上海大学

SHANGHAI UNIVERSITY

课程设计（论文）

UNDERGRADUATE COURSE (THESIS)

题目:基于机器学习的亚马逊食品评论情感分类

学院	计算机工程与科学学院
专业	计算机科学
学号	19121442
学生姓名	曹卓文
指导教师	武星
起讫日期	2021.03.25 – 2021.06.01

目录

基于机器学习的亚马逊食品评论情感分类

摘要

随着电子商务的快速发展以及移动上网设备的普及，人们越来越热衷于在网上进行购物，并对购物体验做出评价。这些评价反映了商品的质量，影响和引导潜在客户的购买行为。这些评价数据的提取和分析可以更好地帮助商家了解自己的产品在市场上的反馈，同时能够帮助电商平台更好地识别客户情绪，提升服务质量。

本文从文本评论切入，以 Kaggle 平台上亚马逊食品评论数据集为数据源，采用机器学习的方法对用户情感进行分类，并对不同种机器学习方法进行比较，并找出最优方法。

本文采用了词向量特征表示模型，通过多种机器学习模型对不同词向量特征表示模型进行训练和预测，最后将预测结果进行评比，探索在正负样本数量不均衡的数据集建立最优的分类模型。

关键词：情感分类，意见挖掘，机器学习，自然语言处理

Sentiment classification of Amazon food review using machine learning

ABSTRACT

With the rapid development of e-commerce and the popularity of mobile Internet devices, people are increasingly eager to shop online and rate their shopping experience. These reviews reflect the quality of the products and influence and guide the purchasing behavior of potential customers. The extraction and analysis of these evaluation data can help merchants understand the feedback of their products in the market, and can also help e-commerce platforms identify customer sentiment better in order to improve service quality.

In this paper, we use text reviews as research subjects and use the Amazon food review dataset on Kaggle platform as the data source to classify user sentiment by machine learning methods.

In this paper, word vector feature representation models are used, and different word vector feature representation models are trained and predicted by multiple machine learning models, and finally the prediction results are evaluated to explore the optimal classification models in data sets with unbalanced number of positive and negative samples.

Keywords: sentiment classification, opinion mining, machine learning, natural language process

第 1 章 绪论

本章主要描述了文本情感分类的背景、意义，分析了相关课题国内外的研究现状，进而提出了本文所要研究的内容及目标。

§1.1 文本情感分类的背景及意义

文本情感分类也可以被称为情感分析(Sentiment analysis)或意见挖掘(Opinion mining)是指对带有情感色彩的主观性文本进行分析、处理、归纳和推理的过程，是自然语言处理(Natural Language Process, NLP)领域的重要课题之一。

文本情感分类无论是对用户还是商户，乃至平台都有着重要意义。Chintagunta等研究发现用户电影评论对电影票房产生影响。[1]王伟等基于评论数据研究发现评价对销量存在正相关。[2]小到通过评价数据评判单个商品质量好坏，大到平台对其整体商品评价的分析，都有意见挖掘的一席之地。而伴随着大数据和机器学习领域的浪潮，情感分析将会在各个领域起到越来越重要的作用。

而目前电商网站的评论伴随着电商的蓬勃发展积累了海量的数据，而这些数据如何从中挖掘有价值的信息成为了重要课题。

§1.2 文本情感分类研究现状及存在的问题

国内外学者对文本情感分析做了大量的研究，接下来笔者将从研究发展起步，谈论目前的研究现状并讨论当前文本情感分类研究的难点。

§1.2.1 研究发展

自 1997 年 MIT 媒体实验室 Picard 教授提出情感计算，并在 2000 年出版著作 *Affective computing*[3]以来，情感计算已经诞生了 24 年。文本情感分类按照处理文本的粒度不同，可分为词语短语级、句子级、篇章级等几个研究层次[4]。

最早开始的是基于字典的情感分类方法。1997 年，Riloff 和 Shepherd[5]在文本数据的基础上进行了构建语义词典的相关研究。同年，McKeown[6]发现连词对大规模的文本数据集中形容词的语义表达的制约作用，通过聚类算法对英文的形容词与连词做情感倾向研究。2003 年，Turney 和 Littman[7]提出利用点互信息的方法扩展了正负面情感词典。2004 年，Sista 等[8]将 General Inquirer 和 WordNet 中的褒义和贬义词作为种子词，得到一个扩展后的较大规模情感词集合，并以此作为分类特征，利用机器学习方法对文本褒贬义进行了自动分类。同年，Faye Baron 和 Graeme Hirst[9]从文档中抽取倾向性强的搭配作为种子词汇，取得了较好的分类效果。

§1.2.2 研究现状

近年来，主要采用词向量方法将字词映射到向量空间并采用机器学习的方法实行分类任务。Pang 等运用朴素贝叶斯网络、支持向量机和最大熵模型这三种方法对影评进行分类。[10]张景阳等使用多元线性回归模型及 BP 神经网络模型对农村居民纯收人进行对比预测研究，结果表明 BP 神经网络具有较好的效果。[11]

§1.2.3 文本情感分析的难点

虽然文本情感分析技术已经发展许久，但这一研究领域仍然存在以下方面的技术难题：

（1）情感语义的机器理解问题。人类的自然语言情感表达十分复杂，特别是网络评论的形式更加灵活多变。要使机器精确的理解文本中的情感内容，不能简单的提取词语作为特征，还必须结合语言学方面的知识，借助于文本上下文和领域相关性对情感语义进行分析处理。

（2）特征提取的问题。由于情感表达中有许多使用诸如反语和隐喻等修辞手法的表达，并且上下文相关。例如“我喜欢吃苹果”和“我用苹果电脑”根据文章的主题和上下文不同，两个“苹果”具有不同的意思。

（3）数据集正负样本不均衡问题。人往往对产品的正面评价多于负面评价，因此往往训练数据会得到数倍于负样本的正样本数量，这对机器学习的训练造成了明显的困扰。

§1.3 本文研究内容及目标

在用户对产品进行评分时，根据现实经验，人们往往倾向于做出正面评价。因此本文尝试解决第三类难题，以正负样本不平衡的亚马逊用户评论数据作为数据集，尝试不同的机器学习算法，对比指标以期寻找出在这类数据集上最优的算法。

§1.3.1 研究内容

本文研究具体研究内容有以下几个部分组成：

（1）数据清洗；

（2）代入不同模型；

（3）分析训练结果；

（4）选取最优结果；

§1.3.2 研究目标

选取多种模型评价指标，将多种机器学习模型的测试结果进行横向对比，在

正负样本不均衡的数据集上建立最优模型。

§1.4 本文组织结构

整篇论文分为五章。

第一章介绍了研究背景、研究意义，分析了文本情感分类的发展和现状以及存在的问题和难点，并提出了本文的研究内容以及研究目标。

第二章主要介绍本文所采用的词向量化方法，机器学习方法和模型评价指标。

第三章主要描述了数据清洗和提取关键词过程。

第四章主要描述了各种模型的训练结果和比较结论。

第五章对全文进行了总结，归纳了本文的主要工作与创新点，并指出了需要进一步研究的问题。

第 2 章 机器学习方法与模型评价指标

本章主要介绍本文所采用的词向量化方法，机器学习方法和模型评价指标。

§2.1 词向量

第一种表示方式是热独(one-hot)编码，就是用单维度的向量表示一个词，向量的长度为词典的大小。但这种词表示有两个缺点：

（1）容易受维数灾难的困扰，尤其是用于深度学习时；

（2）不能很好地刻画词与词之间的相似性。

第二种是分布式表示(Distributed Representation)，它最早由 Hinton[12]于 1984 年提出的，以克服热独编码的缺点。其基本思想是通过训练将文本中的每一个词映射成一个固定长度的向量，然后构成一个新的词向量空间，每一个向量对应该空间中的一个点，在这个空间上引入距离，根据词之间的距离来判断它们之间的相似性。

