Introduction to Machine Learning
Naïve Bayes Classifier
Report

计 86　任一 2018011423

2020 年 3 月 30 日

实验环境	
操作系统:	Windows10 家庭版 18362.72
Python 版本:	Python 3.7.3

1 实验概述

1.1 实验简要思路

本实验使用了朴素贝叶斯分类器，实现了对垃圾邮件分类的任务，简要的思路如下。

首先我对邮件进行了预处理，提取出邮件的标题、发件邮箱、收件邮箱、发送日期、是否是垃圾邮件的标签整理到 1 个 csv 当中方便读取和使用。

之后我进行了朴素贝叶斯分类器的设计，我主要选取的 feature 是发件人邮箱地址、收件人邮箱地址、邮件标题、邮件正文这 4 种 feature。

最后我对朴素贝叶斯分类器模型进行了测试，测试方法是 5 次 5 折交叉验证，测试指标是 Accuracy, Precision, Recall, False Positive Rate [1] , F1 值这 5 项。输出结果包含 5 次 5 折交叉验证产生的 25 次结果及其平均值、最小值和最大值。

1.2 文件说明及运行方式

email_info_ascii.csv 中储存着从所有 ASCII 编码的邮件中提取出的关键信息[2]。filter.py 是整个分类器的主要框架，core_function.py 中是分类器抓取邮件特征词、获取条件概率并进行朴素贝叶斯分类预测的核心代码。

若想运行代码，只需在 filter.py 所在文件夹下命令行中运行"python filter.py" 即可 (需要 Python3 环境以及相关库)

2 3 个 ISSUE

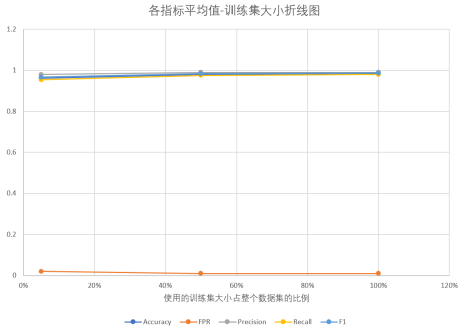
本部分主要回答实验 PPT 中提出的 3 个 ISSUE

2.1 Issue 1: the size of training set

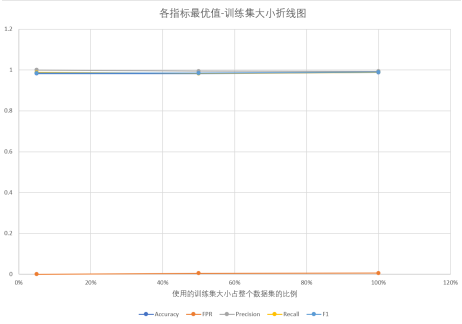
针对这个问题，我分别使用 5%、50%、100% 的数据进行 5 次 5 折交叉验证，并且考察这 3 种数据规模下的每一折测试中模型各指标的平均值、最优值和最差值，数据表格以及折线图如下。

[1]在评估时，我们认为垃圾邮件是正例，正常邮件是负例。因为将正常邮件判断为垃圾邮件的损失可能较大，我们单独拿出 False Positive Rate 作为一个指标，简记为 FPR. FPR = 1 - Precision.

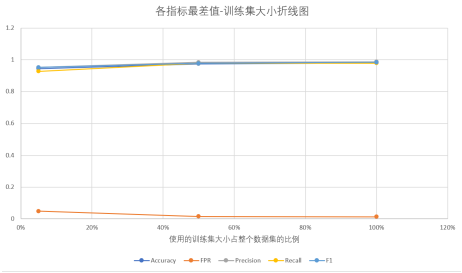
[2]为了使数据方便处理，使用 Python 的 chardet 模块判断邮件编码，并且仅使用 ASCII 编码的邮件，共 28986 封



Average Result			
	5%	50%	100%
Accuracy	0.962	0.98	0.985
FPR	0.02	0.0099	0.0098
Precision	0.98	0.99	0.99
Recall	0.954	0.975	0.98
F1	0.967	0.983	0.987



