浙江农林大学

本 科 生 毕 业 设 计（论文）

（2019 届）

信息工程学院

题　　目：在线商品用户评论分析系统设计与实现

学　　号：201505010323

姓　　名：陈怡

专业班级：计算机科学与技术 153 班

指导教师：黄雷君　　职称：讲师

2019 年　　**5** 月　　**8** 日

浙江农林大学

本科生毕业设计（论文）诚信承诺书

我谨在此承诺：本人所写的毕业设计（论文）《在线商品用户评论分析系统设计与实现》均系本人独立完成，没有抄袭行为，凡涉及其他作者的观点和材料，均作了引用注释，如出现抄袭及侵犯他人知识产权的情况，后果由本人承担。

承诺人（签名）：

年　　月　　日

在线商品用户评论分析系统设计与实现

信息工程学院　计算机科学与技术 153 班 陈怡　指导教师：黄雷君

摘要：随着网络的普及，在线购物已经是我们生活中的常态，在网购时我们经常通过他人的商品评价来分析商品的优劣。但是我们人工查看耗时耗力。

本系统期望将爬虫技术和自然语言处理技术结合起来实现对在线商品评论的自动分析。系统通过使用爬虫自动抓取商品的评论数据，并通过情感倾向分析获取每条评论的好评度，通过图表向用户直观地展示商品的评论情况和基本信息。系统当前内置了多种分析方式，包括基于情感字典、基于机器学习以及基于百度 AI 的不同模块，通过配置文件进行更改。以期望使用者拥有良好的使用体验，系统具有操作简便、易扩展等优点。

关键词：网购；爬虫；自然语言处理；情感倾向分析

DESIGN AND IMPLEMENTATION OF ONLINE PRODUCT USER COMMENT ANALYSIS SYSTEM

Abstract: With the popularity of the Internet, online shopping has become the norm in our lives, and in online shopping, we often analyze the benefits of products by evaluating the products of others. But we look manually at the time and effort.

The system expects to combine crawler technology and natural language processing techniques to enable automated analysis of product reviews over the Internet. The system automatically captures the review data for the product using the crawler, obtains the popularity of each comment by analyzing the sentiment trend, and visually displays the review status and basic product information through the schema. The system currently contains a variety of built-in analysis methods, including emotion-based dictionaries, machine-based learning, and various Baidu AI-based modules, which are changed through configuration files. In order to have a good user experience, the system has the advantages of easy operation and easy expansion.

Keywords: Online Shopping,Web Crawler,Natural Language Processing,Sentiment Orientation Analysis;

目　录

1 绪论

1.1 开发背景

这是一个互联网的时代，随着互联网的持续发展，人们与网络的距离日益贴近，其中很为明显的就是网络购物。随着电子商务的急剧增长，网络购物越来越受到欢迎，占据了零售的很大一部分。而类似“双 11”之类的活动的推出更是将网络购物推向了巅峰。现在的生活节奏越来越快，以往的慢节奏已经不大适应现在的人们了，显然网络购物这种快节奏的活动方式更适合现在的人们。

当我们想要购买一个不是立即需要的商品时，我们往往想到的是网络购物，例如淘宝、天猫、京东或者考拉，因为在网络上的商品常常会比实体商铺里的更加便宜也更加方便，我们通过网购平台就可以购买到心仪的商品。据调查显示，网购在购物消费总数中的占比逐年增加。但是同时由于监管不便的原因，网购也存在很多坑点。用户通过商品的评价自我分析商品的优劣，这往往耗时耗力。本系统本着方便买家的立场上进行开发，目的是降低网络购物的风险。系统主要采用爬虫技术和文本情感分析来实现，通过参考系统给出的分析结果来判断商品的优劣。

1.2 设计目的

本课题以设计实现对在线商品[1]的评论进行自动抓取，分析评论的情感倾向[2]，展示分析结果为目的。并以锻炼和检验大学四年的学习成果、程序设计能力、编码开发能力，目标实现一个简单易操作、后续扩展方便、具有一定准确度的在线商品评论分析系统。网络购物时所购买的商品良莠不齐，虽然网络平台提供了商品评论区域，但商品评论过于繁多杂乱，并且有水军混杂其中，对于我们鉴别商品好坏存在扰乱和不便，而且人力查看较慢，一般无法查看所有评论数据，往往只会浏览其中几页甚至几条评论，对于整体的评论信息没有一个更多的把握。本系统旨在爬取所有的评论数据，生成所有评论数据的情感倾向分布图，以及整体评价的分布图，使用直观的图表展示数据，给使用者简单易用的使用体验。

1.3 相关技术

本系统使用 Python 语言进行程序的编写。Web 框架使用 Django，Django 是基于 MTV 模式的的 Web 框架，使用 Python+Django 可以快速的构建应用。评论文本分析中使用了自然语言处理[3]技术和情感倾向分析[4]技术，系统中分别使用了基于情感字典的分析、基于机器学习的分析以及基于百度开放平台的分析。情感字典和机器学习中使用 jieba、snownlp 进行文本的分词，使用 NLTK 进行特征向量的提取，使用 Scikit-learn(sklearn)进行分类器的训练和分类器的使用。

2 可行性研究

可行性研究的目的是用最小的代价在最短的时间内确定问题是否值得解决、是否能够使用现有的技术解决，将从以下三个方面进行本系统的可行性分析。

2.1 技术可行性

系统使用 django 作为基准开发框架。python 的用途广泛，目前分别可以应用于爬虫技术、数据挖掘、数据分析、web 交互等。技术比较成熟，可行性高，并且具有众多第三方开发库。对于爬虫，python 具有强大的 requests 第三方开发库，可以使用较少的语句实现一个简单的 http/https 爬虫请求发送和处理。对于数据分析处理，python 可以使用 jieba 分词进行中文文本的分词。NLTK 是在自然语言处理中最常使用的一个 python 库。SnowNLP 可以进行中文文本内容的分析和挖掘[5]。更进一步，各大 IT 公司具有许多强大的 AI 平台，如百度 AI 开放平台、腾讯 AI 开放平台等开放平台可以使用，其中百度 AI 开放平台具有免费、使用方便等优点，本系统借助百度 AI 开放平台完成一部分的评论分析引擎的编写。系统使用 Mysql 作为数据库引擎，Mysql 具有简单易用，对于个人开发者免费，具有轻量化，性能较强，可移植性高、运行速度较快等优点，且部署方便。系统各方面技术均成熟可用，在技术方面可行。

2.2 经济可行性

本系统开发工具采用 pycharm、vscode、datagrip，其中数据库使用 Mysql，开发语言使用 python，Web 框架使用 django。数据库部署平台为 Linux，Web 应用部署平台为 Windows。以上产品，对于个人为免费产品。开发成本低，经济方面可行。

2.3 操作可行性

本系统设计清晰，具有简单易操作的用户界面，有完善的异常处理机制和错误信息提醒和后台日志记录，用户通过操作界面的提示即可完成操作。通过系统的后台管理界面可以操作系统数据库，方便完成日常的数据维护。操作方面可行。

3 需求分析

3.1 系统需求目标

本次设计目标开发基于 python3、django 的在线商品用户评论分析系统，采用 b/s 模式。以下是本设计要开发完成的五个需求目标：

1. 实用性：设计界面美观实用简洁，软件与用户交互性较强。
2. 稳定性：软件目标是实现用户通过网页界面可实现在线商品的评论分析。采用成熟框架编写，稳定性较高。
3. 友好性原则：操作简单是任何软件的开发的目标，操作简单并不意味着功能简单，将复杂的功能简单化会给用户留下很深的印象。软件在各个阶段给用户以良好的提示。能保证用户的使用体验。
4. 安全性原则：每个使用者无需经过任何的身份验证。各项连接出错要有良好的出错提示。关闭服务端的信息泄露，保证服务器的安全。
5. 可扩展性原则：设计友好的接口，方便系统后续的叠代更新。对业务代码服务化，交互和逻辑分离，为了以后更好拓展，更好维护，重要功能实现独立化，可以脱离整体系统使用。

3.2 系统功能分析

软件要求不断完善，功能也是逐步添加的，以下为该在线商品用户评论分析系统的基本功能需求：

- 用户通过搜索栏输入需要进行分析的商品地址，系统将商品地址返回后台处理。
- 读取商品链接，如果该商品已经进行过评论分析，则读取该商品的分析总结发送给用户，否则通知爬虫引擎进行爬取。
- 爬虫引擎读入商品链接，爬取商品评论和相关信息，存入数据库，等待分析引擎处理。
- 编写多个分析引擎，可以通过配置文件更改分析引擎的使用，并实现单独的接口化。
- 分析引擎读取商品评论，分析评论信息，生成商品分析总结，存入数据库。
- 用户在输入新的商品链接同时可以添加自己的邮箱，在分析完成后，如果该商品有绑定的需要提醒的邮箱，则发送相应邮件提醒用户分析已完成，并附带相关分析结果的链接。
- 具有后台管理页面，使管理员可以通过网页端操作数据库和系统。