本节介绍了三种字词分布式表示的方法。

§2.1.1 BoW

在信息检索中，词袋模型(Bag of words model)假定对于一个文档，忽略它的单词顺序和语法、句法等要素，将其仅仅看作是多少个词汇的集合，文档中每个单词的出现都是独立的而不依赖于其它单词是否出现。也就是说，文档中任何一个位置出现的任何单词，都不受该文档语意影响而独立挑选的。

但是词袋模型严重缺乏相似词之间的表达。例如“苹果好吃”和“苹果很甜”，“苹果味道好”，这几句话意思都是差不多的，但在词袋模型看来，它们和“苹果好吃”的相似性不如“苹果不好吃”，而“苹果不好吃”表达的是完全相反的意思。

在本文中，使用函数 CountVectorizer(stop_words = 'english').fit_transform(X) 来实现向量化。

§2.1.2 TF-IDF

TF-IDF(Term Frequency-Inverse Document Frequency, 词频-逆文件频率)是一种用于资讯检索与资讯探勘的常用加权技术。TF-IDF 是一种统计方法，评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数增加，但同时会随着它在语料库当中出现的频率减少。

TF（Term Frequency,词频）表示词条在文本中出现的频率，这个数字通常会被归一化（一般是词频除以文章总词数），防止偏向长文件。其中 TF 的计算公

式为：

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

IDF(Inverse Document Frequency, 逆文件频率)表示关键词的普遍程度。如果包含词条 i 的文档越少，IDF 越大，则说明该词条具有很好的类别区分能力。某一特定词语的 IDF，可以由总文件数目除以包含该词语之文件的数目，再将得到的商取对数得到:

$$IDF_i = \log \frac{|D|}{1 + |j : t_i \in d_j|}$$

其中|D|代表所有文档数量，$|j:t_i \in d_j|$表示包含词条 t_i 的文档数量。分母加一防止除以 0 发生。

某一特定文件内的高词语频率，以及该词语在整个文件集合中的低文件频率，可以产生出高权重的 TF-IDF。因此，TF-IDF 倾向于过滤掉常见的词语，保留重要的词语，表达为：

$$TF\text{-}IDF = TF \cdot IDF$$

本文中，使用函数 TfidfVectorizer(stop_words = 'english').fit_transform(X)实现向量化。

§2.1.3 Embedding

在深度学习的应用过程中，Embedding 层将离散变量转变为连续向量，将文本中的字词从原先所属的空间映射到新的多维空间，为神经网络在各方面的应用带来了极大的扩展。

本文中通过 model.add(Embedding())为网络直接添加 embedding 层。

§2.2 机器学习模型

在本文中，机器学习被分为两类，第一类是以逻辑回归(Logistic Regression)为代表的经典机器学习,第二类是以长短期记忆人工神经网络(Long Short-Term Memory, LSTM)为代表的深度学习模型

§2.2.1 逻辑回归(Logistic Regression)

逻辑回归（Logistic Regression）是一种用于解决二分类问题的机器学习方法，

其假设数据服从 Logistic 分布，然后使用极大似然估计对参数做估计。Logistic 分布是一个连续型的概率分布，其分布函数 F(x)和概率密度函数 f(x)为：

$$F(x) = P(X \le x) = \frac{1}{1 + e^{-(x-\mu)/\gamma}}$$

$$f(x) = F'(X \le x) = \frac{e^{-(x-\mu)/\gamma}}{\gamma(1 + e^{-(x-\mu)/\gamma})^2}$$

Logistic 分布是由其位置 μ 和形状参数 γ 定义的连续分布,其分布的形状与正态分布形状类似，但是尾部更长，所以笔者使用 Logistic 分布来建模比正态分布具有更长尾部和更高波峰的数据分布。

使用逻辑回归的训练速度较快，计算量和特征数目正相关并且可以直接从特征权重看出对结果的影响，本文的分类问题属于二分类问题，逻辑回归较为适合该类问题。

在本文中，笔者直接使用 sklearn.linear_model 中的 LogisticRegression 函数。

§2.2.2 长短期记忆人工神经网络(Long Short-Term Memory, LSTM)

LSTM 是一种时间递归神经网络[13]，它出现的原因是为了解决循环神经网络(Recurrent Neural Network, RNN)的一个致命的缺陷。RNN 会遇到一个严重的问题，称为 RNN 的梯度消失（The vanishing gradient problem for RNNs），这使得 RNN 无法在深度学习上得到应用。之后研究人员开始使用递归神经网络来对时间关系进行建模，LSTM 网络已被证明比传统的 RNNs 更加有效。

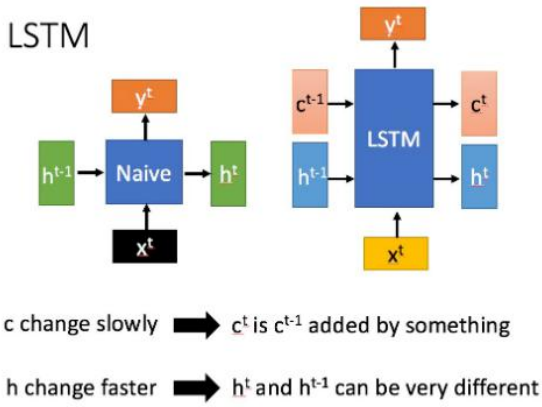

图 1 RNN 和 LSTM 的神经节点

如图 1 所示，左边为 RNN 的神经节点，右图为 LSTM 的神经节点。LSTM 和 RNN 最大的区别体现在三个门：

1）遗忘门：这个阶段主要是对上一个节点传进来的输入进行选择性忘记。

2）记忆门：这个阶段将这个阶段的输入进行选择性记忆。

3）输出门：这个阶段将决定哪些将会被当成当前状态的输出。

以上三个阶段帮助 LSTM 网络解决了梯度消失问题，LSTM 得以训练数百层的网络进行深度学习。

LSTM 的门机制使得网络可以捕捉到时序上的情感关系，因此在情感分类问题上非常有效。

本文中使用 keras 的自带函数构建网络模型，代码见附页。

§2.3 模型评价标准

在机器学习中，笔者需要一定的指标来评测算法的优劣，以下介绍本文中使用的评价指标。

§2.3.1 Accuracy

准确率(Accuracy)代表预测正确样本和被预测总样本数的比值。但是该评价标准对分布不均匀样本不能做出准确评估。

§2.3.2 F1-Score

F1 分数（F1-score）是分类问题的一个衡量指标。多分类问题的机器学习问题中，一般将 F1-score 作为最终测评的方法。它是精确率和召回率的调和平均数，最大为 1，最小为 0。公式为：