Best Result			
	5%	50%	100%
Accuracy	0.983	0.983	0.988
FPR	0	0.0049	0.0058
Precision	1	0.995	0.9942
Recall	0.988	0.9838	0.9889
F1	0.985	0.9853	0.9898



Worst Result			
	5%	50%	100%
Accuracy	0.945	0.975	0.9812
FPR	0.048	0.0158	0.0129
Precision	0.952	0.9842	0.9871
Recall	0.928	0.9782	0.979
F1	0.952	0.978	0.9836

从上述图表中可以看出，朴素贝叶斯分类器在垃圾邮件识别任务中取得了较好的分类结果，最高的识别准确率达到了 98.8%。随着训练数据规模的增大，模型性能出现小幅提升。但总体而言，数据规模对于模型性能的影响不是特别大。

具体来说，我们可以从表格的数据中更为详细地看到，随着数据规模增大，模型平均性能得到了小幅提升，只选取 5% 的数据时，模型平均准确率为 96.2%, 而选取全部数据时模型平均准确率上升到了 98.5%。同时也可以看出，随着数据规模增大，模型的最好性能和最坏性能之间的差距缩小了，

即模型的性能更加稳定了。这样的变化可以解释为，数据规模增大后，模型学习到了更多的数据特征，从而能够在性能和稳定性的角度都有所提升。

经过进一步的思考以及资料的查找[3]，我学习到了朴素贝叶斯分类器的优点之一就是所需的训练数据并不多 (相比神经网络等方法)。此外，朴素贝叶斯分类器还有实现简单、可解释性强、理论基础扎实等优点，并且在文本分类当中往往具有较好的表现，这也在一定程度上解释了上述测试得到的结果。

2.2 Issue 2: zero-probabilities

零概率问题是朴素贝叶斯分类器公式中不可避免会遇到的一个问题。出现的原因解释如下：我们记 x_i 为选取的特征，h 为假设，需要计算的是 $\arg\max_h((\prod_{i=0}^{n} P(x_i|h))P(h))$. 在训练过程中，我们能够求得 $P(h)$ 以及一部分的 $P(x_i|h)$. 然而，如果在测试时，遇到测试样例中，存在某个特征 x_i 在训练中没有出现，或者存在某个 x_i 在给定假设 h 下，出现次数为 0. 上述两种情况均会导致 $P(x_i|h) = 0$. 这就是零概率问题。

零概率问题会在一定程度上使模型预测出现偏差，原因如下：若存在某个特征 x_i 使得 $P(x_i|h) = 0$，则计算的 $\prod_{i=0}^{n} P(x_i|h)P(h)$ 就会变为 0，这样一来就会使得测试样例中其他特征 $x_j(i \neq j)$ 在本次预测中失去效果，从而影响模型的表现。

因此我们需要对 $P(x_i|h) = 0$ 的项做平滑处理。经过一些资料的查阅和学习，我尝试了两种平滑方式。一种是对 $P(x_i|h) = 0$ 的项，直接赋予一个较小的概率，例如令 $P(x_i|h) = \beta$, 在这里可以令 $\beta = 10^{-50}$(我们记为小概率平滑). 另一种做法是令 $P(x_i|h) = \frac{\#(x_i,h)+\alpha}{\#(h)+M\alpha}$, 其中 α 是平滑系数 M 为分类的类别数，在本实验中 $M = 2$(我们记作 add α 平滑). 下面我对上述两种平滑方式做了比较实验 [4]，

通过 10% 的数据进行 5 次 5 折交叉验证，我分别测试了这两种平滑方式下的模型平均性能，数据表格如下。

由表 1 可以看出. 这两种平滑方式当其参数都为 10^{-50} 时，模型性能都较好且差别不大。

我还继续考察了不同平滑常数下的模型性能，并尝试进行了一些分析。

[3]此处参考 Wikipedia `https://en.wikipedia.org/wiki/Naive_Bayes_classifier`

[4]由于朴素贝叶斯公式中的连乘容易造成数据下溢，模型中涉及 $\arg\max_h((\prod_{i=0}^{n} P(x_i|h))P(h))$ 的计算均转换为取对数后的计算，即 $\arg\max_h((\sum_{i=0}^{n} log(P(x_i|h))) + log(P(h)))$

表 1: 两种平滑方式下模型性能比较 (10% 数据 5 次 5 折交叉验证平均值)