4 总体设计

4.1 系统流程图

该软件为在线商品用户评论分析系统，总的系统规划基于传统的 b/s 模式。可以实现多用户同时在线，如下图所示：

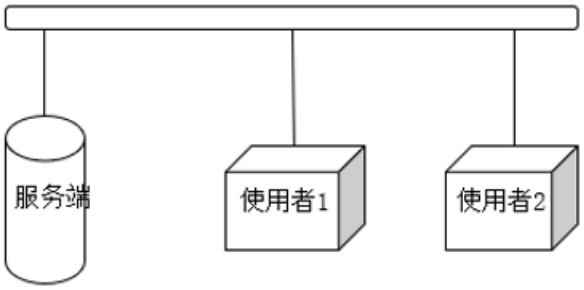

图 4-1 系统模式

系统对于操作成功和失败，异常错误都应存在相应的提示，以便用来帮助用户使用操作系统，同时也可以帮助开发者和管理员更好的开发、维护系统。

系统的流程图如下：

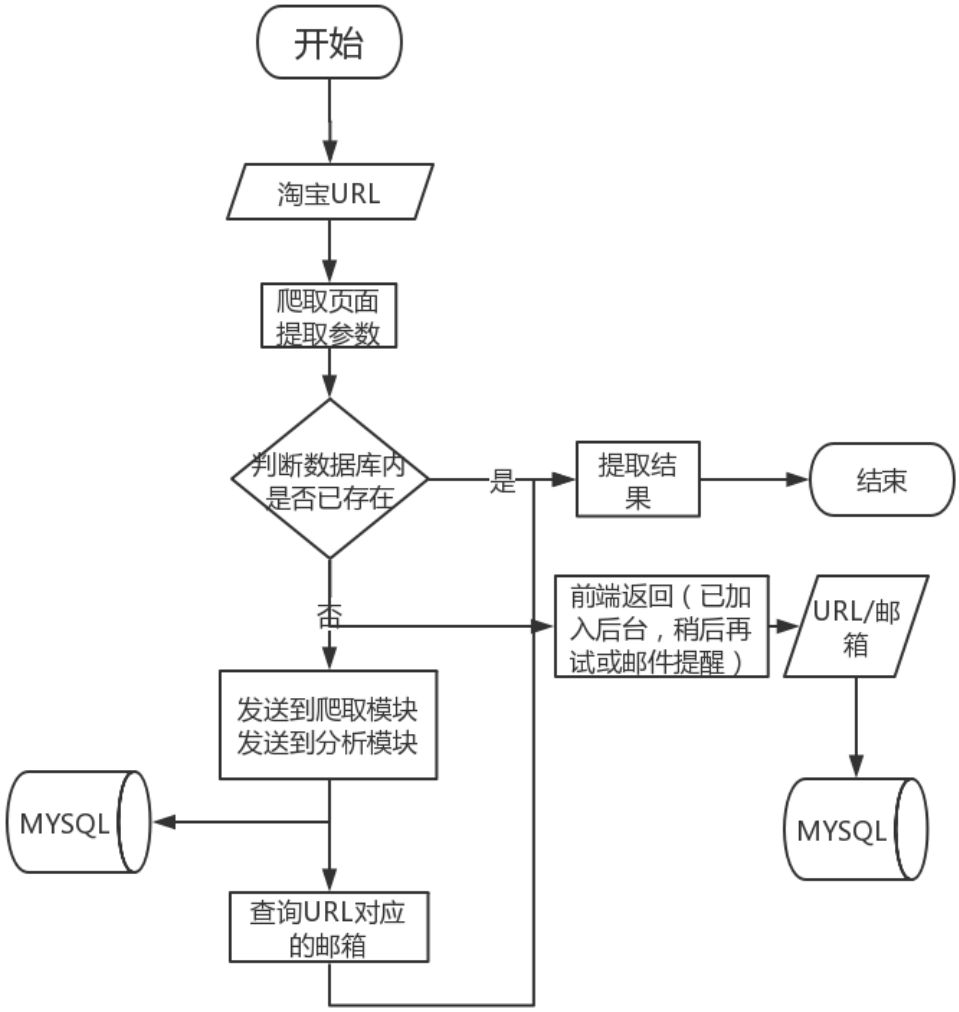

图 4-2 系统流程图

4.2 数据流程图

数据流程图显示了通过系统的数据流的逻辑，以下是本系统的数据流程图：

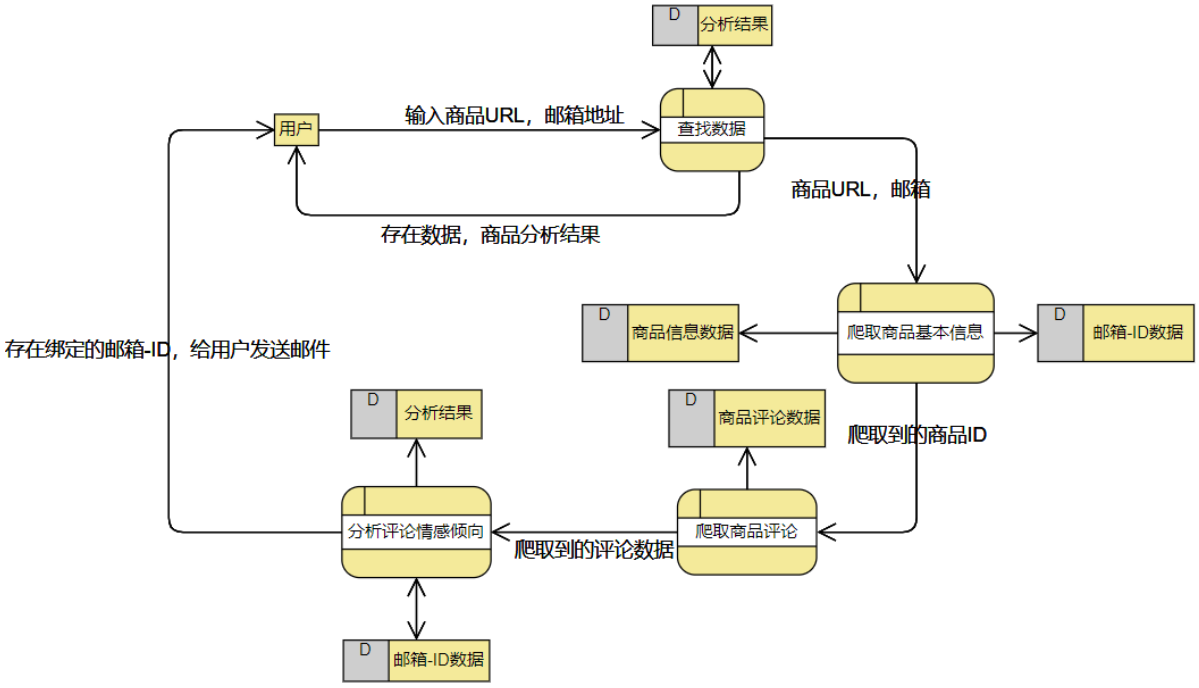

图 4-3 数据流程图

4.3 数据库设计

本系统共设计了商品信息、商品评论、评论分析、邮箱地址，共四个数据库表，以下为数据库的 E-R 图：

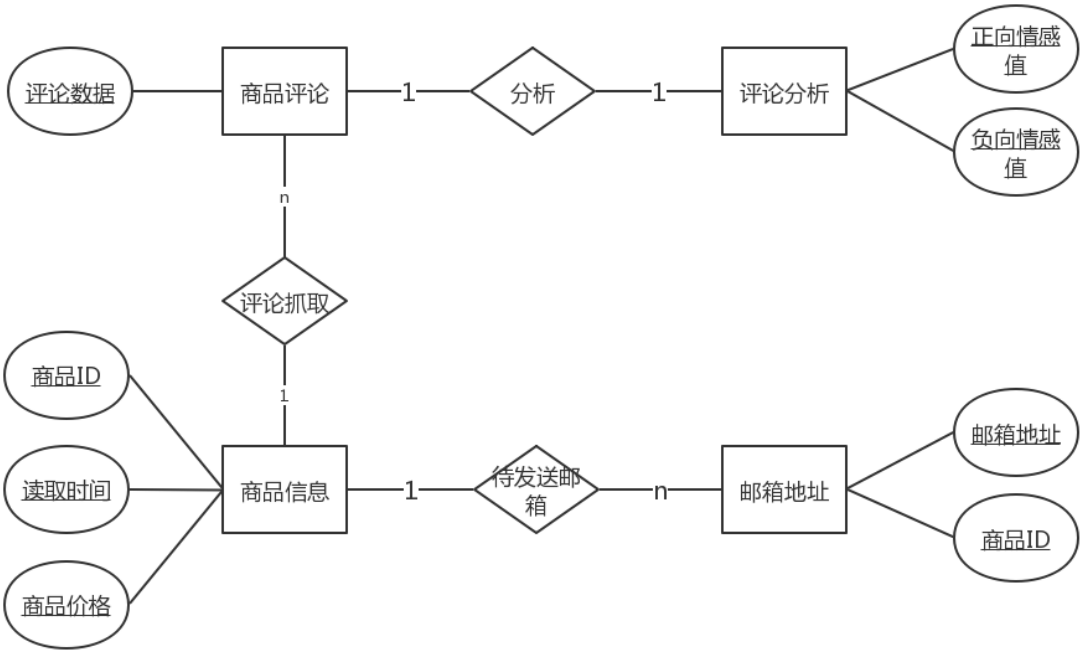

图 4-4 数据库 E-R 图

4.3.1 商品信息表

商品信息表以保存商品的有关信息，包括商品 ID，商品 URL 地址，商品价格，保存时间等字段，表名为：MyModel_taobao，商品信息表的表结构设计如下：

表 4-1 商品信息表

序号	字段名	字段意义	字段类型	是否主键	允许为空	备注
1	id	序号	int	是	是	自动生成
2	taobao_name	商品名称	varchar(100)		否	
3	taobao_id	商品 ID	varchar(50)		否	
4	taobao_url	商品 URl	varchar(512)		否	
5	taobao_shop_name	商品店铺名称	varchar(100)		否	
6	taobao_price_now	商品价格	varchar(50)		否	
7	taobao_time	保存时间	date		否	自动生成