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

其中 precision 代表精确率，recall 代表召回率。

精确率是针对预测结果，表示的是预测为正的样本中有多少是真正的正样本。一种就是把正类预测为正类(TP)，另一种就是把负类预测为正类(FP)，也就是：

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

召回率是针对原样本的，它表示的是样本中的正例有多少被正确预测。一种是把原来的正类预测成正类(TP)，另一种就是把原来的正类预测为负类(FN)。公式为：

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

§2.3.3 混淆矩阵

混淆矩阵也称误差矩阵，是表示精度评价的一种标准格式，普遍用于分类问题中。混淆矩阵的列代表了预测类别；每行代表了数据的真实类别。但是在本文

中，正负样本数量不均衡，参考价值不大，固不采用。

§2.3.4 ROC 曲线和 AUC 值

ROC 是一个用于度量分类中的非均衡性的工具，ROC 曲线及 AUC 常被用来评价一个二值分类器的优劣。ROC 曲线在实际的数据集出现类别不平衡时，即负样本比正样本多很多（或相反），且测试数据中的正负样本的分布也可能随着时间而变化。在这种情况下，ROC 曲线能够保持不变。

其中横坐标为 FPR（False positive rate，假阳率），纵坐标为 TPR（True positive rate，真阳率）。公式如下：

$$FPR = \frac{FP}{FP + TN}$$

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

AUC（Area Under Curve） 定义为 ROC 曲线下的面积，所以取值范围在 0.5 和 1 之间，其值越大代表分类器效果更好。

§2.3.5 基线模型

该模型应该是“虚拟”模型。提供一个基准来作为自定义模型的基准，通过和“虚拟”模型对比就可以确保自定义模型性能优于随机结果。

§2.4 本章小结

在本章中，笔者讨论了词向量化的方法，选用的机器学习模型以及模型的评价标准。最终得出采用逻辑回归和 LSTM 作为机器学习模型，以 F1,Accuracy 和 AUC 作为评价标准。[14]

第 3 章 数据清洗与关键词提取

在本章中，讨论了如何对原有数据进行了清洗和关键词再提取，为后续工作做出了重要的铺垫。

§3.1 数据集基本信息

通过 info 函数不难得到数据集的基本列信息。其中包含了包括以下数据：

```
Data columns (total 10 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Id                      525814 non-null  int64
 1   ProductId               525814 non-null  object
 2   UserId                  525814 non-null  object
 3   ProfileName             525798 non-null  object
 4   HelpfulnessNumerator    525814 non-null  int64
 5   HelpfulnessDenominator  525814 non-null  int64
 6   Score                   525814 non-null  int64
 7   Time                    525814 non-null  int64
 8   Summary                 525789 non-null  object
 9   Text                    525814 non-null  object
dtypes: int64(5), object(5)
memory usage: 44.1+ MB
```

图 2 数据集基本信息

其中，Text，Summary 和 Score 是笔者的主要分析对象。本文希望通过 NLP 技术，由 Text 和 Summary 信息推测出评论用户的情感。

通过词云函数，笔者可以简单了解数据集中最常出现的词汇：

图 3 原始数据词云

§3.2 评价分数二值化

在本文中，笔者只希望分辨评论者积极(Positive)或消极情感(Negative)，定义

Score 中大于 3 的为 1，小于 3 的为 0，并且去除 Score 为 3 的中立情感。

§3.3 数据过滤

通过寻找 HelpfulnessNumerator 大于 HelpfulnessDenominator 的数据可以得出，认为有帮助的次数比被评价次数总数还多的数据是异常数据。因此该类数据都应该被清除。

由上面的词云不难看出 br 为原数据集获取过程中混入的 HTML 标签。因此，通过 beautiful soup 库的 get_text 函数获取去除 HTML 标签的文本。

接着进行 NLP 文本分词标准化的常规步骤，将所有的文本都变换为小写。

其次缩写词应该进行扩展展开，如 won't 展开为 will not 等。因此定义缩写展开函数，展开常用缩写。

有一部分用户在评论中使用符号表情如":)"等，因此需要过滤英文以外的文本数据，采用正则表达式，不难剔除除去英文字符外的任何符号。

对于停止词，笔者不希望使用默认的停止词，因为会消除有用信息影响情感表达，例如"I do not like this food"在经过默认停止词过滤后会被变为"like food"，从而直接影响句子的情感表达，因此采用自定义的停止词。

过滤数据使用的代码见附页。

§3.4 清理后数据基本信息

数据清理后，词云如图所示：

图 4 清理后数据的词云

通过数据清理后，统计剩余的积极或消极情感评论数据：

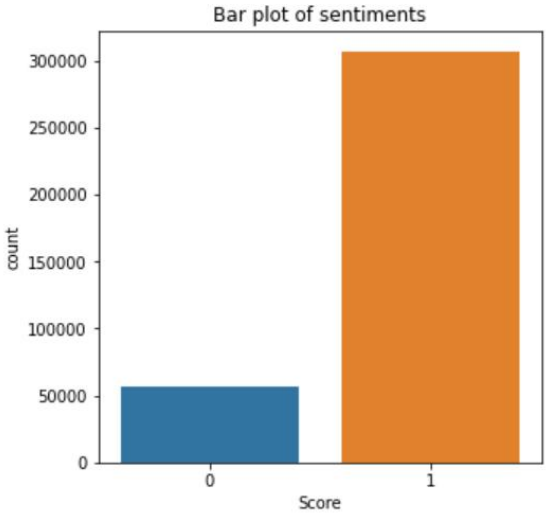

图 5 评论情感统计

可以看出，积极情感的评论远超消极情感评论，正负样本数量极其不均衡，因此使用混淆矩阵作为评价标准效果并不好。因此，本文尝试在样本数量不均衡的数据集寻找表现良好的分类模型。

§3.5 本章小结

在本章中，通过一系列手段，清除了数据集中无关的信息为之后的模型训练做出了重要铺垫，有效提高模型的预测准确率和其他指标。

第 4 章 模型结果与比较结论

本章是本文中最重要的章节，主要展示了各模型的运行数据和结果，并对结果进行了对比和分析，寻找出了最优模型。

§4.1 逻辑回归模型

在本文中，逻辑回归模型的输入数据类型有两种，Text 或 Mixed。Mixed 为 Text 和 Summary 的合并数据。词向量化方法分别有 BoW，TF-IDF 和 n-gram TF-IDF。

§4.1.1 模型权重

使用词云展示各个模型训练后的权重，越大的字体代表权重越高：

图 6 逻辑回归模型权重左图为 Text 数据集右图为 Mixed 数据集

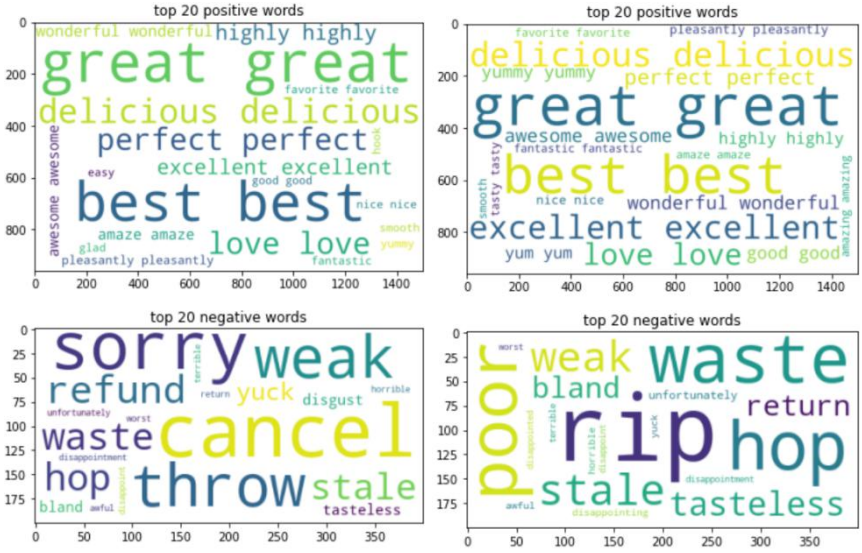

图 7 TF-IDF 逻辑回归模型权重左图为 Text 数据集右图为 Mixed 数据集

图 8 n-gram TF-IDF 逻辑回归模型权重左图为 Text 数据集右图为 Mixed 数据集

从词云可以看出，不同的模型选取的高权重词汇各不相同，但是都符合一般认识中的积极评价或消极评价词汇。逻辑回归模型的工作逻辑清晰明了，通过不同词的权重来计算整个段落的情感。