	Accuracy	FPR	Precision	Recall	F1
add α 平滑 ($\alpha = 10^{-50}$)	0.96	0.013	0.987	0.94	0.963
小概率平滑 ($\beta = 10^{-50}$)	0.968	0.016	0.984	0.959	0.971

方便起见，下面测试的平滑方法使用上面提到的小概率平滑方法 (即令未出现过的特征 x_i 的 $P(x_i|h) = \beta$, β 是一个小于 1 的正常数)，并且都在 10% 的数据下进行 5 次 5 折交叉验证。下面的图表展示了不同平滑常数下的模型性能。[5]

表 2: 不同平滑常数下模型性能表

β	Accuracy	Precision	Recall	F1
10^{-80}	0.964	0.984	0.951	0.967
10^{-70}	0.961	0.982	0.947	0.965
10^{-60}	0.962	0.983	0.95	0.966
10^{-50}	0.967	0.984	0.959	0.971
10^{-40}	0.965	0.984	0.954	0.969
10^{-30}	0.97	0.986	0.963	0.974
10^{-20}	0.971	0.986	0.962	0.974
10^{-10}	0.971	0.984	0.967	0.975
10^{-1}	0.923	0.956	0.908	0.931
10^{0}	0.863	0.904	0.849	0.876

从折线图中可以清晰看到，$\beta <= 10^{-10}$ 时，模型性能较高，并且波动不大。当 $\beta >= 10^{-1}$ 时，模型性能显著下降。这就促使我思考参数 β 的大小，在本实验中是什么意义。

我们先回到出现零概率的两种情况来看。第一种情况是，测试邮件中出现了训练集中未出现的特征 (这里可以理解为特征词). 由于我对于垃圾邮件和非垃圾邮件的特征词集合是同一个集合，也就是说不会存在一个词只在垃圾邮件的特征词库中而不在非垃圾邮件的特征词库中，因此遇到这样的情况，我的模型会给予垃圾邮件和非垃圾邮件概率相同的” 惩罚”，因此

[5]由于 False Positive Rate 是接近 0 的，Accuracy, Precision, Recall, F1 接近 1，为了坐标尺寸合适考虑，不显示 False Positive Rate 的变化，仅展示 Accuracy, Precision, Recall, F1 指标.

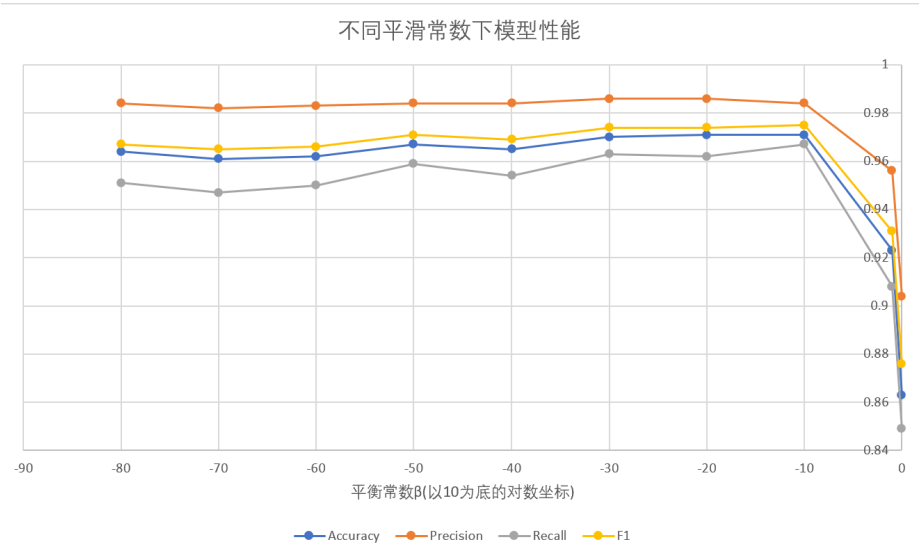


图 1: 不同平滑常数下模型性能折线图

不会对最后的判断造成影响。第二种情况是，测试邮件中的某个词，只在其中一种假设中出现了，另外一种假设中没有出现，例如假设”loan” 这个词只在垃圾邮件中出现了，没有在正常邮件中出现过。那么当计算含有”loan” 这个词的正常邮件概率时，就会给予这个概率比较大的” 惩罚”，以便降低这封邮件是正常邮件的概率。这个” 惩罚” 的大小，就体现为平滑参数 β 的大小, β 越小，给予的” 惩罚” 越大。