4.3.2 商品评论表

商品评论表以保存商品评论的有关信息，包括商品 ID，商品总体评论，商品详细评价，保存时间等字段，表名为：MyModel_spider，商品评论表的表结构设计如下：

表 4-2 商品评论表

序号	字段名	字段意义	字段类型	是否主键	允许为空	备注
1	id	序号	int	是	是	自动生成
2	spider_id	商品 ID	varchar(50)		否	
3	spider_detail_Common	商品总体评价 JSON	varchar(3000)		否	
4	spider_detail_All	商品详细评价 JSON	longtext		否	
5	spider_time	保存时间	date		否	自动生成

4.3.3 评论分析表

评论分析表以保存评论的分析结果的有关数据，包括商品 ID，商品评价数据的正、负向情感倾向值，保存时间等字段，表名为：MyModel_analyse，评论分析表的表结构设计如下：

表 4-3 评论分析表

序号	字段名	字段意义	字段类型	是否主键	允许为空	备注
1	id	序号	int	是	是	自动生成
2	analyse_id	商品 ID	varchar(50)		否	

续表 4-3

3	analyse_positive_prob	正向情感倾向值 JSON	longtext		否	
4	analyse_negative_prob	负向情感倾向值 JSON	longtext		否	
5	analyse_time	保存时间	date		否	自动生成

4.3.4 邮箱地址表

邮箱地址表以保存待发送的邮箱的有关数据，在评论分析完成后可以通过该表查询是否有需要发送的邮件，该表包括商品 ID，邮箱，保存时间等字段，表名为：MyModel_mail，邮箱地址表的表结构设计如下：

表 4-4 邮箱引用表

序号	字段名	字段意义	字段类型	是否主键	允许为空	备注
1	id	序号	int	是	是	自动生成
2	taobao_id	商品 ID	varchar(50)		否	
3	mail	待发送邮件的邮箱地址	varchar(100)		否	
4	add_time	保存时间	date		否	自动生成

4.4 系统模块介绍

系统主要模块包括管理模块、图表显示模块、爬虫模块、分析模块、交互模块、邮件发送模块这六个主要模块，下图为系统功能模块图：

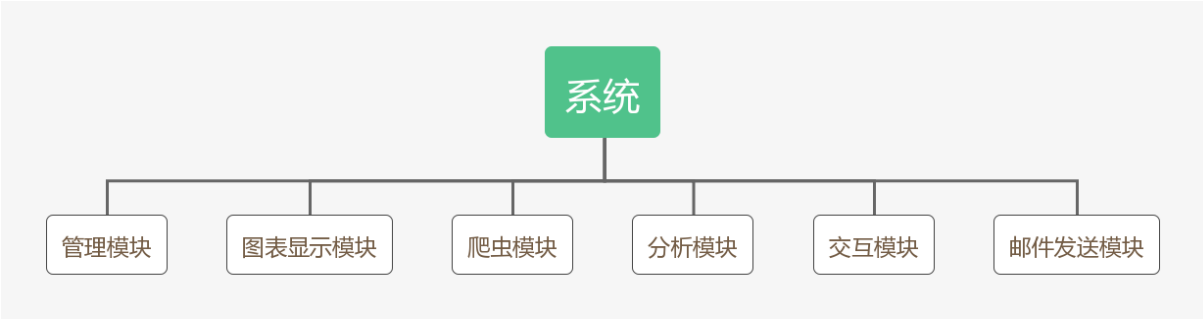

图 4-5 系统模块

4.4.1 管理模块介绍

管理员可以登录管理员界面查看系统数据库信息，进行系统常规设置等，下图为管理模块的设计概要：

表 4-5 管理模块概要

功能编号	4.4.1	功能名称	管理模块
功能描述		管理员通过该模块进行操作数据库	

输入项	操作数据库的数据
处理描述	查询、增加、删除、更新数据库的数据
输出项	数据库详情
界面要求	图形化用户界面

4.4.2 图表显示模块介绍

读取分析结果，生成简单易懂的图表，为用户直观的呈现分析结果，下图为图表显示模块的设计概要：

表 4-6 图表显示模块概要

功能编号	4.4.2	功能名称	图表显示模块
功能描述		商品 ID 对应的分析结果图表显示	
输入项		商品 ID	
处理描述		通过商品 ID 查询商品分析数据表中的分析结果，通过 HTML 页面的图表向用户显示分析结果	
输出项		无有效数据信息提示 分析结果数据、图表	
界面要求		显示有图形化用户界面，通过 url 接口调用	

4.4.3 爬虫模块介绍

将商品链接发送到爬虫引擎，爬取商品链接下的评论，存入数据库等待分析模块运行。适当使用反爬技术，设置合理的爬取速度防止 IP、账户被封，下图为爬虫模块的设计概要：

表 4-7 爬虫模块概要

功能编号	4.4.3	功能名称	爬虫模块
功能描述		商品 ID 对应的评论数据爬取	
输入项		商品 ID	
处理描述		通过请求相关的信息接口，爬取商品评论数据，并进行数据清理和分类	
输出项		商品评论数据	
界面要求		无图形化用户界面，通过函数接口调用	

4.4.4 分析模块介绍

将爬取到的商品评论发送到分析引擎，进行评论处理和分析[6]，去除无用信息，合并相似信息，分析评论信息，最后生成分析结论，下图为分析模块的设计概要：

表 4-8 分析模块概要

功能编号	4.4.4	功能名称	分析模块
功能描述		评论数据情感倾向值分析	
输入项		评论数据	
处理描述		分析评论数据的情感倾向值，分别保存正向情感倾向值和负向情感倾向值	
输出项		情感倾向值数据	
界面要求		无图形化用户界面，通过 url 接口调用	

4.4.5 交互模块介绍

用户通过网页来查看分析的结果，也可以通过邮件接收分析的结果。由用户输入商品链接，传输到服务器，服务器进行评论爬取、分析，并生成分析结果发送给用户。商品链接可一次性输入多条，使用分号（“;”）或回车分割。用户通过填写邮箱，可以接收分析完成的提醒，减少不必要的等待时间，下图为交互模块的设计概要：

表 4-9 交互模块概要

功能编号	4.4.5	功能名称	交互模块
功能描述		用户使用的交互界面	
输入项		商品 URL 地址	
处理描述		接收用户输入的 URL 地址，调用其他模块进行交互，完成评论抓取、分析模块的衔接，存在错误时及时提醒用户	
输出项		开始分析 分析结果 错误提示	
界面要求		图形化用户界面	

4.4.6 邮件发送模介绍

读取设置好的邮箱地址和密码，以此作为发件人，向需要发送提醒的用户发送邮件，提示用户分析已完成，下图为邮件发送模块的设计概要：

表 4-10 邮件发送模块概要

功能编号	4.4.6	功能名称	邮件发送模块
功能描述		发送需要进行提醒的邮件	
输入项		商品 ID	
处理描述		查找邮箱地址数据库表内，该商品 ID 是否有对应需要发送的邮件，如果存在邮件则给用户发送提醒邮件	
输出项		发送成功 发送失败	
界面要求		无图形化用户界面，通过函数接口调用	

5 详细设计

系统主要模块包括管理模块、图表显示模块、爬虫模块、分析模块、交互模块、邮件发送模块，接下来将从这六个主要模块对系统的设计进行描述。

5.1 管理模块

管理员可以登录管理员界面查看系统数据库信息，进行系统常规设置等。

django 预设了后台管理的模板，可以通过 django 的后台管理模板快速地生成一个管理页面。通过激活管理工具和创建超级用户即可使用：

```
# python manage.py createsuperuser
Username (leave blank to use 'root'): admin
Email address: admin@runoob.com
Password:
Password (again):
Superuser created successfully.
[root@solar HelloWorld]#
```

图 5-1 创建管理员账户

在 admin.py 文件中绑定模型，即可方便地进行数据库的管理，同时设置数据库显示的字段，配置条件筛选，字段查找等功能，如下所示，进行模型注册和相关的配置：

```
@admin.register(Taobao)
class TaobaoAdmin(admin.ModelAdmin):
    list_display = ('taobao_id', 'taobao_name', 'taobao_price_now', 'taobao_time',)
    list_filter = ('taobao_time',)
    search_fields = ('taobao_name', 'taobao_id',)  # 搜索字段
    date_hierarchy = 'taobao_time'  # 详细时间分层筛选
```