§4.1.2 F1 score 与 Accuracy

如图 9 所示，为逻辑回归模型在分别采用两种数据和多种词向量化方法的 F1 Score 和 Accuracy 柱状图。如图 10 所示是两者的详细数据。

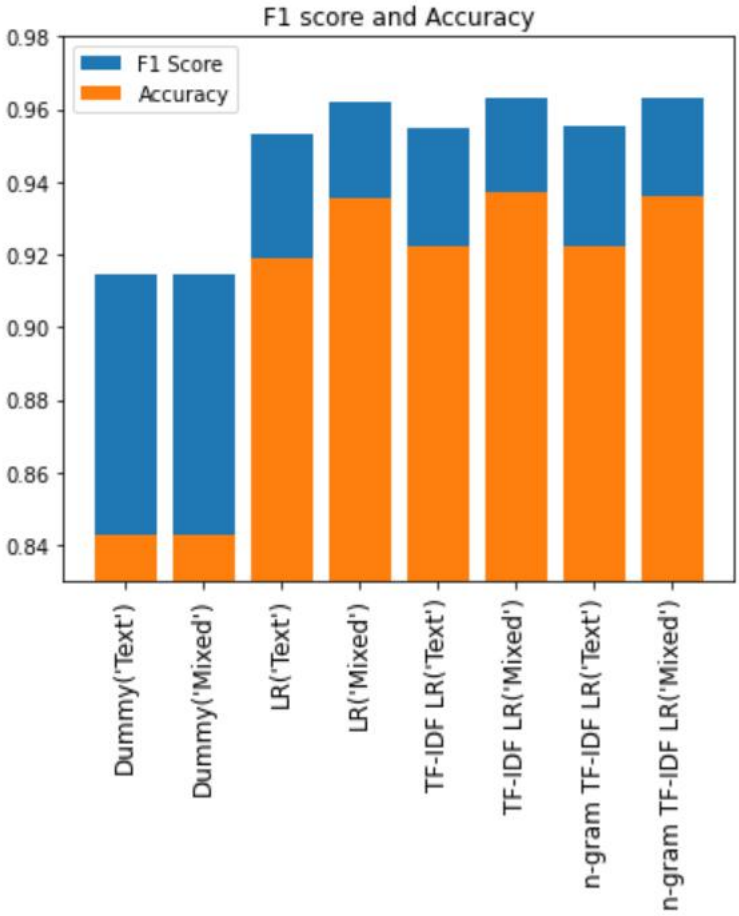

图 9 逻辑回归模型的 F1score 与 Accuracy

```
f1 score list:
0.9147 Dummy('Text')
0.9147 Dummy('Mixed')
0.9529 LR('Text')
0.9623 LR('Mixed')
0.9548 TF-IDF LR('Text')
0.9633 TF-IDF LR('Mixed')
0.9551 n-gram TF-IDF LR('Text')
0.9631 n-gram TF-IDF LR('Mixed')

accuracy list:
0.8429 Dummy('Text')
0.8429 Dummy('Mixed')
0.9192 LR('Text')
0.9357 LR('Mixed')
0.9221 TF-IDF LR('Text')
0.9371 TF-IDF LR('Mixed')
0.9220 n-gram TF-IDF LR('Text')
0.9362 n-gram TF-IDF LR('Mixed')
```

图 10 逻辑回归模型的 F1score 与 Accuracy 的详细数据

从图 7 可以看出，使用 TF-IDF 和 Mixed 作为数据集的逻辑回归模型在 F1

score 和准确率表现优于其他任何模型，F1 score 达到 0.9633，准确率达到 0.9371.

不难从数据中看出，所有采用 Mixed 数据集的模型效果都要优于使用 Text 数据集的模型。

§4.1.3 ROC 曲线与 AUC

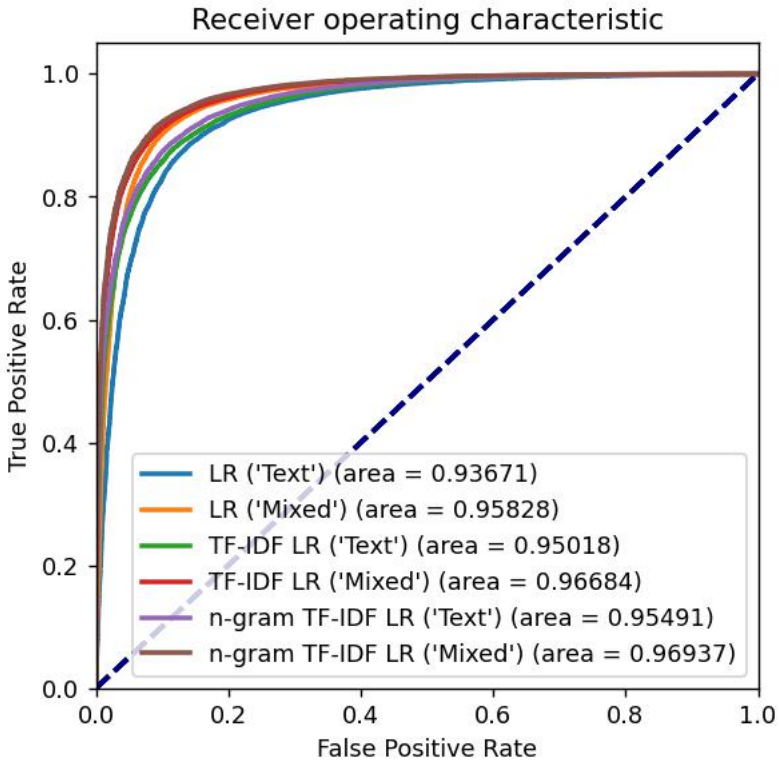

图 11 逻辑回归模型的 ROC 曲线与 AUC

如图 11 所示，得分最高的是 n-gram TF-IDF 和 Mixed 数据集的逻辑回归模型，AUC 为 0.96937，而 F1 和 Accuracy 最高的 TF-IDF 和 Mixed 数据集的逻辑回归模型排名次之。同样的，可以发现，所有 Mixed 数据集都有比 Text 数据集训练的模型拥有更高的 AUC 分数。而各个逻辑回归模型都可以在较低的 FPR 的情况下取得较高的 TPR。

我们不难看出 Mixed 数据集对情感的表达更加明显，比 Text 数据集而言能取得更好的效果。但是是否是因为 Mixed 集合中包含 Summary 数据从而导致预测结果更加精准呢？为了研究单纯采用 Summary 数据集是否能取得更好的结果。因此，额外增加一组使用 Summary 作为数据集的模型作为对照，使用的模型为 AUC 得分最高的 n-gram TF-IDF 模型。