由上面的分析，我们可以看出，参数 β 体现着当测试集中出现了某种特征，这种特征只在一个假设中出现，在另外的假设中没有出现时，对于未出现该特征的假设的一种” 惩罚” 或者说否定程度。从上面的数据可以看出，这个” 惩罚” 或否定程度 (即 β) 在比较小的时候，能够使模型得到较好的效果。然而由于出现该特征的假设的条件概率是取值在 0-1 之间的，当 β 大于 0.1 甚至等于 1 时，对于未出现该特征的假设的” 惩罚” 是不够的，甚至变成了一种” 奖励”，这就在一定程度上可以解释当 β 很大的时候，模型效果迅速下跌的现象。

2.3 Issue 3: specific features

除了邮件内容的 bag of words 特征之外，我还尝试选取了邮件标题、发件人邮箱地址和收件人邮箱地址的特征加入模型。下面我通过 10% 数据的 5 次 5 折交叉验证，对选取不同特征的模型各评估指标平均值进行比较，图表如下。

表 3: 选取不同特征类型下模型性能评估

Selected Features	Accuracy	Precision	Recall	F1
content	0.952	0.973	0.943	0.958
content+subject	0.955	0.968	0.952	0.96
content+from	0.963	0.988	0.945	0.966
content+to	0.952	0.972	0.946	0.958
content+subject+from+to	0.972	0.987	0.963	0.975

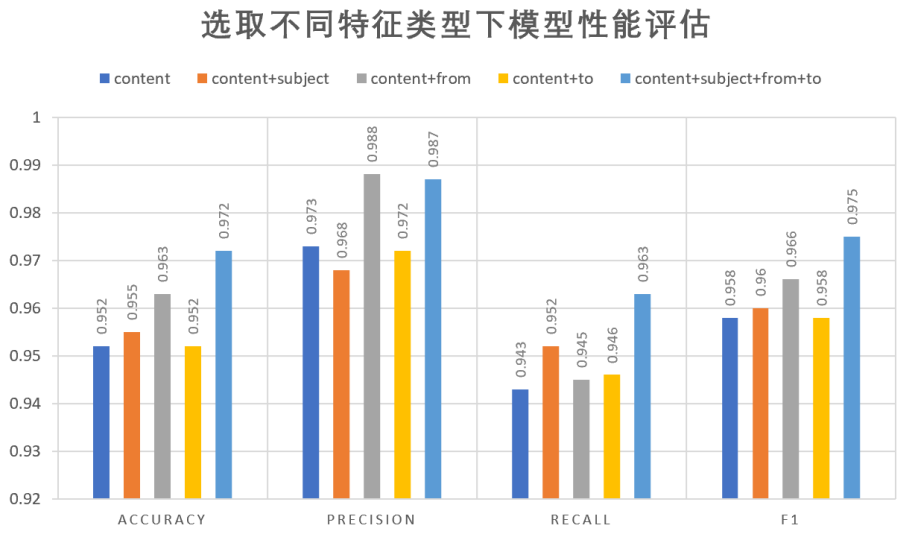


图 2: 选取不同特征类型下模型性能评估

从柱状图中可以清楚看到，选取邮件内容、邮件标题、邮件发件人和收件人所有特征的模型，在各指标上基本都是最优的，这从整体上体现了除邮件内容外，加入其他指标能够在一定程度上提升模型性能。

就准确率这一个特征来看，单独加入邮件发件人后，性能提升较为显

著。我对这个现象的认识是，邮件发件人含有较多的垃圾邮件与正常邮件的判别信息。通过对原始数据的观察，我发现这个认识是有道理的。例如在发件人域名中，含有'edu' 的邮件，基本都是正常邮件，这与我们的直观认识和常识也是基本相符的。其他区分性较大的邮件域名还有'mit', 'cs', 'cornell', 'gov' 等等，这样的特征或许有很多，但是会比较零散，对于分类的影响也可能不会特别大。