图 5-2 注册模型

通过登陆账户，可以看到管理模块可以进行管理员账户的认证和授权以及数据库的维护相应的功能：

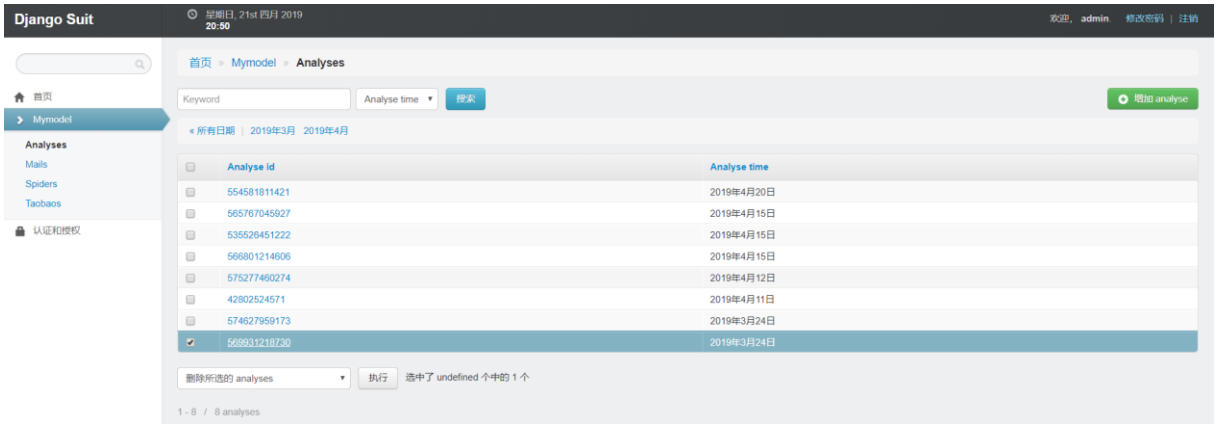

图 5-3 后台页面

5.2 图表显示模块

读取分析结果，生成简单易懂的图表，为用户直观的呈现分析结果。

图表显示模块需要接收 GET 方法传入的商品 ID，查询数据相应的数据，再通过 django 的模板和数据进行绑定，生成相应的图表。

这里通过 Chart.js 的图表显示插件，方便地进行柱状图、饼状图、折线图等的生成。

通过商品 ID 从数据库中查询相应的数据。

```
def get_barjson(taobao_id):
    try:
        analyse = Analyse.objects.get(analyse_id=taobao_id)
        return analyse.analyse_positive_prob
    except Analyse.DoesNotExist:
        print("情感倾向查询不存在")
        return None
```

图 5-4 查询数据

之后进行相应的处理，将数据调整成 Chart.js 所支持的数据格式，并设置到 list 中，方便之后传递到前端。

```
# 准备图表数据
json_bar = get_barjson(taobao_id)
if json_bar == None:
    return HttpResponse("柱状图ID错误")
# 转list，排序
json_bar = re.sub(' ', '', json_bar[1:-1]).split(",")
json_bar.sort(key=None, reverse=False)
# 转str
barstr = ",".join(json_bar)

if barstr == "":...
else:
    print("正向情感数组：" + barstr)
context['barlables'] = context['bardata'] = barstr
```

图 5-5 处理查询的数据

最后将数据返回到前端界面，通过 Chart.js 插件完成相应的图表生成。

```
{#画柱状图#}
var popCanvas = document.getElementById("popChart").getContext("2d");
var popChart = new Chart(popCanvas, {
    type: 'bar',
    data: {
        labels: [{% autoescape off %}{{ newbarlables }}{% endautoescape %}],
        datasets: [{
            fillColor : "rgba(0,0,0,0.5)",
            label: '评论分值排列',
            data: {{ newbardata }},
            backgroundColor: [
                {% autoescape off %}
                    {{ newbarcolor }}
                {% endautoescape %}
            ]
        }]
    },
    options:{
        legend:{display:false},
        title: {
            display: true,
            text: '好评度分布图'
        }
    }
});
```

图 5-6 生成图表

5.3 爬虫模块

将商品链接发送到爬虫引擎，爬取商品链接下的评论，存入数据库等待分析模块运行。适当使用 IP 代理池和反爬技术，设置合理的爬取速度防止 IP、账户被封。

爬虫模块需要将商品的评论和总体评价数据进行爬取，常规做法是通过 scrapy 爬取页面数据，但是这样需要爬取整个淘宝页面的数据，而且对于 AJAX 请求的数据无法很好地进行处理，这里选用分析淘宝商品详情页面载入过程中的请求，选取其中返回评论 JSON 的请求，分析该请求的参数和认证授权方式，通过使用该请求接口来获得淘宝商品的评论。这样的好处是减少代码量，减轻系统的体量，同时也可以提供爬虫模块的运行速度，爬虫极度依赖网络的良好，若是爬取的请求过多，必然会增大系统运行不流畅，而使用特点请求接口可以大大地减少请求数，也可以减轻主机的 CPU 占用和网络消耗。

这里通过使用 burp suite 和火狐浏览器进行配合查看在浏览淘宝商品的过程中，会产生哪些请求，首先要对火狐浏览器设置代理和导入证书，之后开启 burp suite 的监听功能，浏览一个淘宝商品的评论后，再查看 burp suite 抓取到的请求，这里我们可以看到 https://rate.taobao.com/feedRateList.htm 请求是淘宝返回商品评价的 JSONP 请求

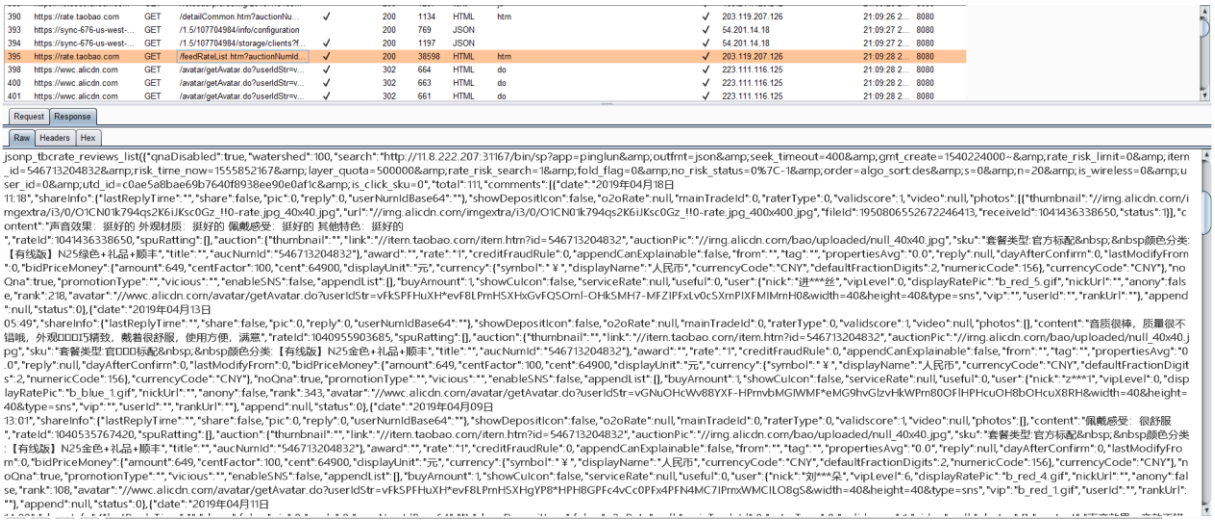

图 5-7 获取 https 请求

通过逐步删除请求中的相关参数，最后分析得到 GET 参数中的 auctionNumId 控制商品的 ID，控制 JSONP 返回的评论数据是哪一个商品的，currentPageNum 控制商品评论的页数，控制返回的评论数据是第几页的数据，pageSize 控制商品评论的页面大小，控制返回的评论数据的多少，Cookie 参数中的 x5sec 控制身份认证，在多次请求后，如果这个参数不存在或为无效参数，将会跳转到验证码机制的页面，但是通过分析发现，这个 x5sec 参数的数值不会过期，长时间或退出登陆任然有效，所以可以通过获取一个 x5sec 参数的数值并使用这个数值长期的获取商品评论的 JSON 数据。

