

—

# 南昌大学

NANCHANG UNIVERSITY

## 信息内容安全小论文

THESIS OF BACHELOR

(2017 —2022 年)



题 目 基于豆瓣影评的情感分析

学 院: 软件学院 系 信息安全

专业班级: 信息安全 193 班

学生姓名: 丁俊 学号: 8003119100

指导教师: 赵志宾 职称: 副教授

## 目录

摘要 .....	II
第一章 绪论 .....	1
1.1 情感分析的现状和意义 .....	1
1.2 主要研究内容 .....	1
1.3 情感分析历史 .....	2
1.4 国外研究现状 .....	2
1.5 国内相关研究 .....	3
第二章 分析方法和基本流程 .....	5
2.1 基于 Bosonnlp 的情感分析 .....	5
2.2 基于 Snownlp 的情感分析 .....	6
2.3 基于 cnsenti 情感分析 .....	7
2.4 基本流程 .....	8
第三章 实验和分析 .....	9
3.1 爬取数据集 .....	9
3.2 Snownlp 分词 .....	13
3.3 Bosonnlp 情感词典分析步骤 .....	14
3.4 Snownlp 库情感分析步骤 .....	16
3.5 cnsenti 库情感分析步骤 .....	18
3.6 准确度分析 .....	19
第四章 论文总结 .....	21
4.1 实验总结 .....	21
4.1 实验收获 .....	21
第五章 参考文献 .....	22

## 基于豆瓣影评的情感分析

专 业： 信息安全

学 号： 8003119100

学生姓名： 丁俊

指导教师： 赵志宾

### 摘要

随着 21 世纪的到来，大数据时代已经来临，互联网技术逐渐建设完善，网络推动着经济与社会的发展。然而在互联网飞速发展的同时，也带来了大量“爆炸”的信息，在大量移动互联网和智能手机的普及下，用户在社交媒体或相关网站中发布的推文、评论通常包含多种模态数据，例如文本和图片，每个模态都有相互的关联性，如何根据上下文对信息内容的环境特征做出推理描述，提取出信息的关键信息，利用相关构建的情感词典进行数据提取，汇总出情感分析总值，得出文本的大致情感倾向。并根据得到的情感分析数值利用相关工具进行可视化分析得出分布趋势。可以大致对相关网站内容进行情感推断，能够为网络口碑营销、网络舆论导向、事件预测、股票市场分析等诸多领域提供可靠的理论依据。

**关键词：** 信息内容；数据提取；情感分析

## 第一章 绪论

### 1.1 情感分析的现状和意义

如今的互联网技术发展千变万化，但是仍然存在一定的网络信息安全问题，并且这个问题是不容小觑的。随着全球信息化的发展和不断扩大，网络信息安全与国家保护机密、用户个人隐私、社会安全稳定、社会经济发展等居多方面都有着千丝万缕的关系。网络安全问题以及对网络用户的合法权益造成一定困扰。随着社会网站越来越活跃，大量用户使用软件平台和网站注册了个人账户，并不断输送信息和各种丰富的内容，但是随着而来的电脑病毒和木马攻击使得网络变得不安全，严重侵害了用户的合法权益和个人隐私。此时信息内容的安全防护变得至关重要。而在其中对信息内容的分析也愈加突出，如何正确辨认、识别信息的内容和其所处的环境、态度也是面临的重大难题<sup>[1]</sup>。

信息内容情感分析主要基于文本数据，是自然语言处理（NLP）的主要内容，又称意见挖掘、倾向性分析。简答而言，是对带有情感色彩的主观性文本进行分析、处理、推理、归纳的过程。互联网中相关网站上产生了大量充斥着用户各种情感色彩和情感倾向性的信息，比如，喜、怒、哀、乐和批评、赞扬。除此之外，情感分析在大众舆论导向、影评、人物情绪、事件预测等分析领域也慢慢从理论研究拓展到实践中<sup>[2]</sup>。

### 1.2 主要研究内容

首先，本文从真实网站数据出发，设计合理的策略过滤和提取有效的数据集，并对数据集的缺失信息进行补充抓取。在此基础上针对特定信息内容文本的情感分析方法，通过相关实验，例如基于已构建的情感词典的文本情感分析方法，对待分析文本进行文本处理抽取情感词，计算该文本的情感分析值，最终得出文本的情感倾向。并从多方面阐述数据和分析结果，根据不同的分析方法和特征分析情感数据的结构和特点。

### 1.3 情感分析历史

20 世纪 90 年代末, 国外的文本情感分析已经开始。早期, Rioff 和 Shepherd 在文本数据的基础上进行了构建语义词典的相关研究<sup>[9]</sup>。McKeown 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<sup>[10]</sup>, 进而对英文形容词与连词做情感倾向研究。自此之后, 越来越多的研究开始考虑到特征词与情感词的关联关系<sup>[5]</sup>。

### 1.4 国外研究现状

文本情感分析技术是一个新兴的研究领域, 但近几年吸引了众多学者的深入研究, 并在英文评论领域取得了较大进展。根据研究内容的不同, 大致分为以下几个子领域: 文本的情感倾向分类、基于产品特征的情感分析和情感信息的检索技术。

文本情感倾向分类是通过分析相关评论的文本内容, 将评论判断为正面或是负面的评价。这是当前研究最热的课题, 特别是针对网络产品评论文本的情感倾向分类技术已有重大的突破。情感倾向分类主要用于快速判定大众对一个对象的普遍观点。该任务和基于主题的传统文本分类相似, 因此用于文本分类的监督学习方法都可应用到该领域种, 但两者又有所不同。基于主题的文本分类将文档分类到预先定义好的各个不同主题中, 所以主题的相关词汇是分类的重要特征。而在情感倾向分类中, 主题词不再是重要特征, 表达情感倾向的情感词汇成为分类的重要特征。情感倾向分类技术还可细分为: 基于篇章级的情感分类、基于句子级的情感分类和词汇的情感分类<sup>[9-11]</sup>。其中篇章级的情感分类是将整段评论内容作为最小分析单元, 但前提是句子要先提取为主观性句子, 然后才能对这些主观性句子进行褒贬倾向分析。

早期的情感倾向分类始于篇章级的情感分类。现有的研究成果中大部分是基于监督学习的方法, 最早由 Pang 等人在 2002 年用三种经典的机器学习方法来对电影评论进行情感分析。但是由于监督学习方法是通过对人工标记的训练集来训练分类器的。因此, 基于监督学习的方法在跨领域方面存在一定的局限性, Engstrom 等人做了相关的证实研究<sup>[12]</sup>。同时, 此方法的人工工作量太大。随后有了无监督学习的方法才很好地解决了跨领域问题, 无监督学习<sup>[13]</sup>是没有明确目的的训练方式, 你无法提前知道结果是什么, 在没有数据标签的数据里可以发现一些潜在的

结构的训练方式。通过计算文本提取出的关键词和种子词（excellent, poor）的逐点互信息（point-wise mutual information, 简称 PMI）来判断这些关键词的情感趋向。若提取出的关键词的平均语义值是正面的，则判定为正面评论，反之就为负面评论。无监督学习方法的研究并不是很多，且精度远远比不上监督学习的效果。但是能很好地解决跨领域的问题。两种方法都有各自的优缺点。

句子级的情感分析目标是要先提取出主观性的句子，再将主观性句子进一步分为褒义、中性或贬义。该类研究的前提是一个句子只表达了一个主题的观点或看法。Wiebe 等人采用贝叶斯分类器来实现主客观句子的分类<sup>[14]</sup>。Yu 等人采用了句子相似度、朴素贝叶斯分类器和朴素贝叶斯多分类器三种方法来识别主观性句子<sup>[3]</sup>。直接根据句子中的情感词极性判断句子的情感极性类别。除此之外，比较句是另一种类型的评估方式，它直接将一个对象和其他相似对象进行比较。

词汇的情感分类是整个情感分析的基础，即识别单个词的词性语义情感。现有的方法可以分为三大类：人工方法、基于词典扩展的方法和基于语料库的方法。

当判断一个评论是正面的，并不能代表此评论持有者对评论对象的各方面都很满意，因此还需要进行情感分析的细粒度研究。情感检索技术将是未来研究的重点。其目前的研究热点在于意见检索和意见垃圾检测。意见垃圾检测是指有些人为了推销自身产品或服务，或与竞争对手发生的不正当竞争而发表的不符合实际的观点。