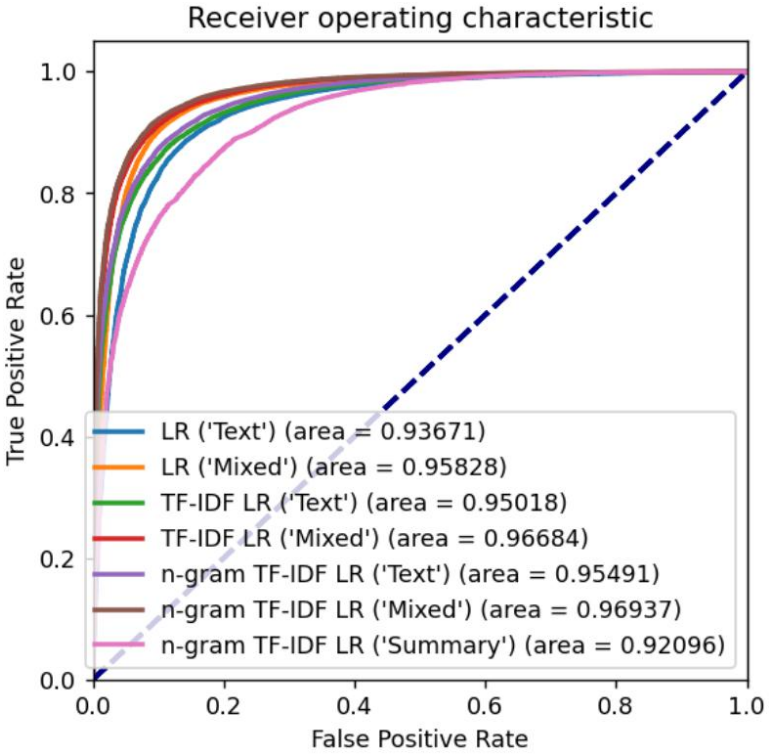

图 12 Summary 数据集的 n-gram 模型 ROC 曲线

```
f1 score list:
0.9147 LR ('Text')
0.9147 LR ('Mixed')
0.9529 TF-IDF LR ('Text')
0.9623 TF-IDF LR ('Mixed')
0.9548 n-gram TF-IDF LR ('Text')
0.9633 n-gram TF-IDF LR ('Mixed')
0.9551 n-gram TF-IDF LR ('Summary')
```

图 13 Summary 数据集得到的 f1 score

可以看出，使用 Summary 的 n-gram TF-IDF 逻辑回归模型模型的 F1 score 低于不同数据集的同类模型，AUC 得分是所有模型中最低的。因此推测 Mixed 模型拥有最好的分类效果。

§4.2 长短期记忆人工神经网络

在本文中，长短期记忆人工神经网络(LSTM)的输入数据类型有两种，Text 或 Mixed。Mixed 为 Text 和 Summary 的合并数据。在本文中网络模型为 Embedding 层+128 个单元的 LSTM 网络+keep_prob 为 0.5 的 Dropout 层和采用 sigmoid 函数作为连接层的全连接网络。详细代码见附页。

§4.2.1 F1 score 与 Accuracy

如图 14 所示，LSTM 模型中，使用 Mixed 数据集的 F1score 和 Accuracy 都取得了最高的得分。根据图 10 的详细数据，F1score 为 0.9683，Accuracy 为 0.9458.

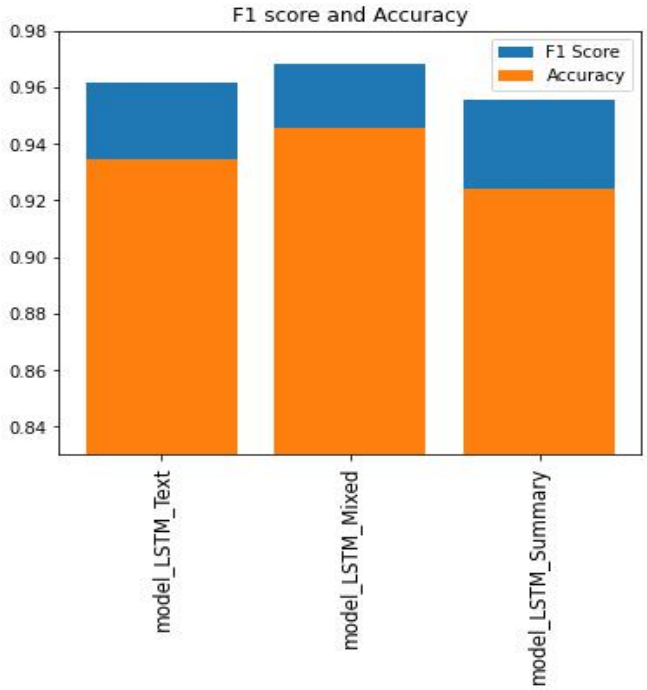

图 14 长短期记忆人工神经网络的 F1score 与 Accuracy

```
f1 score list:
0.9617 model_LSTM_Text
0.9683 model_LSTM_Mixed
0.9555 model_LSTM_Summary

accuracy list:
0.9347 model_LSTM_Text
0.9458 model_LSTM_Mixed
0.9240 model_LSTM_Summary
```

图 15 长短期记忆人工神经网络的 F1score 与 Accuracy 的详细数据

对于逻辑回归模型中取得良好效果的数据集 Mixed，再次作为对照单独使用 Summary 作为训练数据集。但结果显示采用 Mixed 数据集的模型依然拥有最好的分类效果。从图 15 可以看出，分类准确度和 F1 socre 都比同数据集下的逻辑回归模型要更好。

§4.2.2 ROC 曲线与 AUC

如图 16 所示为 LSTM 在不同数据集下 ROC 曲线和 AUC 数据。

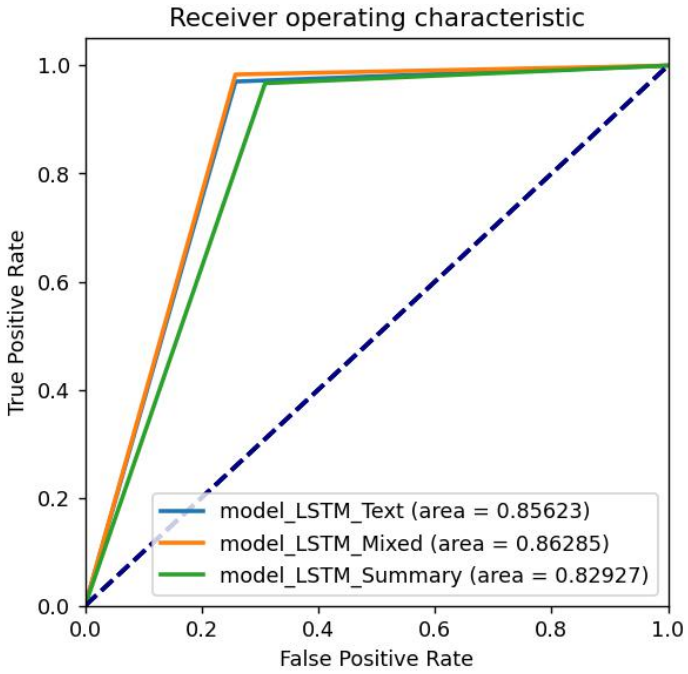

图 16 长短期人工神经网络的 ROC 与 AUC

通过 AUC 数据可知，LSTM 模型的分数都远低于逻辑回归模型，主要反映在 FPR 小于 0.2 后 TPR 数据远不如逻辑回归模型。由于数据集的正负样本严重不平均，因此推测是训练样本的正样本过多导致了过拟合，尝试使用等大的样本进行训练。