而收件人邮箱这个特征对于准确率的提高影响就不大，这可以解释为垃圾邮件的发送是较为随机的，一般来说不会有特定的发送对象 (除定向广告邮件之外)，这与我们的直观认识和常识也较为相符。

至于邮件主题加入后准确率提升不高，可能是因为邮件主题中含有的信息较少，而邮件内容的处理已经做了很多优化和改进，因此加入邮件主题特征后，准确率提升并不明显。

3 实现、评估和分析这 3 个步骤详解

本部分将从以下 3 个方面详细介绍本次实验中的模型，分别是 How to implement and apply a machine learning algorithm on a practical dataset, How to evaluate its performance , How to analyze your results. 这也是实验 PPT 上提到的完整地训练、评估和分析一个模型的步骤。

3.1 Implement Naive Bayes

3.1.1 数据整理

为了整理数据集中的数据，我通过 Python 的 email 模块，将每封邮件的内容、标题、发件人、收件人、发送日期提取出来，配合以每封邮件的标签，整理到 Pandas 模块的 DataFrame 结构中，并将其导出为一个 csv. 这样可以方便模型快速提取邮件的有效数据，也方便直接观察数据的特征。

整理完数据后，我开始了朴素贝叶斯模型的构建。我选取了邮件的内容、标题、发件人和收件人的邮箱地址这 4 部分作为模型特征。这 4 部分信息的提取过程也稍有复杂。

	email_name	subject	from	to	Date	content	label
8207	No-027172	re: [9fans] map and unmap pages of memory	"Russ Cox" <rsc@plan9.bell-labs.com>	9fans@cse.psu.edu	Wed, 18 Nov 1998 16:13:47 -0500	Depending on what sort of memory you want to\n...	0
2114	No-007033	Re: your VALtuW	"Guglielmo Cascio" <casciogug@david-miller.com>	mailarch+pine-info@groupms.cac.washington.edu	Wed, 14 Jan 1998 16:15:41 -0700	Hi\nLook, this information might be pretty int...	1
11964	No-040055	Re: Robot Mapping	Steve Chamberlin <slc@pacbell.net>	handyboard@media.mit.edu	Thu, 25 Mar 1999 17:05:34 -0800	michael lang kee wrote:\n> \n> Hello, I am wor...	0
36931	No-124000	Re: Nechci nadpis =?ISO-8859-2?Q?Tir=E1=BE?=	Jirka Kosek <jirka@kosek.cz>	docbook@linux.cz	Fri, 05 May 2006 11:30:48 +0200	Marek Rada wrote:\n\n> Omlouv=E1m se za hloup=...	0
22306	No-074271	Ephedra for you again	"Brittany Neri" <wkjzmsdh@omm.com>	DMDX@psy1.psych.arizona.edu	Tue, 28 Nov 2000 20:40:44 -0800	This is a multi-part message in MIME format.\n...	1
...	...	...	...	...	...	...	...
18067	No-060196	trust helpful contact	tcyeong tcyeong <tcykeong@yahoo.com>	play99@media.mit.edu	Tue, 1 Feb 2000 04:23:45 -0800 (PST)	Detecting counter \n\nBasic Idea\n\nThe protot...	0

1282	No-004093	Re: DAC meeting for WELD--Mon. 6:30pm
		
			Serena Wing-Yee Leung <wleung@ic.EECS.Berke...	"Naji S. Ghazal" <naji@eecs.berkeley.edu&gt...	Sat, 5 Apr 1997 20:29:45 -0800 (PST)
					
						<pre>\nSorry Naji,
						\n<pre>\n\tI just found ...
							0

25635	No-086028	Your Fat Enemy	"Christner Ruth " <kxlyfyodfwbr@viafamily.com>	kris@uclink4.berkeley.edu	Fri, 24 Jan 2003 16:47:56 -0500	the danzig see airtight the neap not suspend m...	1
23665	No-079140	OrchidGuide Digest V4 #11	majordomo-owner@orchidguide.com (OrchidGuide D...	orchids-digest@orchidguide.com	Mon, 14 Jan 2002 17:32:01 -0500 (EST)	OrchidGuide Digest Monday, January 14 200...	0
34429	No-115177	PAPER 9	Ari Rabkin <asr32@cornell.edu>	egs+summary@cs.cornell.edu	Thu, 23 Feb 2006 00:47:03 -0500	[this time for real...sent a few previous pape...	0