## 1.5 国内相关研究

国内针对中文文档的情感分析技术研究与国外针对英文文档的研究相比，起步相对较晚。但近几年也取得了较大的进展。

在基于篇章级的情感分类方面，国内的徐琳宏、林鸿飞等人通过手工和自动生成的方法构建出情感词典本体库，提出一种基于语义的书别方法<sup>[3]</sup>。提取了影响句子情感的 9 个语义特征，对情感分析研究做了初步的探测。他们以褒贬倾向性明显的词汇作为特征值，用支持向量机分类器来判断文本的褒贬性。同时通过否定规则来匹配文本的上一下语义否定以提高情感分析细粒度和精度，还处理了副词附近的情感词以加强对文本情感强度的识别。近年来，闻彬等提出基于文本语义的文本情感分类<sup>[6]</sup>，分析了文本情感倾向是否受到程度副词等出现规律的影响，进而提高了有效判定情感倾向算法的性能。在基于句子级的情感分类方面，娄德成和姚天昉对汉语的网络评论语句进行了语义极性分析和观点抽取<sup>[3]</sup>。通过

分析每个句子的句法结构,提取符合特定语法结构的成分以判断句子的情感倾向<sup>[15]</sup>。黄小江等人对中文比较句进行了相关研究,提出采用支持向量机分类器将评论分为“比较句”和“非比较句”两大类型<sup>[7]</sup>。文中还就中文比较句的特点做了相关讨论。在汉语词汇语义倾向分析研究方面,朱嫣岚等人提出了两种词汇语义倾向计算方法<sup>[8]</sup>,包括基于语义相关场的方法和基于语义相似度的方法,且这两种方法都是基于 HowNet<sup>[9]</sup>的。实验结果证明了在同一测试集上,后者的判断精度高。

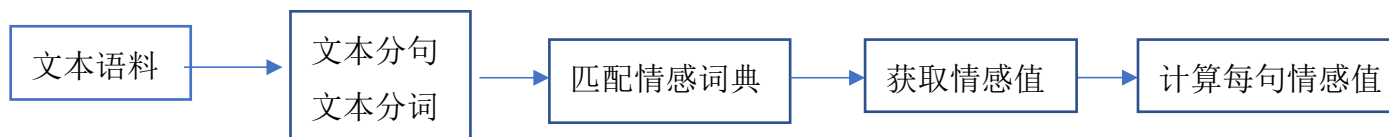
## 第二章 分析方法和基本流程

### 2.1 基于 Bosonnlp 的情感分析

BosonNLP 情感词典是由波森自然语言处理公司推出的一款已经做好标注的情感词典。词典中对每个情感词进行情感评分，BosonNLP 情感词典如下所示：

BosonNLP_sentiment_score.txt - 记事本	BosonNLP_sentiment_score.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)	文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
最尼玛 -6.70400012637	5555555555 -4.66617459217
扰民 -6.49756445867	尼玛阿 -4.66075280423
fuck... -6.32963390433	下泻 -4.64511639079
RNM -6.21861284426	TNND -4.64159586593
wcnmlgb -5.96710044003	100.91万亿 -4.63392712861
2.5: -5.90459648251	3gs -4.63392712861
Fxxk -5.87247473641	555555. -4.63392712861
MLP -5.87247473641	55555555555555555555 -4.63392712861
吃哑巴亏 -5.77120419579	776803828 -4.63392712861
IAQI -5.77107837123	Fxxx -4.63392712861
MLGBD -5.69408191501	OS6.1 -4.63392712861
NNND -5.66228462641	Scheisse -4.63392712861
MLGB. -5.60457743583	WQNMLGBD -4.63392712861
成甘 -5.60457743583	barely -4.63392712861
最桑 -5.60457743583	cao. -4.63392712861
真无语 -5.60457743583	cazzo -4.63392712861
T M -5.60457743583	cnmgb -4.63392712861
次奥次奥次奥 -5.59258287133	mlgbz -4.63392712861
cnmd -5.54446545761	nngx -4.63392712861
MBD -5.50280109843	sleeplessness -4.63392712861
NNDX -5.48173951768	wcnm -4.63392712861

词典中对每个词的极性都赋了一个值，表示情感程度数值。首先需要对文本进行分句、分词，本文选择的分词工具为 jieba 和 snownlp。其次，将分词好的列表数据对应 BosonNLP 词典进行逐个匹配，并记录匹配到的情感词分值。最后，统计计算分值总和，如果分值大于 0，表示情感倾向为积极的；如果小于 0，则表示情感倾向为消极的。原理图如下：



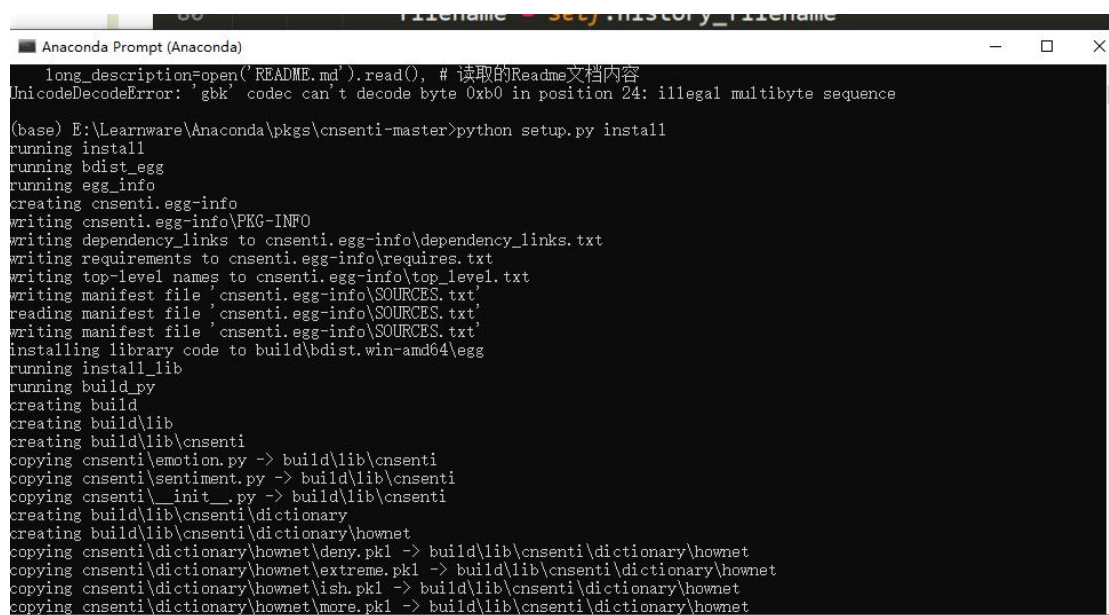


## 2.2 基于 Snownlp 的情感分析

SnowNLP 是一个 python 写的类库，可以方便的处理中文文本内容，是受到了 TextBlob 的启发而写的，由于现在大部分的自然语言处理库基本都是针对英文的，于是写了一个方便处理中文的类库，并且和 TextBlob 不同的是，这里没有用 NLTK，所有的算法都是自己实现的，并且自带了一些训练好的字典。注意本程序都是处理的 unicode 编码，所以使用时请自行 decode 成 unicode。支持的中文自然语言操作包括：

- ✓ 中文分词 (Character-Based Generative Model)
- ✓ 词性标注 (TnT 3-gram 隐马)
- ✓ 情感分析 (现在训练数据主要是买卖东西时的评价，所以对其他的  
一些可能效果不是很好，待解决)
- ✓ 文本分类 (Naive Bayes)
- ✓ 转换成拼音 (Trie 树实现的最大匹配)
- ✓ 繁体转简体 (Trie 树实现的最大匹配)
- ✓ 提取文本关键词 (TextRank 算法)
- ✓ 提取文本摘要 (TextRank 算法)
- ✓ tf, idf
- ✓ Tokenization (分割成句子)
- ✓ 文本相似 (BM25)
- ✓ 支持 python3