§4.2.3 模型优化

通过尝试手工平衡数据集，筛选出正样本和负样本各五万个，并划分数据集最后进行训练，结果如下：

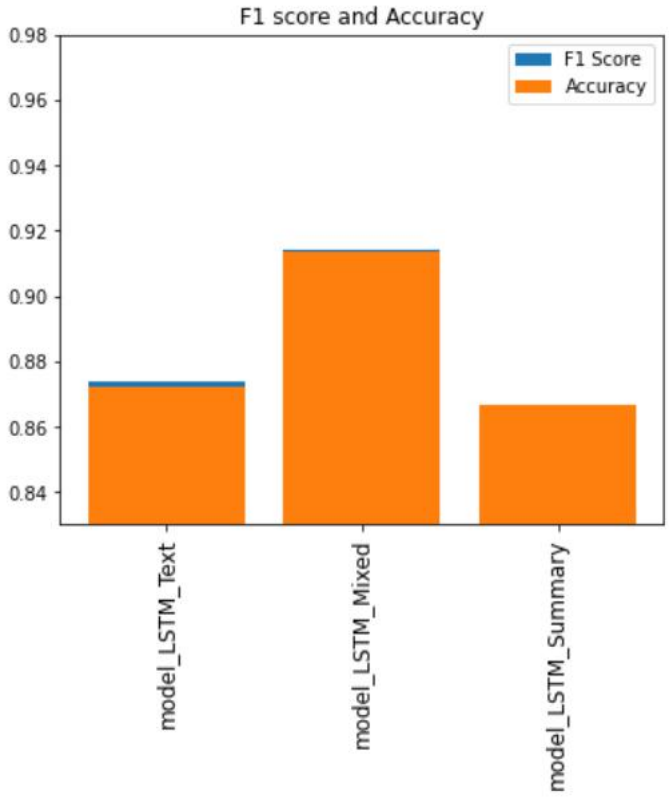

图 17 使用正负样本平衡数据集进行训练的 F1 score 与 Accuracy

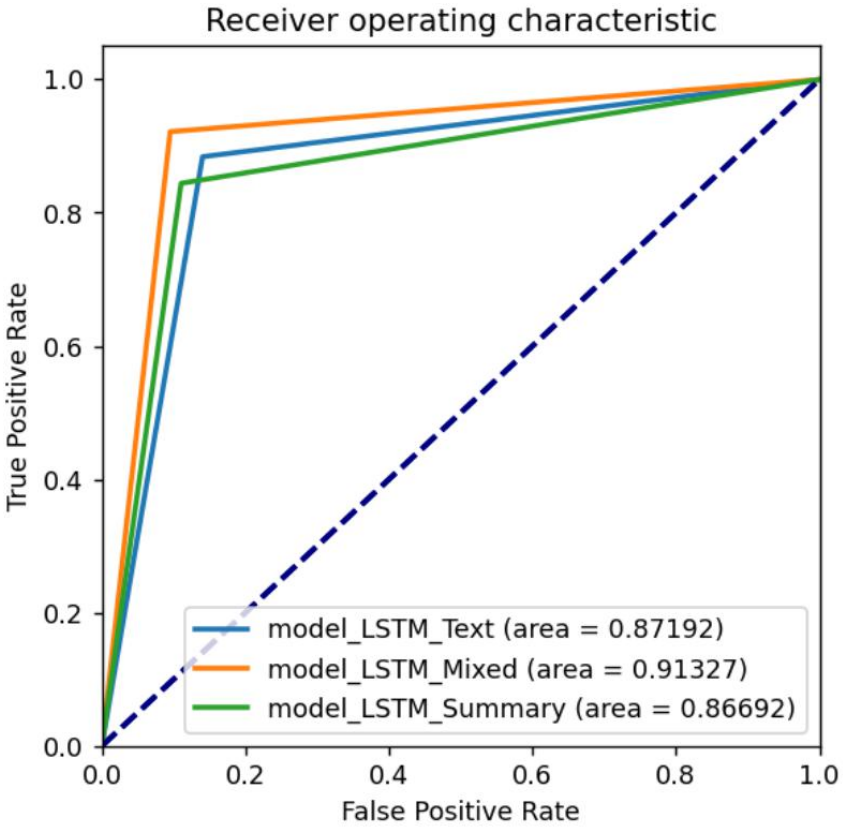

图 18 使用正负样本平衡数据集进行训练的 ROC 曲线与 AUC

可以发现，模型 F1 score 与 Accuracy 得分比原来非平衡数据集训练结果要更加糟糕，AUC 得分在所有数据集上都有所提高，其中 Mixed 数据集模型达到了 0.91327，相较于非平衡数据集提升 5.8%。因此笔者尝试从训练数据上寻找问题，如图所示的是 Mixed 数据集训练的网络训练参数：

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 150, 32)           192032
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               82432
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 274,593
Trainable params: 274,593
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
766/766 [==============================] - 22s 23ms/step - loss: 0.4240 - accuracy: 0.7934 - val_loss: 0.2361 - val_accuracy: 0.9054
Epoch 2/10
766/766 [==============================] - 15s 19ms/step - loss: 0.1933 - accuracy: 0.9281 - val_loss: 0.2243 - val_accuracy: 0.9105
Epoch 3/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1618 - accuracy: 0.9419 - val_loss: 0.2053 - val_accuracy: 0.9203
Epoch 4/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1349 - accuracy: 0.9535 - val_loss: 0.2310 - val_accuracy: 0.9192
Epoch 5/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1144 - accuracy: 0.9598 - val_loss: 0.2189 - val_accuracy: 0.9167
Epoch 6/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1030 - accuracy: 0.9649 - val_loss: 0.2275 - val_accuracy: 0.9209
Epoch 7/10
766/766 [==============================] - 15s 20ms/step - loss: 0.0842 - accuracy: 0.9715 - val_loss: 0.2443 - val_accuracy: 0.9142
Epoch 8/10
766/766 [==============================] - 15s 20ms/step - loss: 0.0745 - accuracy: 0.9760 - val_loss: 0.2725 - val_accuracy: 0.9156
Epoch 9/10
766/766 [==============================] - 15s 19ms/step - loss: 0.0621 - accuracy: 0.9806 - val_loss: 0.2867 - val_accuracy: 0.9134
Epoch 10/10
766/766 [==============================] - 15s 20ms/step - loss: 0.0487 - accuracy: 0.9847 - val_loss: 0.3298 - val_accuracy: 0.9115
Accuracy of the model :  0.9133
F1-score:  0.9143083056040588
```

图 19 采用正负样本平衡的 Mixed 数据集的模型训练结果

其中 train loss 随着 epoch 的增加不断下降，但是可以看到 val_loss 在不断提升网络在第五次 epoch 后出现了过拟合的表现。因此，尝试通过在 LSTM 网络层中加入 L2 正则化以避免过拟合：

