模型本身的能力还没有达到极限，而在少样本情况下这些新的特征带来的增益要小于词袋模型本身，因此引入新的特征对模型性能并没有什么太大的帮助。当样本量达到50%以上后，词袋影响对模型的性能增益已经用尽，接下来的重要工作是如何辨别训练集中因为样本量增加而不可避免产生的噪声，而此时引入这些人为特征就可以帮助模型在词袋模型边际效益递减时再提升一点性能。

综上，通过人为引入一些（筛选后的）特征，可以提升模型的性能，而且这种性能提升的成本很低（相比词袋模型庞大的词表，这些特殊特征的维度往往很小），此外在样本量比较小的时候，如何最大程度上吃词袋模型的红利，而不是引入新的特征，对于提升模型性能更为重要。