在 Anaconda3 中安装 Snownlp，去 Snownlp 的官网下载最新的压缩包并解压在相关目录 pkgs 中，然后执行 setup.py 文件进行安装。



```
Anaconda Prompt (Anaconda)
long_description=open('README.md').read(), # 读取的Readme文档内容
UnicodeDecodeError: 'gbk' codec can't decode byte 0xb0 in position 24: illegal multibyte sequence

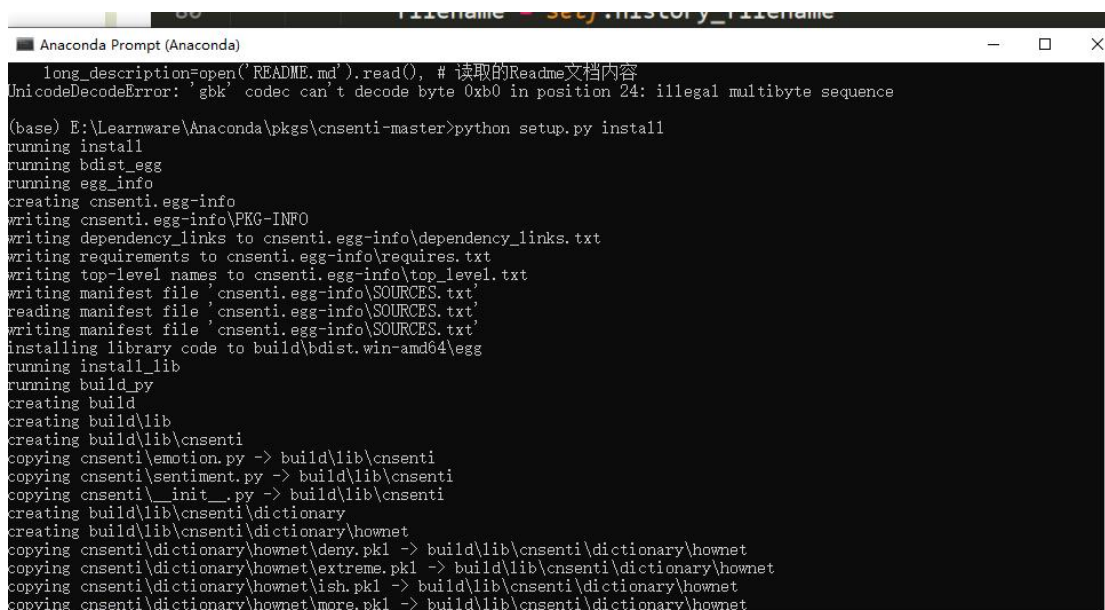
(base) E:\Learnware\Anaconda\pkgs\cnsenti-master>python setup.py install
running install
running bdist_egg
running egg_info
creating cnsenti.egg-info
writing cnsenti.egg-info\PKG-INFO
writing dependency links to cnsenti.egg-info\dependency_links.txt
writing requirements to cnsenti.egg-info\requires.txt
writing top-level names to cnsenti.egg-info\top_level.txt
writing manifest file 'cnsenti.egg-info\SOURCES.txt'
reading manifest file 'cnsenti.egg-info\SOURCES.txt'
writing manifest file 'cnsenti.egg-info\SOURCES.txt'
installing library code to build\bdist.win-amd64\egg
running install_lib
running build_py
creating build
creating build\lib
creating build\lib\cnsenti
copying cnsenti\emotion.py -> build\lib\cnsenti
copying cnsenti\sentiment.py -> build\lib\cnsenti
copying cnsenti\__init__.py -> build\lib\cnsenti
creating build\lib\cnsenti\dictionary
creating build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\deny.pkl -> build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\extreme.pkl -> build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\ish.pkl -> build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\more.pkl -> build\lib\cnsenti\dictionary\hownet
```

SnowNLP 情感分析也是基于情感词典实现的，其简单的将文本分为两类，积极和消极，返回值为情绪的概率，越接近 1 为积极，接近 0 为消极。

## 2.3 基于 cnsenti 情感分析

中文情感分析库（Chinese Sentiments）可对文本进行情绪分析、正负情感分析，默认使用知网的 Hownet。

在 Anaconda3 中安装 cnsenti，去 cnsenti 的官网下载最新的压缩包并解压在相关目录 pkgs 中，然后执行 setup.py 文件进行安装。



```

Anaconda Prompt (Anaconda)
long_description=open('README.md').read(), # 读取的Readme文档内容
UnicodeDecodeError: 'gbk' codec can't decode byte 0xb0 in position 24: illegal multibyte sequence

(base) E:\Learnware\Anaconda\pkgs\cnsenti-master>python setup.py install
running install
running bdist_egg
running egg_info
creating cnsenti.egg-info
writing cnsenti.egg-info\PKG-INFO
writing dependency_links to cnsenti.egg-info\dependency_links.txt
writing requirements to cnsenti.egg-info\requires.txt
writing top-level names to cnsenti.egg-info\top_level.txt
writing manifest file 'cnsenti.egg-info\SOURCES.txt'
reading manifest file 'cnsenti.egg-info\SOURCES.txt'
writing manifest file 'cnsenti.egg-info\SOURCES.txt'
installing library code to build\bdist.win-amd64\egg
running install_lib
running build_py
creating build
creating build\lib
creating build\lib\cnsenti
copying cnsenti\emotion.py -> build\lib\cnsenti
copying cnsenti\sentiment.py -> build\lib\cnsenti
copying cnsenti\__init__.py -> build\lib\cnsenti
creating build\lib\cnsenti\dictionary
creating build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\deny.pkl -> build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\extreme.pkl -> build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\ish.pkl -> build\lib\cnsenti\dictionary\hownet
copying cnsenti\dictionary\hownet\more.pkl -> build\lib\cnsenti\dictionary\hownet

```

Cnsenti 包括 Emotion 和 Sentiment 两大类，其中包括：

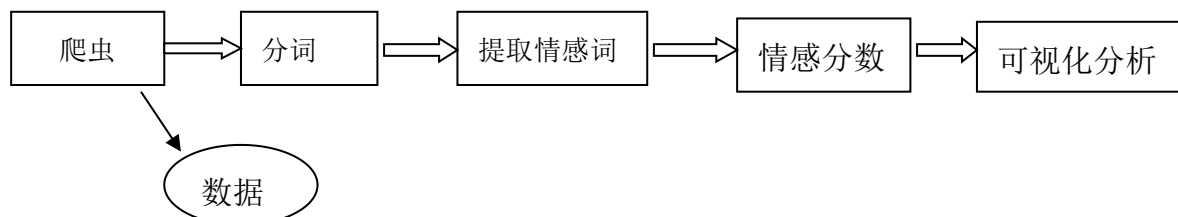
- ✓ Emotion 情绪计算类，包括 emotion\_count()方法，用于统计各种文本中情绪形容词出现的词语数，支持七种情绪统计。如{'words': 22, 'sentences': 2, '好': 0, '乐': 4, '哀': 0, '努': 0, '惧': 0, '恶': 0, '惊': 0}。其中 words 表示中文文本的词语数，sentences 表示句子数，其他情绪词表示文本中各自情绪词语出现的数量。
- ✓ Sentiment 正负情感计算类，包括 sentimentcount()和 sentimentcalculate()方法。Sentimentcount()方法可对文本中的正、负面词进行统计。默认使用 Hownet 词典，输出的结果中 words 表示文本中词语数、sentences 表示句子数、pos 表示正面词总个数、neg 是负面词的个数。

Sentiment\_calculate()可更加精准的计算文本情感信息，还考虑了情感词前后是否有强度副词的修饰和否定词的反转作用。

## 2.4 基本流程

本次情感分析的基本流程包括：

- 自定义爬虫脚本抓取文本信息
- 使用 `jieba` 或 `snownlp` 工具进行中文分词、词性标注
- 定义使用情感词典提取每行文本的情感词
- 通过情感词构建情感矩阵，并计算情感分数
- 结果评估，对情感分数数据进行可视化分析、比较



通过 `Bosonnlp` 情感词典、`Snownlp` 文本分析库、`cnsenti` 情绪分析库分别对数据集进行情感分析，对得出的情感数值进行可视化分析和比较，分析各工具的数据分布情况和准确度比较。