```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_12 (Embedding)     (None, 150, 32)           192032
_________________________________________________________________
lstm_7 (LSTM)                (None, 128)               82432
_________________________________________________________________
dropout_7 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 129
=================================================================
Total params: 274,593
Trainable params: 274,593
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
766/766 [==============================] - 22s 24ms/step - loss: 0.6274 - accuracy: 0.7315 - val_loss: 0.2741 - val_accuracy: 0.8999
Epoch 2/10
766/766 [==============================] - 15s 20ms/step - loss: 0.2374 - accuracy: 0.9198 - val_loss: 0.2485 - val_accuracy: 0.9055
Epoch 3/10
766/766 [==============================] - 15s 20ms/step - loss: 0.2029 - accuracy: 0.9321 - val_loss: 0.2372 - val_accuracy: 0.9132
Epoch 4/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1768 - accuracy: 0.9411 - val_loss: 0.2421 - val_accuracy: 0.9138
Epoch 5/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1694 - accuracy: 0.9436 - val_loss: 0.2388 - val_accuracy: 0.9148
Epoch 6/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1537 - accuracy: 0.9491 - val_loss: 0.2465 - val_accuracy: 0.9145
Epoch 7/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1480 - accuracy: 0.9521 - val_loss: 0.2432 - val_accuracy: 0.9170
Epoch 8/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1368 - accuracy: 0.9570 - val_loss: 0.2413 - val_accuracy: 0.9134
Epoch 9/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1299 - accuracy: 0.9604 - val_loss: 0.2514 - val_accuracy: 0.9140
Epoch 10/10
766/766 [==============================] - 15s 20ms/step - loss: 0.1218 - accuracy: 0.9639 - val_loss: 0.2613 - val_accuracy: 0.9147
Accuracy of the model :  0.9137333333333333
F1-score:  0.9128619528619528
```

图 20 加入 L2 正则化后的 LSTM 网络在 Mixed 数据集下的训练结果

可以看到 val_loss 不再明显上升，呈现一定的摆动。但 AUC 得分在 Mixed 数据集的模型上呈现轻微的提升

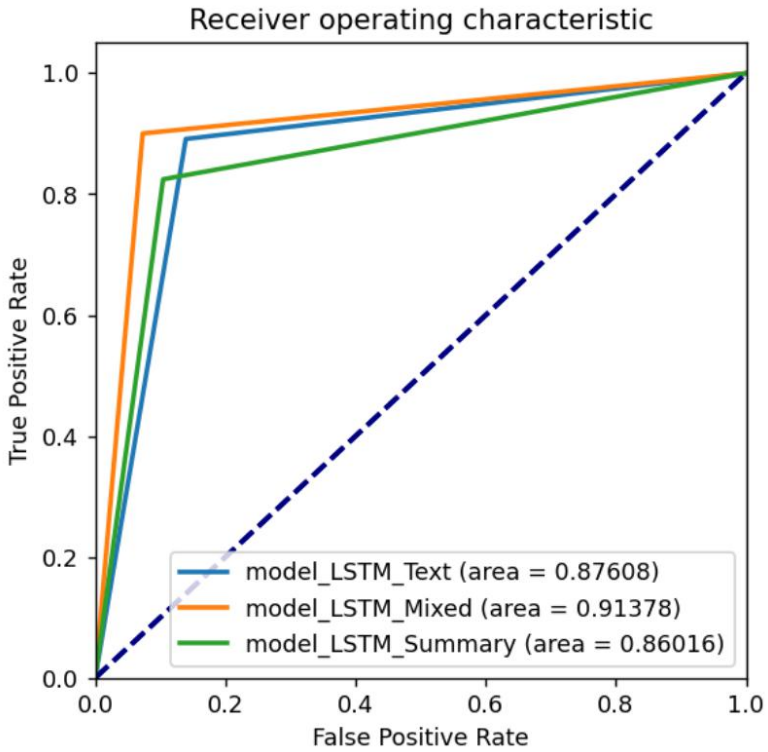

图 21 L2 正则化后的 LSTM 模型的 ROC 曲线和 AUC

笔者推测是由于平衡数据集的数据量较小，达不到训练效果。

§4.3 横向对比

选取三类模型(逻辑回归，LSTM,平衡数据集调优 LSTM)中结果最好的模型进行对比。见表格 1：

模型名称	F1 score	Accuracy	AUC
n-gram TF-IDF 逻辑回归模型 Mixed	0.9633	0.9371	0.96684
LSTM Mixed 数据集	0.9683	0.9458	0.86285
LSTM L2 正则化 Mixed 平衡数据集	0.9129	0.9137	0.91378

表 1 模型指标横向对比

可以看出 n-gram TF-IDF 逻辑回归模型在 Mixed 数据集下有着最好的表现，AUC 达到了最高的 0.96684，正确率和 F1 score 与 LSTM Mixed 数据集相差并不大。而考虑到原数据集为非平衡数据集，因此，AUC 指标相比 F1 score 和 Accuracy 有着更好的参考意义。因此不难得出，在非平衡数据集的条件下，n-gram TF-IDF 逻辑回归模型采用 Mixed 数据集进行训练时能够取得比 LSTM 模型更好的结果。

§4.4 本章小结

在本章节中，对模型的运行结果进行了讨论，并最终通过横向对比寻找到了综合条件下的最优模型。

从以上的数据可以得出，从模型上来讲，当训练数据集的正负样本不平衡时，使用逻辑回归的机器学习模型效果会比 LSTM 模型效果更好；当使用同一种模型时，Mixed 数据集相比单独的 Text 或 Summary 数据集都更能反应用户的情感。

如果使用手动方式，筛选出等量的正样本和负样本进行训练，又会因为受负样本数量影响不能获取足够的训练样本，导致模型在小数据集上过拟合，即使采用了正则化也不能取得足够好的结果。

因此，在预测正负样本不平衡的数据集，且负样本数量非常小的情况下，使用逻辑回归模型的效果比 LSTM 模型效果更佳。

第 5 章 总结与展望

本章对全文的主要工作和创新点作了总结，并提出研究和改进点。

§5.1 本文总结

本文以亚马逊用户评论数据的正负样本非平衡数据集为研究对象，试图尝试不同的机器学习算法，对比性能并寻找出最优的算法。这一结论会为平台，商家对用户的情感分析得到更加准确的结果提供参考，为研究正负样本不平衡数据集情感预测模型提供有价值的参考。

§5.1.1 本文的主要工作

本文主要工作在于通过多个模型和不同数据集以及不同的词向量化方法，寻找找出最优的机器学习模型以进行用户情感分析。本文主要进行了以下几项工作：

1）模型选取

2）模型评估指标的建立

3）数据清洗与关键词提取

4）模型训练与指标对比

§5.1.2 本文的主要创新点

对比了机器学习和深度学习算法，发现了本文采用的机器学习算法在正负样本不平衡的数据集上训练出的模型在 AUC 指标上有比深度学习算法在本文所建立的模型和训练参数以及训练数据的条件下有更好的表现。探索了正负样本非平衡数据集和小样本学习对模型分类性能的影响。

§5.2 展望

虽然本文实现了寻找正负样本非平衡数据集下的最优的机器学习模型，但仍存在不少需要改进的地方：

（1）机器学习领域发展迅速，近年有很多新的模型诞生，本文并没有使用到业界最新的模型进行训练，采用了比较朴素的经典机器学习分类器和比较旧的深度学习模型进行对比，并不能做到全面的横向对比，之后应增加更多更新的机器学习模型。

（2）本文没有对超参进行过多的调教，笔者没有丰富的超参调整经验，在一定程度上影响了深度学习模型的最好发挥，但是超参的调教是一个依靠经验学习的过程，很难在短时间内有效的掌握。后续可以对超参进行调整以期获得更好

的结果。

致谢

感谢武星老师在课上的悉心教导，感谢同学和室友对我的耐心指导和帮助。

参考文献

[1] Chintagunta P K, Gopinath S, Venkataraman S. The effects of online user reviews on movie box office performance: Accounting for sequential rollout and aggregation across local markets[J]. Marketing science, 2010, 29(5): 944-957.

[2] 王伟，王洪伟. The influence of aspect-based opinions on user's purchase intention using sentiment analysis of online reviews%特征观点对购买意愿的影响:在线评论的情感分析方法[J]. 系统工程理论与实践, 2016, 036(001):63-76.

[3] Picard R W. Affective computing[M]. MIT press, 2000.

[4] Huang X J, Zhao J. Sentiment analysis for Chinese text[C]．Proceedings oftheCommunications of CCF,2008：4(2)．

[5] Riloff E, Shepherd J. A corpus-based approach for building semantic lexicons[J]. arXiv preprint cmp-lg/9706013, 1997.

[6] Hatzivassiloglou V，Mckeown K R．Predicting the semanticorientation of adjectives[C]．Proceedings of the 35thAnnual Meeting of the Association for Computational Linguistics and EighthConference of the European Chapter of the Association for ComputationalLinguistics．Association for Computational Linguistics，1997:174-181．

[7] Tumey P D，Littman M L．Measuring praise and criticism：Inference of semantic orientation from association[J].ACM Transactionson Information Systems，2003，21(4)：315-346．

[8] Sista S．P，Srinivasan S．H．Polarized lexicon for review classification[C．Proceedings of the International Conference on Artificial Intelligence．2004：867-872．

[9] Faye B，Graeme H．Collocations as cues to semanticorientation[C].Proceedings of AAAI Spring Symposium onExploring Attitude and Affect in Text，2004．

[10] Pang B, Lee L, Vaithyanathan S. Thumbs up? Sentiment classification using machine learning techniques[J]. arXiv preprint cs/0205070, 2002.

[11] 张景阳, 潘光友. 多元线性回归与 BP 神经网络预测模型对比与运用研究[J]. 昆明理工大学学报(自然科学版), 2013, 038(006):61-67.

[12] Hinton G E. Distributed representations[J]. 1984.

[13] Gers F A, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM[J]. 1999.

[14] Bradley A P. The use of the area under the ROC curve in the evaluation of machine learning algorithms[J]. Pattern recognition, 1997, 30(7): 1145-1159.

附录：部分源程序清单