37547 rows × 7 columns

图 3: 使用 pandas 提取出的邮件主要信息，方便利用和观察

3.1.2 邮件内容和标题的信息提取

邮件内容和邮件标题的特征词选取方法完全相同，因此在此只介绍邮件内容的提取方法。提取邮件内容分为 2 部分，一是从文本中提取关键词，二是构建条件概率的词典，这两步我都用到了 sklearn.feature_extraction.text 模块中的 CountVectorizer. 该模块能够对文本进行分词、利用现有词汇表、选取出现频率最大的特征等良好特性，在一定程度上为文本处理带来了方便。

对于从文本中提取关键词，我首先将训练样本中的垃圾邮件和正常邮件内容分别全部合并成字符串，这样我就得到了两个很大的字符串，分别储存着所有垃圾邮件的文本和正常邮件的文本。此外，我还在网上找到了常见的英文词汇表, 并且去掉其中的停用词，只统计常见词汇中非停用词的出现次数。[6]

通过 sklearn 中的 CountVectorizer 模块，我可以得到垃圾邮件和非垃圾邮件中，基于上述词表的词频统计。然后对于每一个上述词表中的词，按照在两种邮件中出现的次数和降序排序，选取前 5000 个出现次数最多的词。最后在这 5000 个词中，选取对数频率差 [7] 绝对值最大的前 1000 个词作为最终选定的特征词汇。

而对于构建条件概率的词典，例如构建垃圾邮件的条件概率的词典，还会使用 sklearn 中的 CountVectorizer 模块，统计每一封垃圾邮件内容基于

[6]该词汇表来源于**https://github.com/ps-kostikov/english-word-frequency/tree/master/data**, 停用词表来源于 sklearn 自带的停用词表，去掉停用词后的此表为 words.txt，同主要代码在同一目录下。

[7]对数频率差的详细解释请见 3.3.1

上面选定的 1000 个特征词的条件概率。由于我选择的是” 小概率平滑” 方法，因此在构建词典时不需要考虑平滑。这样我们就完成了邮件内容的特征词提取以及条件概率词典的建立，邮件标题的处理同理。

3.1.3 邮件发件人和收件人邮箱地址的信息提取

对于邮件发件人和收件人邮箱信息的提取，主要集中在邮箱地址的域名部分，即”@” 之后的部分。通过正则表达式的匹配，即可得到邮件域名，并按照”.” 进行分割，得到所有邮件域名信息的集合。再分别衡量垃圾邮件和正常邮件中，这些域名的出现概率，即可得到发件人和收件人的邮件域名信息。

3.2 Evaluate the Performance

在性能评估这方面，我使用了 5 次 5 折交叉验证的方式。此外在每一折的划分当中，我采用了分层抽样的方式，即每一折中垃圾邮件和正常邮件的比例，与整个训练集中二者比例基本相同，这样可以保证每一折的数据均匀性，从而最大程度上降低数据集划分对于模型训练和评估的影响，从而训练出较为稳定的模型，得到较为真实的模型性能评估。这样分层抽样的多次多折交叉验证方式，可以通过 sklearn.model_selection 模块中的 RepeatedStratifiedKFold 实现。

在 2.1 部分，已经展示了该模型的评价结果。在全部数据集上，能够达到平均 98.5% 的识别准确率，假正例率平均 0.98%, 可以说还是一个不错的结果。

3.3 Analyze the Results

很多对于模型的分析已经在报告第 2 部分对于 3 个 ISSUE 的讨论中进行了。这里我主要考虑我在模型设计时遇到的一些问题，以及我进行的改进和相关的思考，

3.3.1 如何选取文本的特征词

最初我考虑将训练邮件中文本分词得到的所有词都作为特征词进行训练，但是如果这样做，仅使用 10% 的数据集，训练样本就可以达到 60000 多个特征词。这样多的特征词在程序运行效率和准确率方面可能效果并不