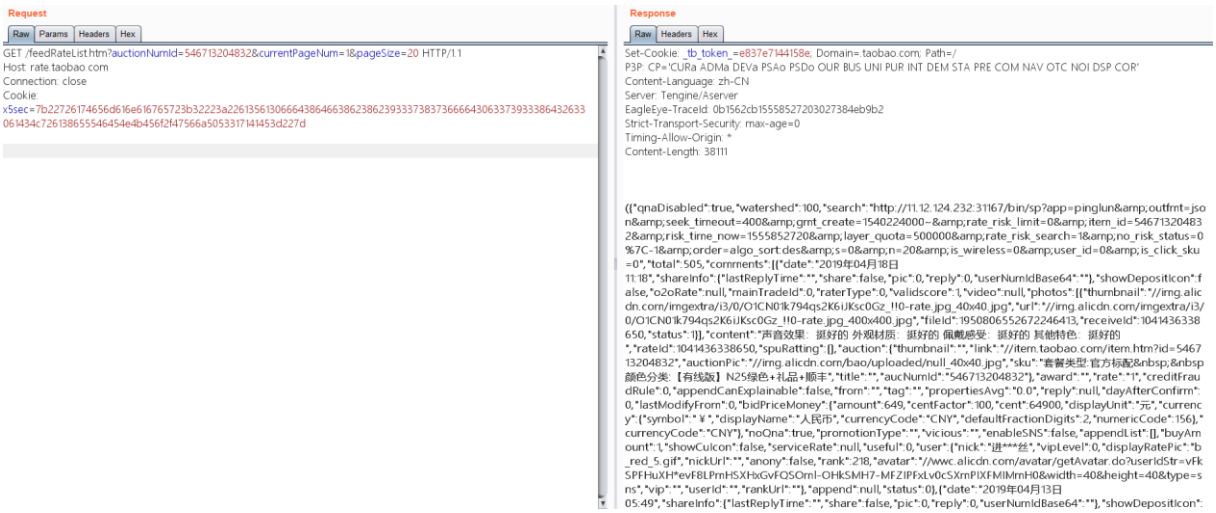

图 5-8 分析请求参数

当爬虫模块被调用并传入商品 ID 时，系统会创建一个新线程并开始评论数据的爬取

```
##########################################多线程
class myThread(threading.Thread):
    def __init__(self, taobao_id):
        threading.Thread.__init__(self)
        self.taobao_id = taobao_id

    def run(self):
        print("开启爬虫线程：  " + self.name)
        # 获取锁，用于线程同步
        threadLock.acquire()
        save_common_mysql(self.taobao_id)
        # 释放锁，开启下一个线程start_thread
        threadLock.release()


threadLock = threading.Lock()
threads = []


def start_thread_spider(taobao_id):
    # 创建新线程
    thread = myThread(taobao_id)

    # 开启新线程
    thread.start()
    threads.append(thread)
```

图 5-9 开启爬虫线程

通过请求之前分析获得请求接口，一页一页地发送请求，并将获得的返回 JSON 的数据进行清洗，整合，并存入数据库，在数据爬取完成后，通过调用分析模块的接口进行分析文本的情感倾向值。

```
def One_Page_Spider(taobao_id, page_num, page_size="1000"):  # 获取单页评价
    parmar = {'auctionNumId': taobao_id, 'currentPageNum': page_num, 'pageSize': page_size}
    cookies = dict(
        x5sec='7b22726174656d616e616765723b32223a22363130623039345567663966303636636363623562316
    taobao_one_page_url = r"https://rate.taobao.com/feedRateList.htm"
    r = requests.get(taobao_one_page_url, params=parmar, cookies=cookies)

    spider_one_page_common = r.text
    r.close()

    return (spider_one_page_common)
def All_Spider(taobao_id):  # 获取所有评论
    list_content = []

    for page_num in range(1, 200):
        content = One_Page_Spider(taobao_id, page_num)
        all_content = re.findall(r"\],\"content\":\"(.+?)\"", content)
        if len(all_content) != 0:
            list_content = list_content + all_content
        else:
            break
    #加入分析
    start_thread_analyse(list_content,taobao_id)
    #
    return (json.dumps(list_content))
    # return (list_content)
```

图 5-10 爬取评论数据

5.4 分析模块

将爬取到的商品评论发送到分析引擎，进行评论处理和分析，去除无用信息，合并相似信息，分析评论信息，最后生成分析结论。

分析模块需要将爬虫模块获取到的商品评论进行分析，这里设置了三种分析方式，并可以独立使用。

自然语言处理的一个研究方向是分析文本的情感倾向值，分析一个文本情感倾向值，可以知道这个文本的情感是善意的还是恶意的，正向的情感倾向值表示文本的观点是积极的、友好的，负向的情感倾向值表示文本的观点是悲观的、恶意的。在在线商品用户评论分析系统中，若是一个评论的情感倾向值是正向的，则代表这个评论是一个好评，这个值越大代表好评度越高，若是一个评论的情感倾向值是负向的，则代表这个评论是一个差评，这个值越大代表差评度越高。目前的情感倾向值分析主流使用两种分析方式，一种是基于情感字典的分析方式，这种分析方式依赖于情感字典，情感字典越丰富，准确率越高，优点是方便移植，情感字典较为通用，缺点是一个好的情感字典是需要一个长时间的构建，较为耗时耗力。另一种较为流行的分析方式是基于机器学习，通过大量的训练集来构建分类器，并通过一定数量的测试集进行校准和筛选不同的分类方法，优点是准确度较高，缺点时依赖分类器，一个领域构建的分类器给另一个领域必然是准确度大大降低的，甚至同一个领域的不同分支构建的分类器也会有很大的差别。

这个模块设置了三种不同的分析模式，可以通过配置文件进行更改，分别为基于机器学习的分析模式、基于情感字典的分析模式和基于百度 AI 的分析模式，接下来将分别介绍这三种模式的详细设计：