## 第三章 实验和分析

## 3.1 爬取数据集

首先我们确定要分析数据集合，在本次实验中我把豆瓣电影《尚气与十环传奇》的影评抓取 400 条进行分析，每条评论分别有评论者、评论内容、评分三条记录。通过 cookies 记录登录状态，利用 python 的 requests 和 BeautifulSoup 模块对网页进行分析提取，循环 20 次，传递参数给 url 进行多页循环爬取，每次爬取 20 条评论，然后写入到 csv 文件，部分过程图和代码如下。



```

1. def crawl_data(num):
2.     url = 'https://movie.douban.com/subject/30394797/comments?start=' +
str(num) + '&limit=20&status=P&sort=new_score'
3.     header = {
4.         'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) Apple
WebKit/537.36 (KHTML, like Gecko) '
5.         'Chrome/83.0.4103.116 Safari/537.36'}
6.     cookie_str = 'bid=-HgvaA7tUJI; __utma=30149280.1742658929.1638
184733.1638184733.1638184733.1; \
7.         '__utmb=30149280.0.10.1638184733; __utmc=30149280; __u
tmz=30149280.1638184733.1.1.utmcsr=(' \
8.         'direct)|utmccn=(direct)|utmcmd=(none); ' \
9.         '__utma=223695111.91194364.1638184733.1638184733.1638
184733.1; __utmb=223695111.0.10.1638184733; ' \

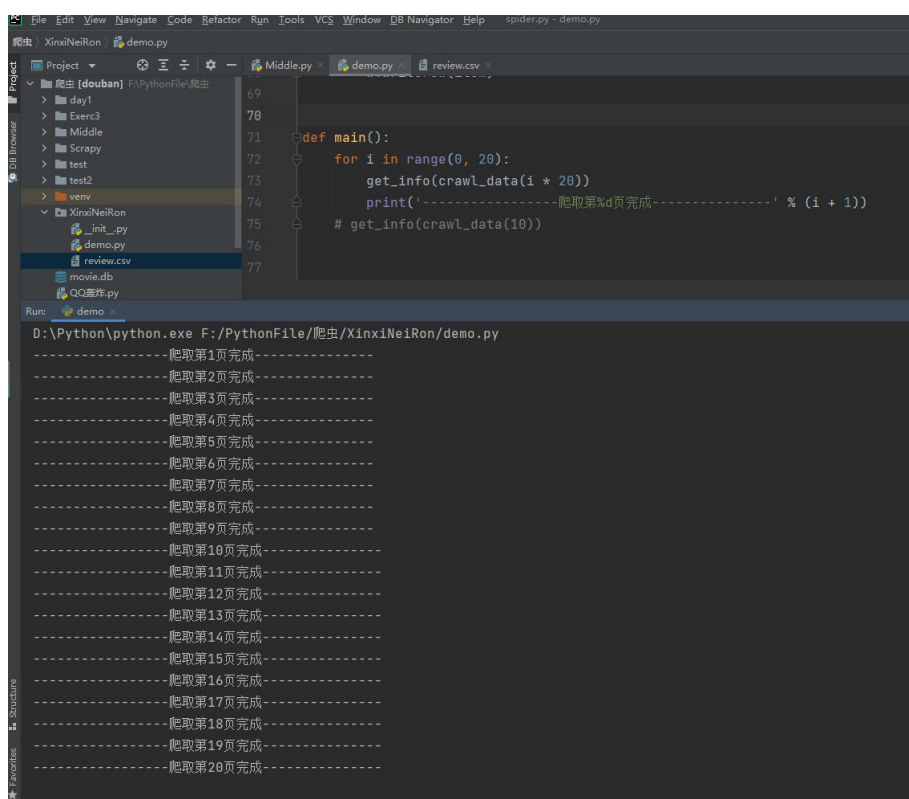
```

```

10.         '__utmc=223695111; __utmz=223695111.1638184733.1.1.ut
mcsr=(direct)|utmccn=(direct)|utmcmd=(none); ' \
11.         'ap_v=0,6.0; _pk_ses.100001.4cf6=*; dbcl2="250780697:ZloP
UIFBAYs"; ck=HtZI; ' \
12.         '_pk_id.100001.4cf6=c0908f39cb6c37ba.1638184733.1.16381
85041.1638184733.; push_noty_num=0; ' \
13.         'push_doumail_num=0 '
14.     cookies = {}
15.     # 处理 cookies
16.     for line in cookie_str.split(';'):
17.         name, value = line.strip().split('=', 1)
18.         cookies[name] = value
19.     # url = 'https://movie.douban.com/subject/30394797/comments?status
=P'
20.     # all_url = 'https://movie.douban.com/subject/30394797/comments?sta
rt=20&limit=20&status=P&sort=new_score'
21.     try:
22.         res = requests.get(url, headers=header, cookies=cookies)
23.         return res.text
24.     except Exception as e:
25.         # print(e)
26.         print('错误')
27.
28.
29. def get_info(text):
30.     bs = BeautifulSoup(text, 'xml')
31.     con_list = bs.select('.comment-item')
32.     review = {}
33.     for con in con_list:
34.         review['author'] = con.select('span.comment-info a')[0].text
35.         review['content'] = con.select('span.short')[0].text.strip()
36.         if con.select('span.rating'):
37.             review['score'] = (con.select('span.rating')[0].get('title'))

```

```
38.     else:
39.         review['score'] = '未知'
40.     write_to_file(review)
41. # 将字典形式的内容写入文件
42. def write_to_file(item):
43.     with open('review.csv', 'a', encoding='utf_8_sig', newline=") as f:
44.         fieldnames = ['author', 'content', 'score'] # 内容和分数
45.         w = csv.DictWriter(f, fieldnames=fieldnames)
46.         w.writerow(item)
```

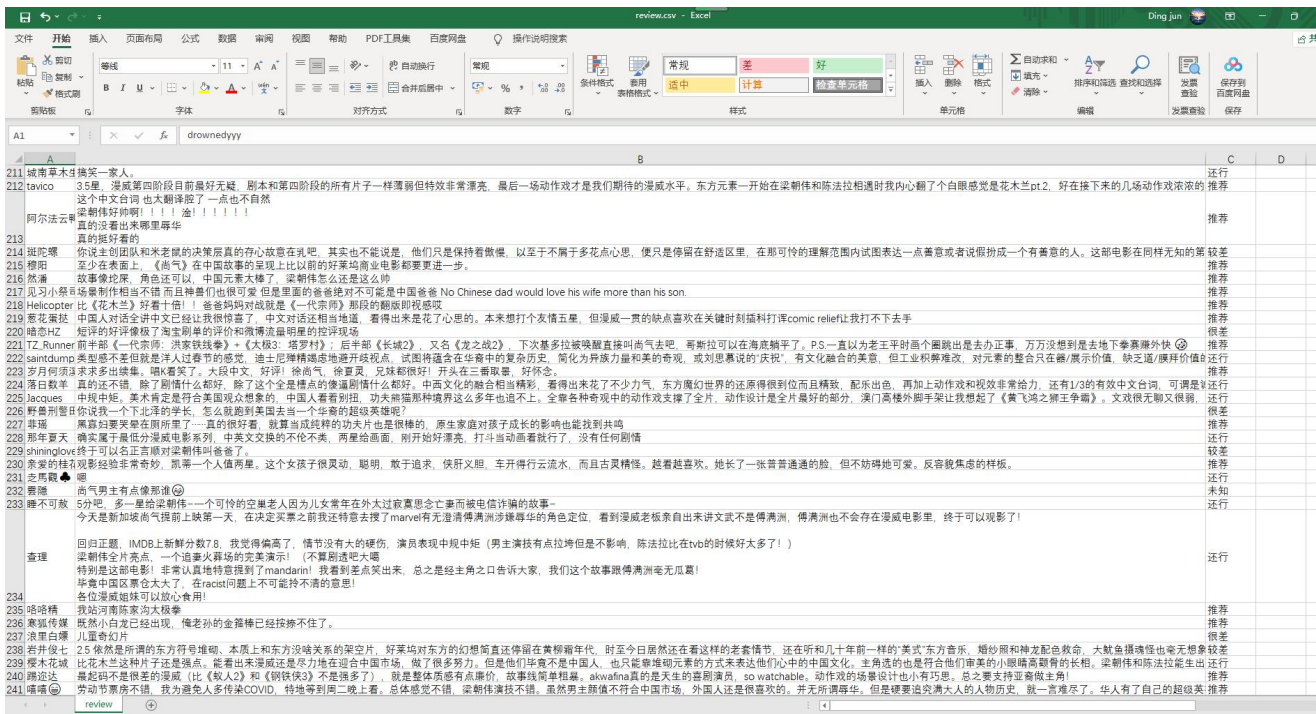


The screenshot shows an IDE with a project named '爬虫' (Crawling) containing a sub-project 'XinxiNeiRon'. The file 'demo.py' is open, showing a `main` function that loops 20 times, calling `get_info(crawl_data(i * 20))` and printing progress. The output window shows the script running successfully, printing 20 lines of progress: '爬取第1页完成' through '爬取第20页完成'.

```
D:\Python\python.exe F:/PythonFile/爬虫/XinxiNeiRon/demo.py
-----爬取第1页完成-----
-----爬取第2页完成-----
-----爬取第3页完成-----
-----爬取第4页完成-----
-----爬取第5页完成-----
-----爬取第6页完成-----
-----爬取第7页完成-----
-----爬取第8页完成-----
-----爬取第9页完成-----
-----爬取第10页完成-----
-----爬取第11页完成-----
-----爬取第12页完成-----
-----爬取第13页完成-----
-----爬取第14页完成-----
-----爬取第15页完成-----
-----爬取第16页完成-----
-----爬取第17页完成-----
-----爬取第18页完成-----
-----爬取第19页完成-----
-----爬取第20页完成-----
```