```
# 完整代码地址见下述网址
#www.kaggle.com/yuban00018/amazon-food-reviews-data-mining-chinese-co
mments


# 数据清理函数
data_score_removed_netural = df[df['Score']!=3]  # 删除中立评分数据
def partition(x):
    if x < 3:
        return 0
    return 1

# 清除分子大于分母的数据
final = final_data[final_data['HelpfulnessNumerator']
                   <= final_data['HelpfulnessDenominator']]

# 缩写展开
def decontract(text):
    text = re.sub(r"won\'t", "will not", text)
    text = re.sub(r"can\'t", "can not", text)
    text = re.sub(r"n\'t", " not", text)
    text = re.sub(r"\'re", " are", text)
    text = re.sub(r"\'s", " is", text)
    text = re.sub(r"\'d", " would", text)
    text = re.sub(r"\'ll", " will", text)
    text = re.sub(r"\'t", " not", text)
    text = re.sub(r"\'ve", " have", text)
    text = re.sub(r"\'m", " am", text)
    return text

# 自定义停止词
stop_words= set(['br', 'the', 'i', 'me', 'my', 'myself', 'we', 'our',
'ours','ourselves','you',"you're","you've","you'll","you'd",'your',
'yours','yourself','yourselves','he','him','his','himself',  'she',
"she's",'her','hers','herself','it',"it's",'its','itself','they',
'them', 'their','theirs', 'themselves', 'what', 'which', 'who', 'whom',
'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was',
'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do',
'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or',
```

```
'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with',
'about', 'against', 'between', 'into', 'through', 'during', 'before',
'after','above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further','then', 'once', 'here',
'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each',
'few', 'more', 'most', 'other', 'some', 'such', 'only', 'own', 'same',
'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don',
"don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're','ve',
'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't",
'doesn', "doesn't", 'hadn',"hadn't", 'hasn', "hasn't", 'haven',
"haven't",'isn',"isn't",'ma','mightn',"mightn't",'mustn',"mustn't",
'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn',
"wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"])

# 文本清洗
def preprocess_text(review):
    review = re.sub(r"http\S+", "", review)
    # 移除网络连接
    review = BeautifulSoup(review, 'lxml').get_text()
    # 删除 HTML 标签
    review = re.sub("\S*\d\S*", "", review).strip()
    # 删除含有数字的单词
    review = re.sub('[^A-Za-z]+', ' ', review)
    # 移除非英文字符
    review = [word for word in review.split(" ") if not word in stop_words]
    # 删除停止词
    review = [lemmatizer.lemmatize(token, "v") for token in review]
    # 词形还原
    review = " ".join(review)
    review.strip()
    return review

# ROC 曲线绘制
def drawROCCurve(Scores,Y_test,Model):
    dic={"scores" : Scores,"test" : Y_test,"label":Model}
    data=pd.DataFrame(dic)
    plt.figure(figsize=(5,5),dpi=128)
    for row in data.iterrows():
        scores = row[1][0]
        y_test = row[1][1]
        fpr, tpr, thresholds = roc_curve(y_test, scores)
        roc_auc = auc(fpr,tpr) # 计算 auc 的值
        lw = 2
```

```
            plt.plot(fpr, tpr,
                     lw=lw, label=row[1][2]+' (area = %0.5f)' % roc_auc)
            plt.plot([0, 1], [0, 1],
                     color='navy', lw=lw, linestyle='--')
            plt.xlim([0.0, 1.0])
            plt.ylim([0.0, 1.05])
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.title('Receiver operating characteristic')
        plt.legend(loc="lower right")
        plt.show()

# 柱状图绘制
def drawbar(List1,List2,x_label):
    plt.figure(figsize=(6,5))
    plt.title('F1 score and Accuracy')
    plt.ylim(.83, .98)
    plt.bar(range(len(List1)), List1, label='F1 Score')
    plt.bar(range(len(List2)), List2, label='Accuracy')
    plt.legend()
    plt.xticks(np.arange(len(x_label)),
                x_label, rotation=90, fontsize =12)
    plt.show()

# 生成并训练 LSTM 模型
def buildAndFitModel(x_train,y_train,top_words,
                     embedding_vector_length,max_review_length):
    with tpu_strategy.scope():
        model = Sequential()
        model.add(Embedding(top_words+1,
                            embedding_vector_length,
                        input_length=max_review_length))
        model.add(LSTM(128)) # units=100
        model.add(Dropout(0.5))
        model.add(Dense(1, activation='sigmoid'))
        model.compile(loss='binary_crossentropy',
                      optimizer='adam', metrics=['accuracy'])
    model.summary()
    model.fit(x_train,y_train,
              epochs=10, batch_size=64, validation_split=0.3)
    return model

# 绘制深度学习训练结果
```

```
def draw(acc_list,f1_list,fpr_list,
          tpr_list,thresholds_list,roc_auc_list,
          con_mat_df_list,model_list):
    drawbar(f1_list,acc_list,model_list)
    dic = {"fpr":fpr_list,"tpr":tpr_list,\
          "auc":roc_auc_list,"model":model_list}
    data = pd.DataFrame(dic)
    plt.figure(figsize=(5,5),dpi=128)
    for row in data.iterrows():
        fpr = row[1][0]
        tpr = row[1][1]
        roc_auc = row[1][2]
        model = row[1][3]
        lw = 2
        plt.plot(fpr, tpr,
                 lw=lw, label=model+' (area = %0.5f)' % roc_auc)
        plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
        plt.xlim([0.0, 1.0])
        plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic')
    plt.legend(loc="lower right")
    plt.show()

    print("f1 score list:")
    i = 0
    for f1 in f1_list:
        print('%0.4f'%f1,model_list[i])
        i+=1
    i = 0
    print("")
    print("accuracy list:")
    for acc in acc_list:
        print('%0.4f'%acc,model_list[i])
        i+=1
```