5.4.1 基于机器学习的分析模式

机器学习是当前进行情感倾向分析[7]的有力途径，通过一定数量的训练数据对分类器进行训练校准，可以得到一个较为准确的分类模型[8]。

使用机器学习进行分析的算法框架为：

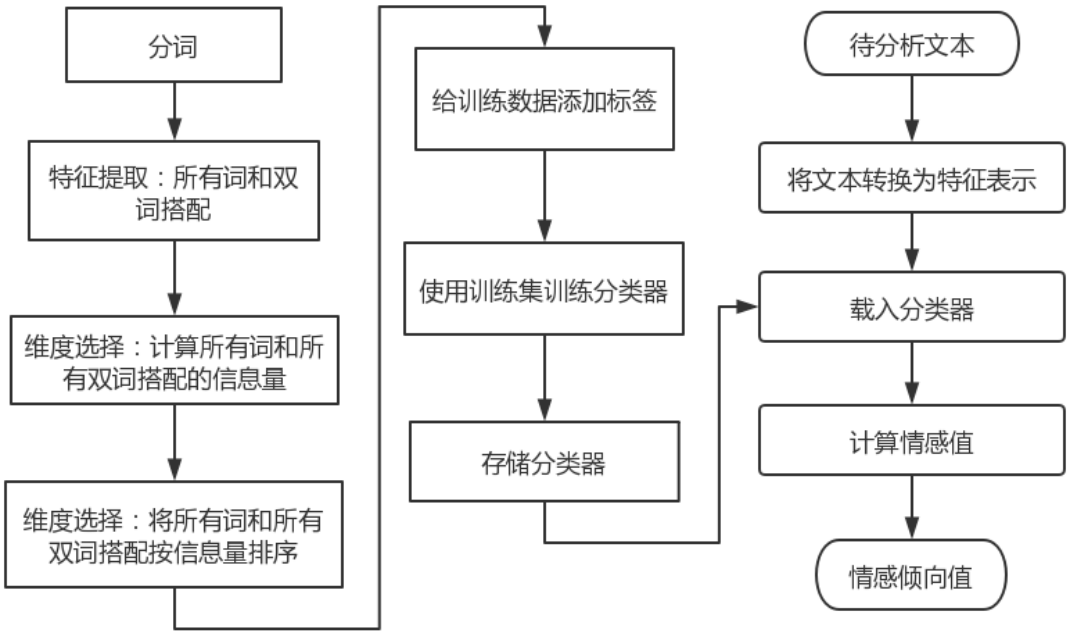

图 5-11 机器学习算法框架

使用机器学习进行分析的基本流程为：

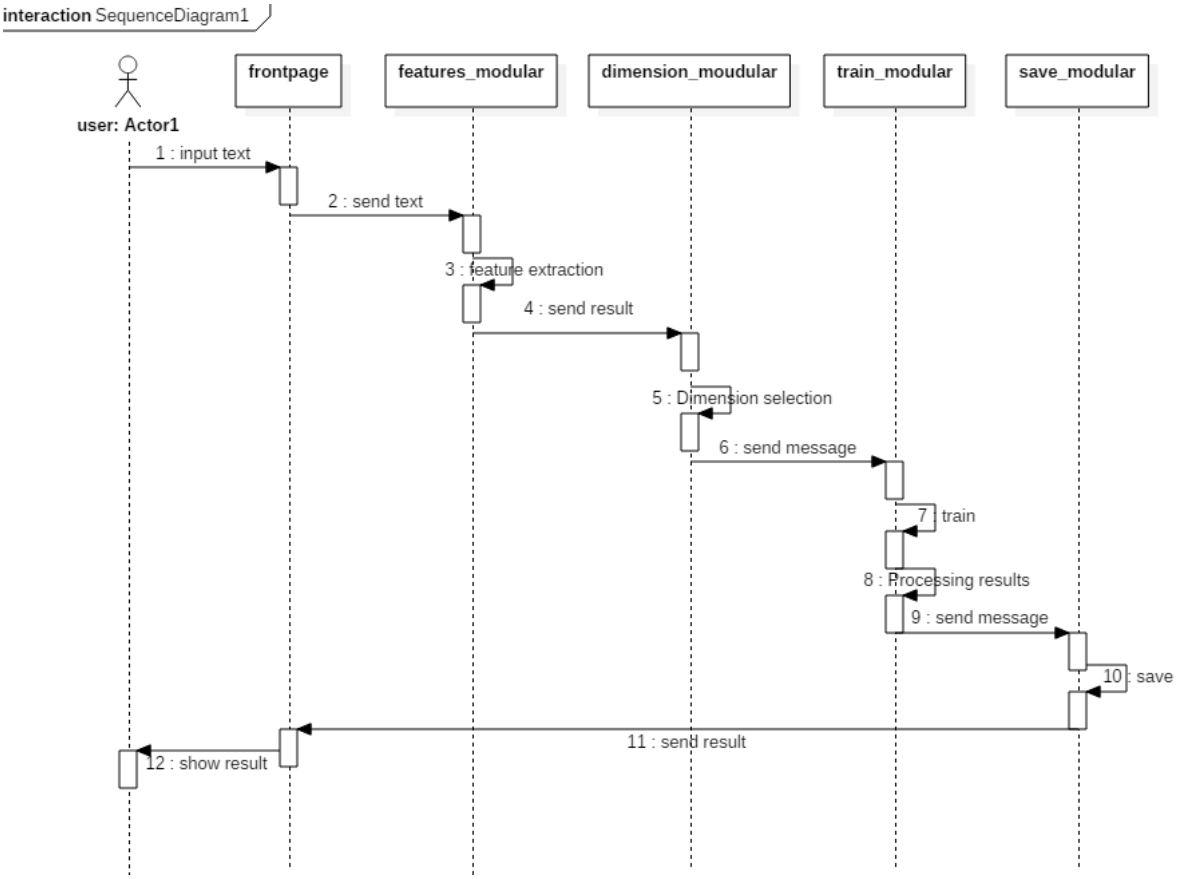

图 5-12 机器学习流程

1、 提取特征：

将文本提取为特征形式表示

把所有词作为特征进行提取[9]

```
def bag_of_words(words):
    return dict([(word, True) for word in words])
```

图 5-13 所有词特征提取

把双词作为特征进行提取

```
# 把双词搭配（bigrams）作为特征
def bigram(words, score_fn=BigramAssocMeasures.chi_sq, n=1000):

    bigram_finder = BigramCollocationFinder.from_words(words)  # 把文本变成双词搭配的形式
    bigrams = bigram_finder.nbest(score_fn, n)  # 使用了卡方统计的方法，选择排名前1000的双词

    return bag_of_words(bigrams)
```

图 5-14 双词特征提取

把所有词搭配双词作为特征提取

```
# 把所有词和双词搭配一起作为特征
def bigram_words(words, score_fn=BigramAssocMeasures.chi_sq, n=1000):

    tuple_words = []
    for i in words:
        temp = (i,)
        tuple_words.append(temp)

    bigram_finder = BigramCollocationFinder.from_words(words)
    bigrams = bigram_finder.nbest(score_fn, n)  # 使用了卡方统计的方法，选择排名前1000的双

    return bag_of_words(tuple_words + bigrams)  # 所有词和双词搭配一起作为特征
```

图 5-15 搭配特征提取

2、 选择特征

选择信息量较大的特性，抛除对分类训练影响较大的特性

计算每个词的信息量

```
def create_word_scores():
    posWords = pickle.load(open(pos_f, 'rb'))
    negWords = pickle.load(open(neg_f, 'rb'))

    posWords = list(itertools.chain(*posWords))  # 把多维数组解链成一维数组
    negWords = list(itertools.chain(*negWords))

    word_fd = FreqDist()  # 可统计所有词的词频
    cond_word_fd = ConditionalFreqDist()  # 可统计积极文本中的词频和消极文本中的词
    for word in posWords:
        word_fd[word] += 1
        cond_word_fd["pos"][word] += 1
    for word in negWords:
        word_fd[word] += 1
        cond_word_fd["neg"][word] += 1

    pos_word_count = cond_word_fd['pos'].N()  # 积极词的数量
    neg_word_count = cond_word_fd['neg'].N()  # 消极词的数量
    total_word_count = pos_word_count + neg_word_count

    word_scores = {}
    for word, freq in word_fd.items():
        pos_score = BigramAssocMeasures.chi_sq(cond_word_fd['pos'][word], (freq, pos_word_count),
                                               total_word_count)  # 计算积极词的卡方统计量
        neg_score = BigramAssocMeasures.chi_sq(cond_word_fd['neg'][word], (freq, neg_word_count),
                                               total_word_count)
        word_scores[word] = pos_score + neg_score  # 一个词的信息量等于积极卡方统计量加上消极卡方统计量

    return word_scores  # 包括了每个词和这个词的信息量
```

图 5-16 计算每个词信息量

计算每个词和双词搭配的信息量

```
# 计算整个语料里面每个词和双词搭配的信息量
def create_word_bigram_scores():
    posdata = pickle.load(open(pos_f, 'rb'))
    negdata = pickle.load(open(neg_f, 'rb'))

    posWords = list(itertools.chain(*posdata))
    negWords = list(itertools.chain(*negdata))

    bigram_finder = BigramCollocationFinder.from_words(posWords)
    posBigrams = bigram_finder.nbest(BigramAssocMeasures.chi_sq, 5000)
    bigram_finder = BigramCollocationFinder.from_words(negWords)
    negBigrams = bigram_finder.nbest(BigramAssocMeasures.chi_sq, 5000)

    pos = posWords + posBigrams  # 词和双词搭配
    neg = negWords + negBigrams

    word_fd = FreqDist()
    cond_word_fd = ConditionalFreqDist()
    for word in pos:
        word_fd[word] += 1
        cond_word_fd["pos"][word] += 1
    for word in neg:
        word_fd[word] += 1
        cond_word_fd["neg"][word] += 1

    pos_word_count = cond_word_fd['pos'].N()
    neg_word_count = cond_word_fd['neg'].N()
    total_word_count = pos_word_count + neg_word_count

    word_scores = {}
    for word, freq in word_fd.items():
        pos_score = BigramAssocMeasures.chi_sq(cond_word_fd['pos'][word], (
            freq, pos_word_count), total_word_count)  # 计算积极词的卡方统计量，这里
        neg_score = BigramAssocMeasures.chi_sq(
            cond_word_fd['neg'][word], (freq, neg_word_count), total_word_count)
        word_scores[word] = pos_score + neg_score

    return word_scores
```

图 5-17 计算搭配的信息量

对信息量进行排序，选出信息量较大的词

```
# 根据信息量进行倒序排序，选择排名靠前的信息量的词
def find_best_words(word_scores, number):
    best_vals = sorted(word_scores.items(), key=lambda w_s: w_s[1], reverse=True)[
        :number]  # 把词按信息量倒序排序。
    best_words = set([w for w, s in best_vals])
    return best_words
```

图 5-18 选择特征值

经过以上步骤完成对文本的特性值的提取[10]

3、 构建分类器，检验分类的准确度，并保存分类器

载入数据

```
def load_data():
    global pos_review, neg_review
    pos_review = pickle.load(open(pos_f, 'rb'))
    neg_review = pickle.load(open(neg_f, 'rb'))
```

图 5-19 载入训练集

赋予类标签

```
# 积极
def pos_features(feature_extraction_method):
    posFeatures = []
    for i in pos_review:
        posWords = [feature_extraction_method(i), 'pos']  # 为积极文本赋予"pos"
        posFeatures.append(posWords)
    return posFeatures


# 消极
def neg_features(feature_extraction_method):

    negFeatures = []
    for j in neg_review:
        negWords = [feature_extraction_method(j), 'neg']  # 为消极文本赋予"neg"
        negFeatures.append(negWords)
    return negFeatures
```

图 5-20 赋类标签

将测试集分割成训练集和开发测试集，并且分割标注的数据和标签

```
# 把特征化之后的数据数据分割为开发集和测试集
def cut_data(posFeatures, negFeatures):
    global train, devtest, test
    train = posFeatures[1500:] + negFeatures[1500:]
    devtest = posFeatures[:500] + negFeatures[:500]


# 开发测试集分割人工标注的标签和数据
def cut_devtest():
    global dev, tag_dev
    dev, tag_dev = zip(*devtest)
```

图 5-21 分割数据

训练分类器，并使用开发测试集校验准确度

```
def score(classifier):
    classifier = nltk.SklearnClassifier(classifier)  # 在nltk 中使用scikit-learn的接口
    classifier.train(train)  # 训练分类器

    pred = classifier.classify_many(dev)  # 对开发测试集的数据进行分类，给出预测的标签
    return accuracy_score(tag_dev, pred)  # 对比分类预测结果和人工标注的正确结果，给出分类器准确度


def try_diffirent_classifiers():

    results = list()
    results.append(score(BernoulliNB()))
    results.append(score(MultinomialNB()))
    results.append(score(LogisticRegression()))
    results.append(score(SVC()))
    results.append(score(LinearSVC()))
    results.append(score(NuSVC()))

    return results
```

图 5-22 计算正确率

选择其中准确率较高的分类方法，这里选择了朴素贝叶斯分类器[11]以及词和双词搭配的特征提取方式，并存储分类器，将训练好的分类器保存到文件中

```
def store_classifier():
    load_data()
    word_scores = create_word_bigram_scores()
    global best_words
    best_words = find_best_words(word_scores, 7500)

    posFeatures = pos_features(best_word_features)
    negFeatures = neg_features(best_word_features)

    trainSet = posFeatures + negFeatures

    MultinomialNB_classifier = SklearnClassifier(MultinomialNB())
    MultinomialNB_classifier.train(trainSet)
    pickle.dump(MultinomialNB_classifier, open('../out/classifier.pkl', 'wb'))
```