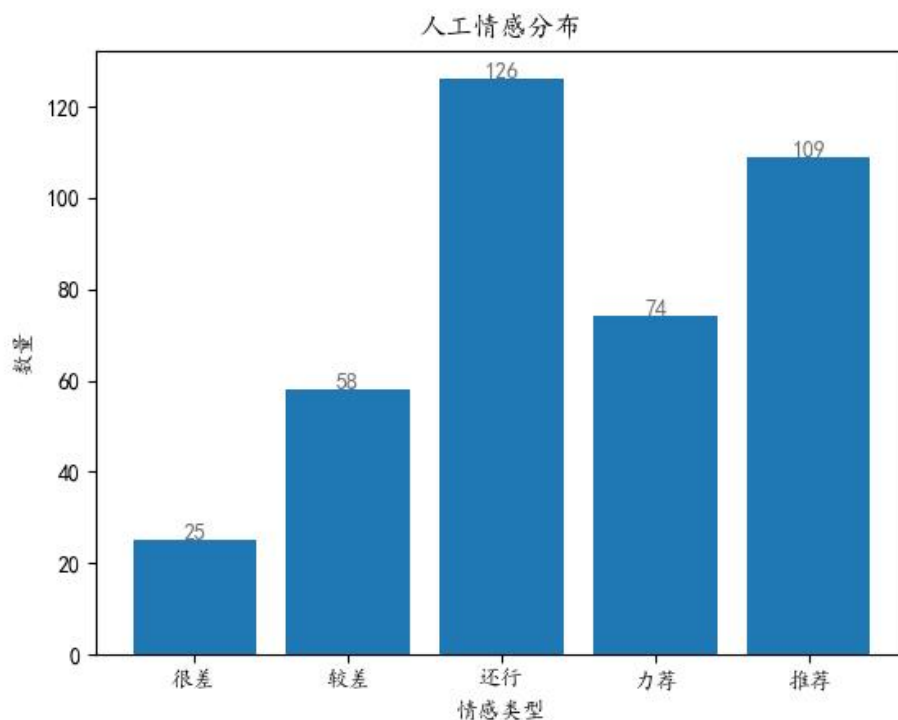
### 第三章 实验和分析



为了使得情感分析的结果有个参考，查看结果分析得准确度，我们把原数据集中的不同评论者对电影的不同态度类型的数量进行划分和统计，分别有 5 种态度：“很差”、“较差”、“还行”、“推荐”、“力荐”。记录个情感态度的数量后，利用 matplotlib 条形图描绘出来。

```
1. def get_score():
2.     matplotlib.rcParams['font.sans-serif'] = ['KaiTi']
3.     df_score = pd.read_csv('review.csv', encoding='utf-8', names=['author',
'content', 'score'])
4.     # print(df_score['content'].head(10))
5.     # print(df_score['score'].head(10))
6.     score_all = {'很差': df_score[df_score['score'] == '很差'].shape[0], '较差': df_score[df_score['score'] == '较差'].shape[0],
'还行': df_score[df_score['score'] == '还行'].shape[0],
'力荐': df_score[df_score['score'] == '力荐'].shape[0], '推荐': df_score[df_score['score'] == '推荐'].shape[0]}
9.     x = list(score_all.keys())
10.    y = list(score_all.values())
11.    plt.bar(range(5), y, tick_label=x)
12.    for x, y in enumerate(y):
```

```
13. plt.text(x, y + 0.1, '%s' % round(y, 1), ha='center', color='#6D6D6D')
14. plt.title('人工情感分布')
15. plt.xlabel('情感类型')
16. plt.ylabel('数量')
17. plt.show()
```



### 3.2SnowNlp 分词

在进行情感分析之前，我们要先进行分词操作，将每一行句子分成若干个中文词。

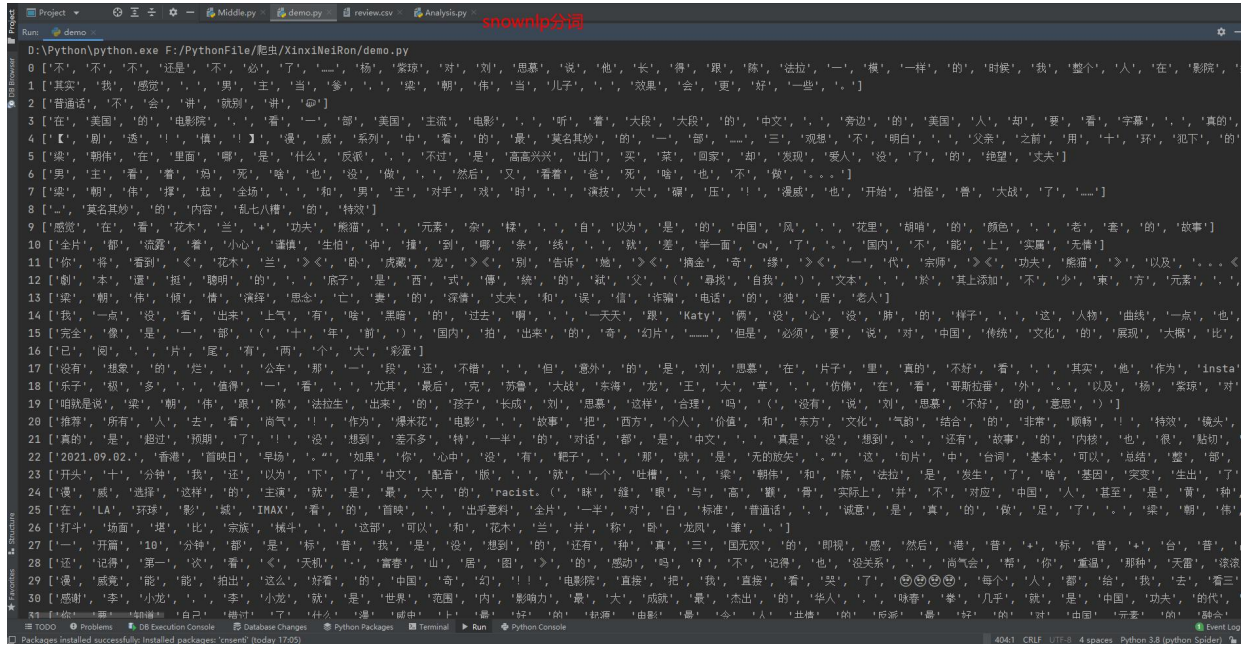
```
18. def snownlp():
19.     df_data = pd.read_csv('review.csv', encoding='utf-8', names=['author', '
content', 'score'])
20.     l_bowl = []
21.     for t in df_data.iloc[:, 1]:
```



```

22.     s = SnowNLP(t)
23.     segs = s.words
24.     l_bowl.append(segs)
25.     return l_bowl

```



### 3.3 Bosonnlp 情感词典分析步骤

基于情感分析得出的数据做出分布条形图

```

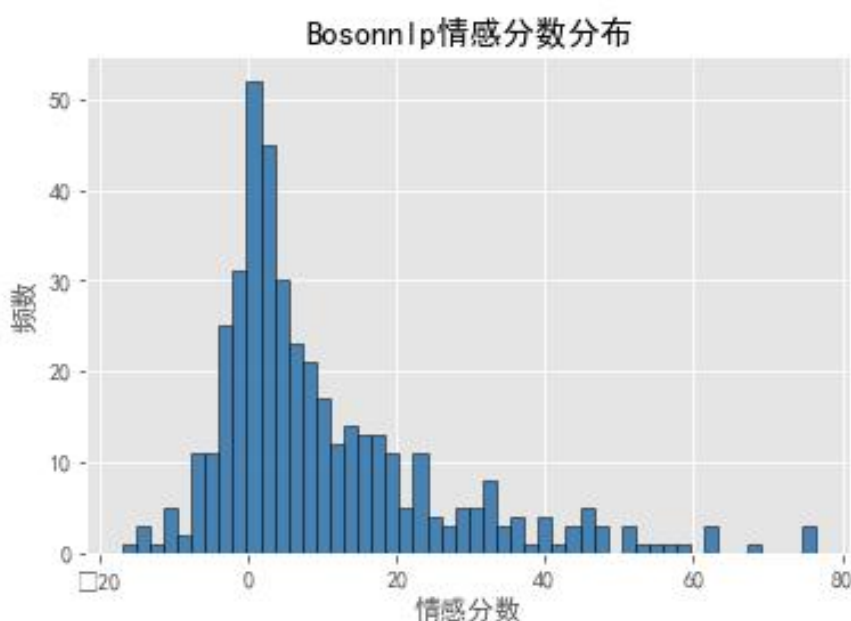
26. def Bosonnlp():
27.     l_bow = snownlp()
28.     df_score = pd.read_csv('BosonNLP_sentiment_score.txt', sep=' ', name
s=['word', 'score'], header=None)
29.     l_seg1 = []
30.     ss = 0
31.     for b in l_bow: # b 是一个评论句子
32.         score = 0
33.         for w in b: # w 是其中分词的一个词
34.             for s in df_score.itertuples():
35.                 if w == s[1]: # 找到相同的字段
36.                     score = score + s[2] # 对应的情感数值