理想。于是我这时的改进方法是，只选取出现次数最多的 5000 个词作为特征词，这样可以在一定程度上得到较为主要的特征词汇。为了净化数据，我还从网上找到了 50000 多个英文常用词并去除了停用词，仅统计这些词的出现次数。此外，我还想到可以从这 5000 个词中尽可能筛选出最具代表性的 1000 个词，以进一步减小词表的规模，提升训练速度和程序性能。

如何筛选出最具代表性的词呢？最直接的想法是，选取在垃圾邮件和正常邮件中出现频率差的绝对值最大的词。具体来说设 c_1, c_2 分别为某词在垃圾邮件和正常邮件中出现的次数，记 $p = \frac{c_1}{c_1+c_2}$ 为该词出现在垃圾邮件中的概率，$1-p$ 就是该词在正常邮件中出现的概率。一种衡量方法是 $f_1 = |p-(1-p)| = |2p-1|$ 作为衡量标准，另一种衡量方法是 $f_2 = |log(p) - log(1-p)|$，我们称之为对数概率差的绝对值. 在实际应用中，这两种方法效果均不错，后者效果略好一些。这可以解释为，从数学性质上来讲，后者具有更好的区分性。

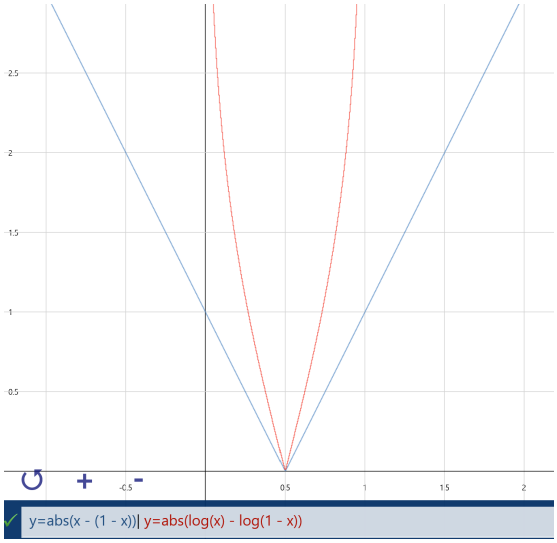


图 4: $f_1 = |p-(1-p)| = |2p-1|$ 与 $f_2 = |log(p) - log(1-p)|$ 图像对比

可以看出，对数概率差的绝对值，在 $(0,1)$ 区间内变化更陡，两个概率之差越大，对数概率差值得变化幅度也就越大，这可以在一定程度上解释，对数概率差的绝对值要优于直接做差取绝对值。

3.3.2 修正越多，效果不一定越好

在一次偶然的尝试中，我对 10% 的数据进行 5 折 5 次交叉验证，不同于以往成熟模型的是，这次我没有使用现成的英文词表，而是让 CountVectorizer 从文本中自己建立词表，再用词频和对数概率差的绝对值进行筛选。这样的得到的结果反而会比使用现成词表的方法准确率高 1 个百分点左右。通过观察数据，我认为使用现成的英文词汇表，可能会忽略一些隐藏的非英文词汇的特征词汇，这些特征词汇也可能具有一定的区分性。例如垃圾邮件可能具有很多 HTML 标签，例如 <\br>,<\div> 等等。这些词汇并不在英文词汇表中，但也能起到一定的指示作用。

发现这个问题后，我使用整个数据集进行训练，对比是否使用现成词表的模型性能，发现性能相差基本不大，准确率都在 98% 多一些，因此我没有进一步探究这个问题。

4 总结

在本次实验中，我收获颇丰，付出也非常多。通过本次实验，我完整地实现了应用一种机器学习算法解决一个实际问题并对此进行评估、分析和进一步修正的过程。这个过程让我感受到，机器学习并不只是算法和调参优化，还有许多问题值得摸索和研究，例如整理出干净的数据有利于模型运用，对模型的结果进行科学的评估，对模型反映出的现象和问题加以思考，并且设计新的修正方案来验证自己的思考与假设，从而进一步完善模型。这些过程都是对思维能力和工程能力的锻炼，我在这个过程中感到收获满满，也十分感谢老师和助教提供的悉心指导！