图 5-23 保存分类器

4、 最后使用训练好的分类器进行文本的情感分析

读入 URL 传递的文本数据，清理读入的数据，删除不必要的特殊符号

```
text = request.GET['query']
type = request.GET['type']

if type == 'MyNLP':
    text = ''.join(text.split())
    text = re.sub("[\s+\.\!\/_,$%^*(+\"\']+|[+——！，。？、~@#￥%……&*（）～-]+", "", text)
    pos_list = jieba.cut(text, cut_all=False)
    res = CEA.application(CEA.transfer_text_to_moto(list(pos_list)))
    print("基于机器学习：" + str(res))

    return HttpResponse(res)
```

图 5-24 清理文本

将文本转换为特征表示

```
def transfer_text_to_moto(data):
    best_words = pickle.load(open('Emotion_Manager/CEA_LIB/best_words.pkl', 'rb'))
    moto_features = best_word_features(data, best_words)
    return moto_features
```

图 5-25 特征转换

载入分类器计算情感倾向的概率值

```
def application(moto_features):
    global result
    clf = pickle.load(open('Emotion_Manager/CEA_LIB/classifier.pkl', 'rb'))  # 载入分类器
    pred = clf.prob_classify_many(moto_features)  # 计算概率值
    for i in pred:
        result =i.prob('pos')
    return round(result * 100, 2)
```

图 5-26 计算情感值

5.4.2 基于情感字典的分析模式

情感字典是另一种较为常用的分析模式，在不同领域经常使用差不多相近情感字典，较为通用但准确率较低。

使用情感字典进行分析的算法框架为：

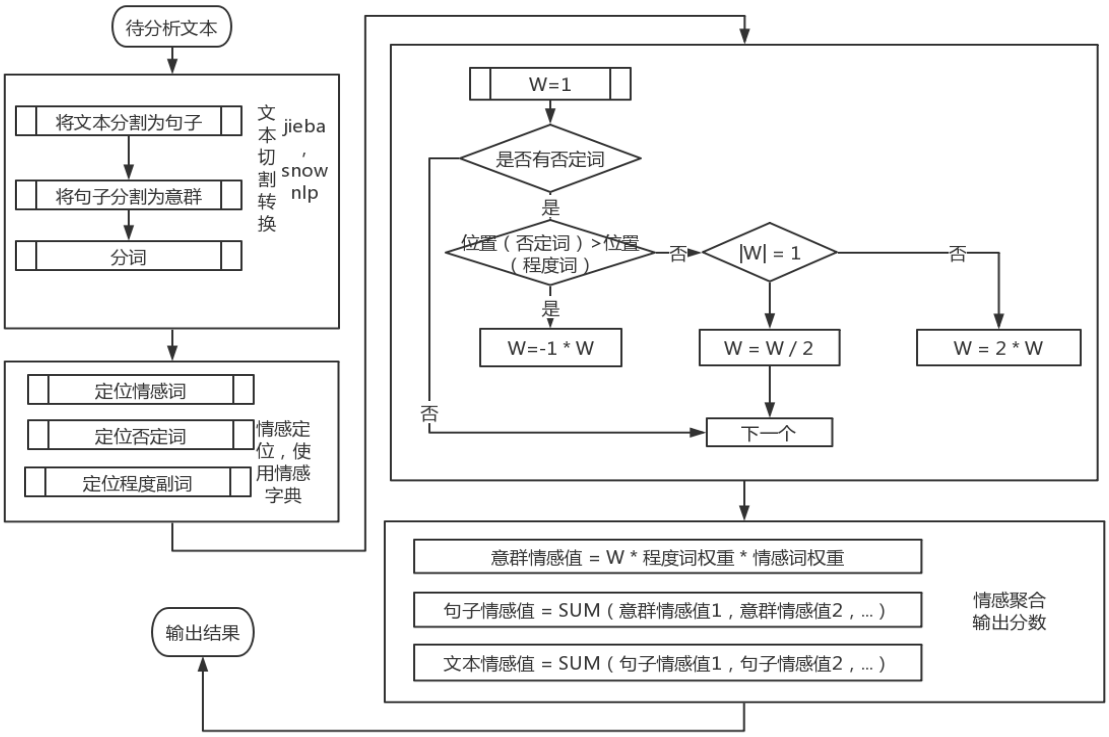

图 5-27 情感字典算法框架

使用情感字典进行分析的基本流程为：

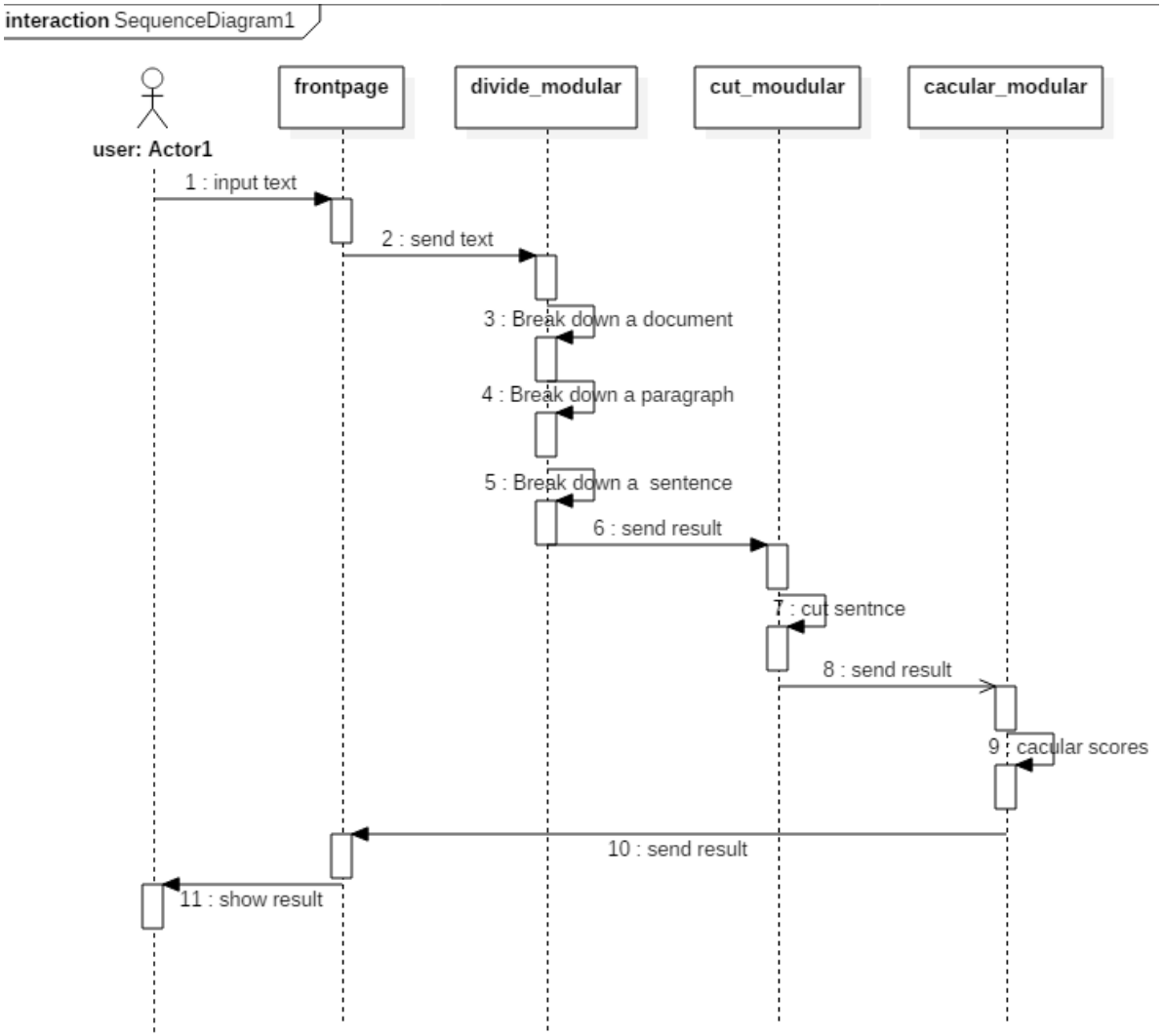

图 5-28 情感字典流程

1、 加载情感字典：

```
def __init_dic__(dic_kind):
    global ext_dic
    jieba.load_userdict(dic_root_path + 'create_by_huzehao/jieba_dic.txt')
    __path__ = dic_root_path + "zhiwang/"
    ext_dic = Emotion_Manager.Modules.load_dic.load_ext_dic(__path__, "extent_Lv_")  # 初始化程度副词词典
    __init__no_word_list()
    if dic_kind == 1:               #知网
        __init_zhiwang_dic__()
    elif dic_kind == 2:             #大连理工
        __init_dllg_dic__()
    elif dic_kind == 3:             #NTUSD
        __init_ntusd_dic__()
    elif dic_kind == 4:             #清华大学 李建军
        __init_tsinghua_dic__()
    elif dic_kind == 5:             #情感极值词典
        __init_extreme_dic__()
    else:
        return None
```