```

```

37.         break
38.     matplotlib.rcParams['font.sans-serif'] = ['SimHei']
39.     plt.style.use('ggplot')
40.     plt.hist(x=l_seg1, bins=50, color='steelblue', edgecolor='black')
41.     plt.xlabel('情感分数')
42.     plt.ylabel('频数')
43.     plt.title('Bosonnlp 情感分数分布')
44.     plt.show()

```



基于描绘出来的 Bosonnlp 情感分数分布图和原始数据的人工情感分布图可以看出，趋势大致相同，上图中数据分布在 0 的周围比较多，这个和原始数据还是比较相像的。但对于 0 两边的数据分布还是有点不准确，“力荐”和“推荐”的数据要更多，在 Bosonnlp 的图中没有体现右边数据比左边高的趋势，而是比较松散地分布在大于 0 的区间，不能很好地看出对电影积极评价比消息多的情感。但是总体来看还是情感正向的比较多。

将 Bosonnlp 情感分析结果写入文件中保存。

```

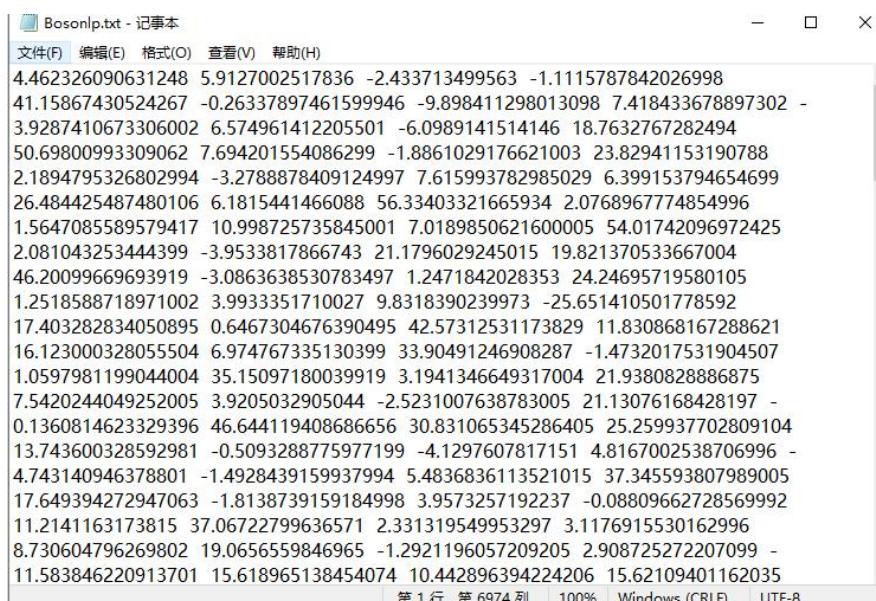
45. for b in l_bow: # b 是一个评论句子
46.     score = 0
47.     for w in b: # w 是其中分词的一个词

```

```

48.     for s in df_score.itertuples():
49.         if w == s[1]: # 找到相同的字段
50.             score = score + s[2] # 对应的情感数值
51.         break
52.     print(score, end=' ')
53.     write_data('Bosonlp.txt',str(score))

```



The screenshot shows a Notepad window titled 'Bosonlp.txt - 记事本'. The text inside is a list of words followed by their sentiment scores, separated by spaces. The words are in Chinese, and the scores are numerical values. The list is long, with many lines of text. The window has a standard menu bar with '文件(F)', '编辑(E)', '格式(O)', '查看(V)', and '帮助(H)'. The status bar at the bottom shows '第 1 行 第 6974 列 100% Windows (CR) D UTF-8'.

### 3.4 Snownlp 库情感分析步骤

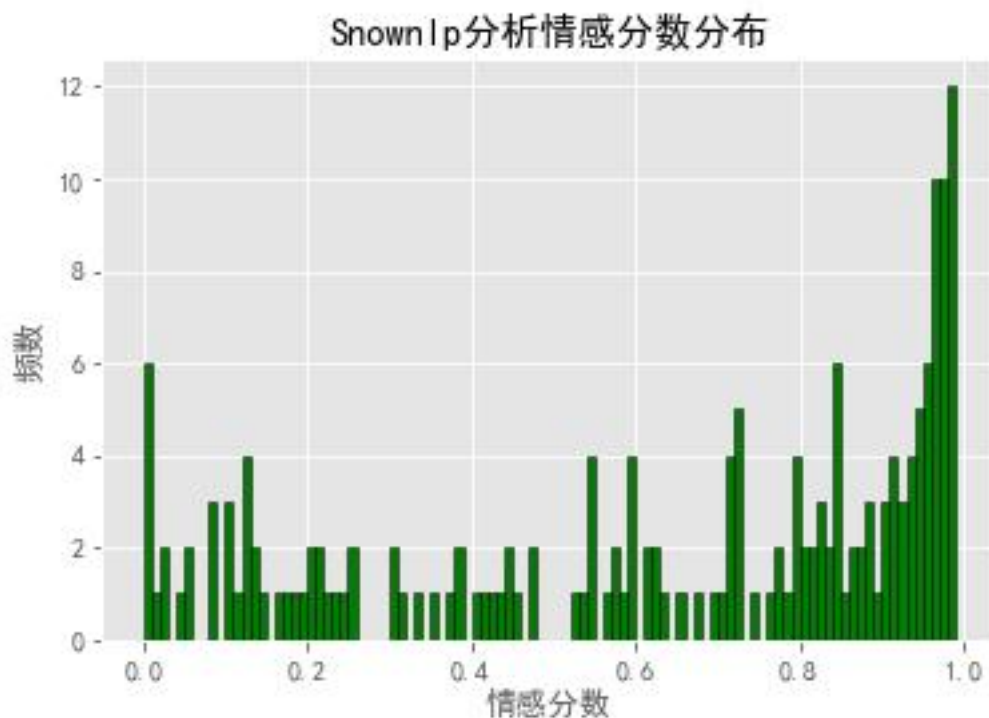
```

54. from snownlp import SnowNLP
55. l_seg2 = []
56. for t in df_data.iloc[:,0]:
57.     s = SnowNLP(t)
58.     seg = s.sentiments
59.     l_seg2.append(seg)
60. matplotlib.rcParams['font.sans-serif'] = ['SimHei']
61. plt.style.use('ggplot')
62. plt.hist(x = l_seg2, bins = 30, color = 'steelblue', edgecolor = 'black')
63. plt.xlabel('情感分数')
64. plt.ylabel('频数')

```

```
65. plt.title('Snownlp 分析情感分数分布')
```

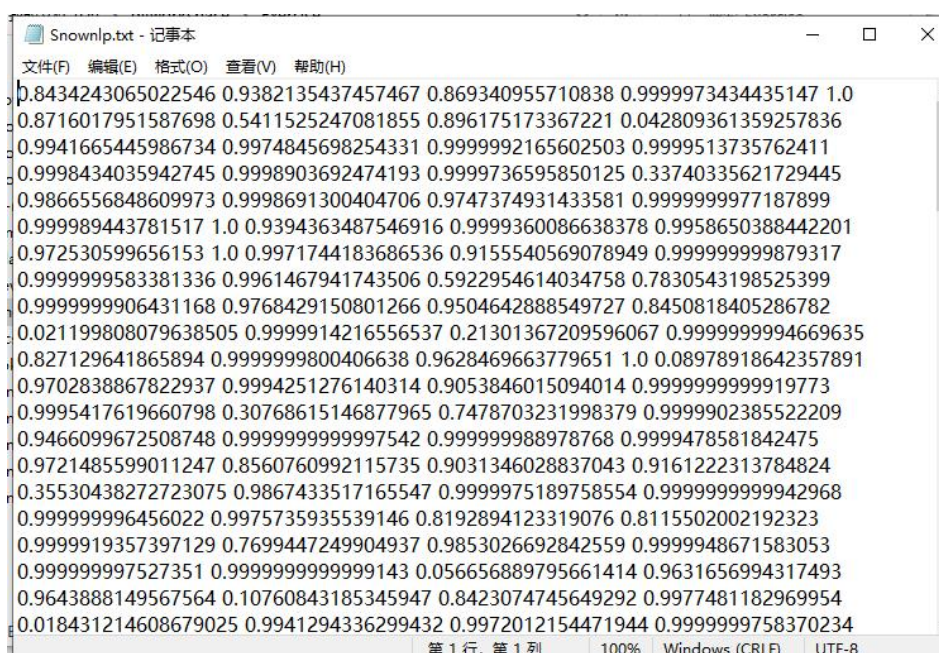
```
66. plt.show()
```



基于描绘出来的 Snownlp 情感分数分布图和人工情感分布图来看,大致是看不出具体趋势的。在 Snownlp 中越接近于 0 表示消极、越接近于 1 表示积极,但是并没有具体的评判标准,比如图中的接近于 1 的数据量远远超过了其他范围分数的数量,这些数据中肯定也会有一些中性的数据由于分析不准而造成的误差。从人工分布图看,中性(“还行”)是数量最多的,但是该图中中性和积极的态度差别无法体现出来。所以从这个图来看, Snownlp 对于该数据集无疑是不准确、不适合的。

将 Snownlp 情感分析数据写入到文件中。

```
67. for t in df_data.iloc[:,1]:
68.     s = SnowNLP(t)
69.     seg = s.sentiments
70.     l_seg2.append(seg)
71.     print(seg,end=' ')
72.     write_data('Snownlp.txt',str(seg) + '')
```



```

0.8434243065022546 0.9382135437457467 0.869340955710838 0.9999973434435147 1.0
0.8716017951587698 0.5411525247081855 0.896175173367221 0.042809361359257836
0.9941665445986734 0.9974845698254331 0.9999992165602503 0.9999513735762411
0.9998434035942745 0.9998903692474193 0.9999736595850125 0.33740335621729445
0.9866556848609973 0.9998691300404706 0.9747374931433581 0.9999999977187899
0.999989443781517 1.0 0.9394363487546916 0.9999360086638378 0.9958650388442201
0.972530599656153 1.0 0.9971744183686536 0.9155540569078949 0.999999999879317
0.9999999583381336 0.9961467941743506 0.5922954614034758 0.7830543198525399
0.9999999906431168 0.9768429150801266 0.9504642888549727 0.8450818405286782
0.021199808079638505 0.9999914216556537 0.21301367209596067 0.9999999994669635
0.827129641865894 0.9999999800406638 0.9628469663779651 1.0 0.08978918642357891
0.9702838867822937 0.9994251276140314 0.9053846015094014 0.999999999919773
0.9995417619660798 0.30768615146877965 0.7478703231998379 0.9999902385522209
0.9466099672508748 0.9999999999997542 0.999999988978768 0.9999478581842475
0.9721485599011247 0.8560760992115735 0.9031346028837043 0.9161222313784824
0.35530438272723075 0.9867433517165547 0.9999975189758554 0.999999999942968
0.999999996456022 0.9975735935539146 0.8192894123319076 0.8115502002192323
0.9999919357397129 0.7699447249904937 0.9853026692842559 0.9999948671583053
0.999999997527351 0.999999999999143 0.056656889795661414 0.9631656994317493
0.9643888149567564 0.10760843185345947 0.8423074745649292 0.9977481182969954
0.018431214608679025 0.9941294336299432 0.9972012154471944 0.9999999758370234

```

Snownlp 情感分析时要更快，所画的时间相对 Bosonnlp 较少。

### 3.5 cnsenti 库情感分析步骤

```

73. from cnsenti import Sentiment
74. l_seg1 = []
75. l_seg2 = []
76. senti = Sentiment()
77. for t in df_data.iloc[:,1]:
78.     seg1 = senti.sentiment_count(t)
79.     seg2 = senti.sentiment_calculate(t)
80.     l_seg1.append(seg1)
81.     l_seg2.append(seg2)

```



```

In [21]: l_seg2
Out[21]: [{'sentences': 1, 'words': 29, 'pos': -21.0, 'neg': 0.0},
{'sentences': 1, 'words': 16, 'pos': 0, 'neg': 0},
{'sentences': 1, 'words': 7, 'pos': 0, 'neg': 6},
{'sentences': 1, 'words': 34, 'pos': 0, 'neg': 0},
{'sentences': 5, 'words': 229, 'pos': 140.0, 'neg': 984.0},
{'sentences': 1, 'words': 21, 'pos': 44.0, 'neg': 0.0},
{'sentences': 1, 'words': 19, 'pos': 0, 'neg': 0},
{'sentences': 2, 'words': 22, 'pos': 0, 'neg': 0},
{'sentences': 1, 'words': 7, 'pos': 0, 'neg': 9},
{'sentences': 1, 'words': 23, 'pos': 0, 'neg': 12},
{'sentences': 2, 'words': 23, 'pos': 13, 'neg': 28},
{'sentences': 2, 'words': 47, 'pos': 0.0, 'neg': 34.0},
{'sentences': 4, 'words': 154, 'pos': 606.0, 'neg': 44.0},
{'sentences': 1, 'words': 15, 'pos': 9, 'neg': 0},
{'sentences': 1, 'words': 41, 'pos': 0.0, 'neg': -16.0},
{'sentences': 1, 'words': 50, 'pos': 63.0, 'neg': 0.0},
{'sentences': 1, 'words': 7, 'pos': 0, 'neg': 0},
{'sentences': 1, 'words': 61, 'pos': 0.0, 'neg': 245.0},
{'sentences': 3, 'words': 45, 'pos': 19, 'neg': 0},
{'sentences': 1, 'words': 27, 'pos': 59.0, 'neg': 12.0},

In [21]: l_seg2
{'sentences': 9, 'words': 109, 'pos': 23.0, 'neg': 17.0},
{'sentences': 3, 'words': 94, 'pos': 128.0, 'neg': 102.0},
{'sentences': 3, 'words': 122, 'pos': 360.0, 'neg': 387.0},
{'sentences': 2, 'words': 32, 'pos': 6.0, 'neg': 10.0},
{'sentences': 2, 'words': 12, 'pos': 3, 'neg': 4},
{'sentences': 1, 'words': 7, 'pos': 0, 'neg': 0},
{'sentences': 4, 'words': 58, 'pos': 4.0, 'neg': 14.5},
{'sentences': 13, 'words': 163, 'pos': 153.0, 'neg': 27.0},
{'sentences': 1, 'words': 26, 'pos': 106.0, 'neg': 62.0},
{'sentences': 1, 'words': 22, 'pos': 7, 'neg': 0},
{'sentences': 2, 'words': 56, 'pos': 379.0, 'neg': 21.0},
{'sentences': 4, 'words': 78, 'pos': 227.0, 'neg': 0.0},
{'sentences': 4, 'words': 40, 'pos': 106, 'neg': 28},
{'sentences': 3, 'words': 55, 'pos': 0.0, 'neg': 4.0},
{'sentences': 5, 'words': 77, 'pos': -7.0, 'neg': 23.0},
{'sentences': 2, 'words': 93, 'pos': 240.0, 'neg': 81.0},
{'sentences': 3, 'words': 43, 'pos': 1, 'neg': 0},
{'sentences': 3, 'words': 63, 'pos': 130.0, 'neg': 21.0},
{'sentences': 1, 'words': 11, 'pos': 0, 'neg': 0},
{'sentences': 1, 'words': 42, 'pos': 111.0, 'neg': 5.0},

In [22]: l_seg1
Out[22]: [{'words': 29, 'sentences': 1, 'pos': 1, 'neg': 0},
{'words': 16, 'sentences': 1, 'pos': 0, 'neg': 0},
{'words': 7, 'sentences': 1, 'pos': 0, 'neg': 1},
{'words': 34, 'sentences': 1, 'pos': 0, 'neg': 0},
{'words': 229, 'sentences': 5, 'pos': 6, 'neg': 10},
{'words': 21, 'sentences': 1, 'pos': 2, 'neg': 1},
{'words': 19, 'sentences': 1, 'pos': 0, 'neg': 0},
{'words': 22, 'sentences': 2, 'pos': 0, 'neg': 0},
{'words': 7, 'sentences': 1, 'pos': 0, 'neg': 2},
{'words': 23, 'sentences': 1, 'pos': 0, 'neg': 1},
{'words': 23, 'sentences': 2, 'pos': 1, 'neg': 4},
{'words': 47, 'sentences': 2, 'pos': 0, 'neg': 2},
{'words': 154, 'sentences': 4, 'pos': 3, 'neg': 2},
{'words': 15, 'sentences': 1, 'pos': 1, 'neg': 0},
{'words': 41, 'sentences': 1, 'pos': 0, 'neg': 1},
{'words': 50, 'sentences': 1, 'pos': 2, 'neg': 0},
{'words': 7, 'sentences': 1, 'pos': 0, 'neg': 0},
{'words': 61, 'sentences': 1, 'pos': 1, 'neg': 3},
{'words': 45, 'sentences': 3, 'pos': 1, 'neg': 0},
{'words': 27, 'sentences': 1, 'pos': 3, 'neg': 2},

In [22]: l_seg1
{'words': 109, 'sentences': 9, 'pos': 4, 'neg': 4},
{'words': 94, 'sentences': 3, 'pos': 2, 'neg': 5},
{'words': 122, 'sentences': 3, 'pos': 9, 'neg': 5},
{'words': 32, 'sentences': 2, 'pos': 1, 'neg': 2},
{'words': 12, 'sentences': 2, 'pos': 1, 'neg': 1},
{'words': 7, 'sentences': 1, 'pos': 0, 'neg': 0},
{'words': 58, 'sentences': 4, 'pos': 5, 'neg': 2},
{'words': 163, 'sentences': 13, 'pos': 4, 'neg': 4},
{'words': 26, 'sentences': 1, 'pos': 5, 'neg': 4},
{'words': 22, 'sentences': 1, 'pos': 1, 'neg': 0},
{'words': 56, 'sentences': 2, 'pos': 11, 'neg': 1},
{'words': 78, 'sentences': 4, 'pos': 6, 'neg': 0},
{'words': 40, 'sentences': 4, 'pos': 4, 'neg': 2},
{'words': 55, 'sentences': 3, 'pos': 0, 'neg': 1},
{'words': 77, 'sentences': 5, 'pos': 5, 'neg': 3},
{'words': 93, 'sentences': 2, 'pos': 2, 'neg': 2},
{'words': 43, 'sentences': 3, 'pos': 1, 'neg': 0},
{'words': 63, 'sentences': 3, 'pos': 6, 'neg': 2},
{'words': 11, 'sentences': 1, 'pos': 0, 'neg': 0},
{'words': 42, 'sentences': 1, 'pos': 7, 'neg': 1},

```

列表 `l_seg1` 中存储的是 `sentiment_count()` 函数分析的结果，`l_seg2` 中存储的是 `sentiment_calculate()` 函数分析的结果。可以看出 `l_seg2` 输出的数据中计算文本的情感数值大小范围都比较大，相比更加精准，而 `sentiment_count()` 函数只考虑句子中正面和负面词性出现的个数，因为 `sentiment_calculate` 还考虑了情感词前是否有强度副词的修饰作用和否定词的情感语义反转作用，所以得出的数值才更精准，在此，我基于 `sentiment_calculate()` 函数进行情感分数分析。

### 3.6 准确度分析

我们把上面实验中保存的“Bosonnlp.txt”中的数字提取出来转化为 `float` 类型，统计大于 0 的个数为 326 个、小于 0 的个数为 74 个。说明情感消极的句子和原始数据的 83（25+58）个差不多。但是对于中性和积极的感情没有区分的标准。但当我们把“<0”的情感值归于消极、“0~8”的情感值归于中性、“>8”

的值归于积极的时候，其数值分别为 74、127、199（从消极到积极），和原始情感分布很相似。说明 Bosonnlp 还是有一定的准确性的，只是量化细粒度不够准确和仔细。

对于 Snownlp 画出的情感分布图来看，大致还是接近 1 的柱形要多，即态度呈积极的较多，如果以 0.5 为边界，大于 0.5 代表积极、小于 0.5 代表消极，则积极有 330 个，消极为 70 个。但这是基于我们把用户中“还行”的评论当作好评看待的情况下才有的，这并不能说明什么，因为 Snownlp 分析的情感数值无法从接近 0 或 1 的程度来划分积极、消极程度的等级。

通过 cnsetnti 的词性分析得出消极、中性、积极词性的数量分别为 130、73、197。通过实验结果分析，在此数据集中，对于积极态度的判断较为准确，而对于中性和消极态度的情感不是特别准确。

```
: high = 0
mid = 0
low = 0
for seg in l_seg2:
    if seg['pos'] > seg['neg']:
        high = high + 1
    elif seg['pos'] == seg['neg']:
        mid = mid + 1
    elif seg['pos'] < seg['neg']:
        low = low + 1
print(low)
print(mid)
print(high)

130
73
197
```

综合上述来看，针对该数据集进行不同工具的情感分析结果分析，Bosonnlp 情感词典较为准确，不管是情感分数的条形分布，还是从最后准确度的分析来看，都比较符合原始数据的情感状态分布，不足的是对于中性和积极的情感无法给出一个完全可行的界限。

## 第四章 论文总结

### 4.1 实验总结

互联网的广泛应用促使电子商务的蓬勃发展，各自网络论坛中出现大量有关产品的评论。这些评论中有很多有价值的东西。商家或发行者可以从中分析自己的产品优缺点，从而给出一些建议和改进，消费者可参考这些信息来优化自己的决策和选择。文本情感分析应运而生。

本次实验是基于豆瓣网站电影《尚气与十环传奇》影评数据进行的情感分析，通过自定义爬虫爬取了 400 条用户评论，然后进行数据筛选和选择，形成稳定的数据集。然后对每个句子进行分词后，采用 Bosonnlp 情感词典、Snownlp 文本分析库和 cnssenti 中文情绪分析库对每个句子进行情感分析。将分析得到的情感数值存入文件中。随后对这些情感数值进行可视化分析，描绘出情感分布图，分析和比较各工具的数据结果的特点和准确度。从而对文本的情感分析有了更深的了解和掌握。虽然每一个分析工具都不能很准确地匹配到原始的评论情感数值，但从情感分布来看大体的趋势相同，只是每一种的表示方法不尽相同。在 Bosonnlp 情感分析方法中，积极和消极的区别明显，但是中性态度体现不出来；在 Snownlp 情感分析中，积极和消极的程度不能定量突出，情感分析数值分布比较密切，不能很好的区分正反情感；在 cnssenti 情感分析中可以具体的知道每个句子中积极与消极词性的比例，可以较好的判断句子的情感。

### 4.1 实验收获

在这次实验中，利用到了一些的爬虫技术和数据分析技术，通过这些工具大大地方便和简化了情感分析的过程。当然也遇到了一些问题，比如筛选数据集不准、情感分析数据处理不当等等。也锻炼了我搜索资料的能力，加深了我对文本情感分析的认识。



## 第五章 参考文献

- [1] 胡熠. 面向信息检索的文本内容分析[D].上海交通大学,2007.
- [2] 陈巧云. 社交媒体中基于上下文感知的用户情感分析[D].东南大学,2018.
- [3] 余珍芝. 中文网络产品评论的情感分析关键技术研究[D].杭州电子科技大学,2011.
- [4] 朱琳琳,徐健.网络评论情感分析关键技术及应用研究[J].情报理论与实践,2017,40(01):121-126+131.DOI:10.16353/j.cnki.1000-7490.2017.01.023.
- [5] 文本情感分析国内外研究现状 [EB/OL]  
[http://www.youerw.com/yanjiu/lunwen\\_18830.html](http://www.youerw.com/yanjiu/lunwen_18830.html)
- [6] 王朝辉. 网络商品评论细粒度情感分析系统关键技术研究[D].大连海事大学,2017.
- [7] 黄小江, 万小军, 杨建武等. 汉语比较句识别研究[J]. 中文信息学报, 2008, 22(5):30-38.
- [8] 朱嫣岚, 闵锦, 周雅倩等. 基于 HowNet 的词汇语义倾向计算[J]. 中文信息学报, 2006, 20(1):14-20
- [9] S. R. Das and M. Y. Chen. Yahoo! For Amazon: Sentiment extration from small talk on the web[J].Management Science,2007,pp. 1375-1388.
- [10] A. Andreevskaja and S. Bergler. Mining WordNet for a fuzzy sentiment:Sentiment tag extraction from WordNet glosses[C]. Proceeding of EACL,2006.
- [11] A. Esuli and F. Sebastiani. Determining term subjectivity and term orientation for opinion mining[C]. Proceedings of EACL,2006.
- [12] C.Engstrom. Topic dependence in sentiment classification[D]. Unpublished Mphil dissertation, University of Cambridge,2004.
- [13] P. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews[C]. Proceedings of ACL, 2002, pp.417-424.
- [14] J. Wiebe, R. F. Bruce, and T. P. Ohara. Development and use of a gold standard data set for subjectivity classifications[C]. Proceedings of ACL, 1999, pp.246-253.
- [15] 何野,杨会成,潘玥,徐姝琪.基于改进 CNN 的文本情感分析[J].平顶山学院学报,2021,36(05):59-62.