图 5-29 载入情感字典

2、 得出词语详细信息：

```
def find_word_info(word, kind, dic_kind, All = True):
    dic = {}
    def __setdic__(k, s, p = None):
        dic['n'] = word                    #word
        dic['k'] = k                       #kind
        dic['s'] = s                       #score
        dic['p'] = p                       #property

    def __common__(pos_dic, neg_dic):      #如果词语是正面情
        score = __getScore__(pos_dic, word)
        if score != 0:
            __setdic__(kind, score, 'pos')
        else:...

    if word in no_word_set:
        __setdic__('no', None, None)
    elif  kind in sense_word_kind_set:...

    if len(dic) > 0:
        return dic
    elif All:...
    else: return None
```

图 5-30 获取词语信息

3、计算短句子情感值：

```
def _get_group_score(tiny_sentence,group = [{}], stream = None):
    if len(group) > 0:
        stack = []
        score = None
        score_item = None
        pair = getCommentPair(tiny_sentence,group)
        if pair != None:
            score_item = conn.execute('select s from polysemy where a = ? and n = ?', pair).fetchone()
            if score_item != None:
                score = score_item[0]
                stack.append(score)
                for item in group:
                    if item.get('k') == 'no':
                        stack.append(-1)
                    elif item.get('k') == 'ext':
                        stack.append(item.get('s'))
                return  __CaculateScoreOfGroup__(stack, False), stack, pair
        #else:
        score, stack = get_group_score(group)
        return score, stack, pair
    return 0, None
```

图 5-31 计算句子情感值

4、获取整个中文文本的情感值：

```
def getScoreFromString(text, dic_kind):
    __init_dic__(dic_kind)
    pgen = get_paragraph(text)
    ggen = get_group(pgen)
    _score_sum_ = 0
    if dic_kind == 4:
        _score_sum_ = _text_processing_(text)
    else:
        for group in ggen:
            wordList = splict_group_into_list(group, dic_kind)
            score, stack = get_group_score(wordList)
            _score_sum_ += score
            #print(group, score, stack, wordList)
    return _score_sum_
```

图 5-32 计算文本情感值

5.4.3 基于百度 AI 的分析模式

百度 AI 开放平台提供了文本情感倾向分析的相关接口，通过使用开放平台可以快速地构建系统，缩短情感倾向分析模块的开发时间。在百度 AI 平台获得相关授权后，通过 API 文档的说明进行构建情感倾向分析模块。

以下是通过百度 AI 获取中文文本情感分析的流程图。

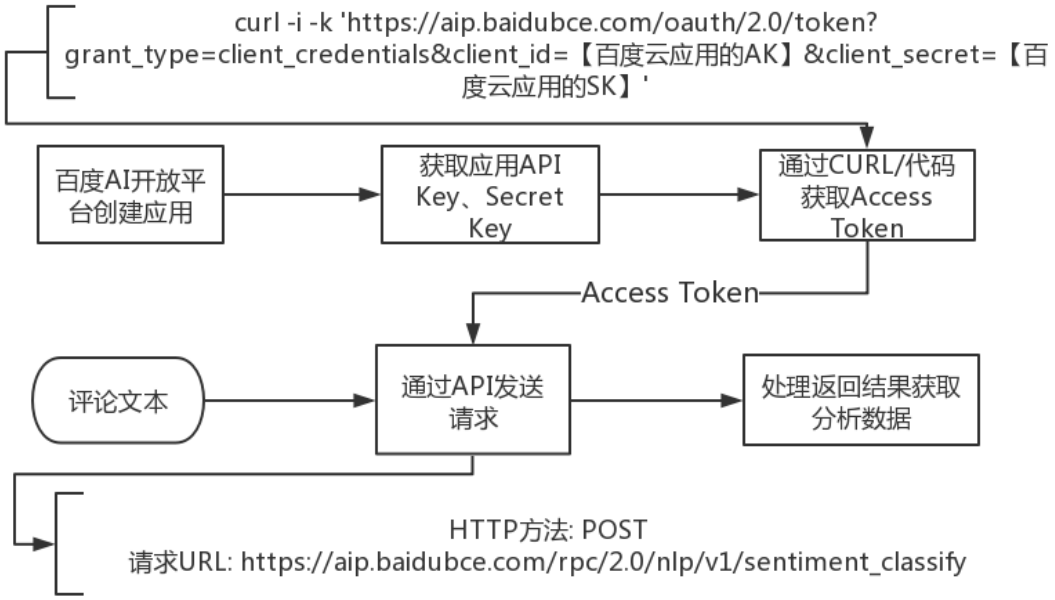

图 5-33 百度 AI 流程图

以下是通过百度 API 获取文本分析结果的代码。

```python
def sentiment_classify(text):
    """
    获取文本的感情偏向（消极 or 积极 or 中立）
    参数:
    text:str 本文
    """
    # 读config.ini文件
    dir_now = os.path.dirname(__file__)
    conf = configparser.ConfigParser()
    conf.read(dir_now + '/config.ini')
    raw = {"text": "内容"}
    raw['text'] = text
    data = json.dumps(raw).encode('utf-8')
    AT = conf.get('baidu', 'AT')
    host = conf.get('baidu', 'host') + AT
    request = urllib.request.Request(url=host, data=data)
    request.add_header('Content-Type', 'application/json')
    response = urllib.request.urlopen(request)
    content = response.read().decode('utf-8')
    rdata = json.loads(content)
    # print("百度AI返回" + str(rdata))
    return rdata
```

图 5-34 获取文本情感值

目前百度 AI 开放平台还支持定制训练集数据的情感倾向分析，可以使分析获得的数据更加准确。

5.5 交互模块

用户通过网页来查看分析的结果，也可以通过邮件接收分析的结果。由用户输入商品链接，传输到服务器，服务器进行评论爬取、分析，并生成分析结果发送给用户。商品链接可一次性输入多条，使用分号（“;”）或回车分割。用户通过填写邮箱获取分析结果完成的提醒，减少不必要的等待时间。

交互模块需要将用户传入的商品 URL 地址进行接收，并通过调度爬虫模块和分析模块进行数据的爬取和分析。

以下为交互模块的流程图：

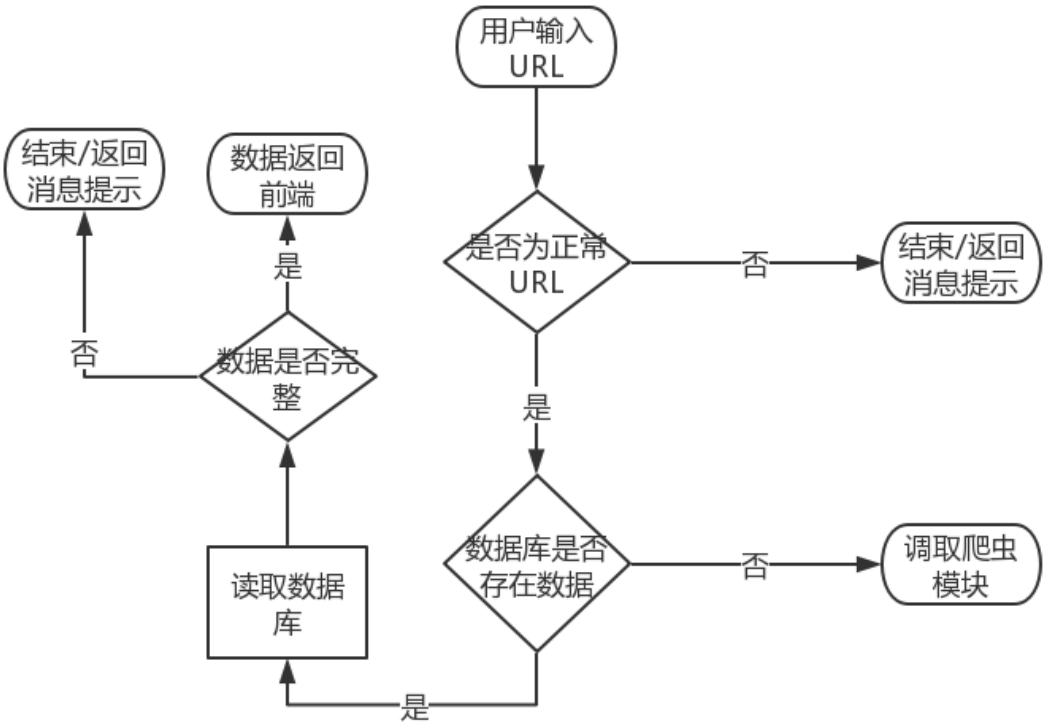

图 5-35 交互模块流程图

为了用户良好的感官体验，这里运用 AJAX 技术进行数据的传递，用户可以传入多个 URL 地址，并通过回车、逗号或者分号进行分隔，用户在输入 URL 地址时也可以同时输入自己的邮箱地址，前端会先进行一个校验，排除错误的、不规范的 URL 地址和邮箱地址

```
function ajax_post(url, email) {
    htmlobj = $.ajax(
        {
            url:"get_taobao_url/",
            timeout:1000,
            data:{csrfmiddlewaretoken:'{{ csrf_token }}',
            taobao_url:url,
            email:email},
            type:"post",
            success:function(result) {
                console.log("发送成功！");
                console.log(result);

                {#$("#request_id").html(result);#}
                var patrn = /^(-)?\d+(\.\d+)?$/;
                if (patrn.exec(result) == null || result == "") {
                    alert(result)
                    return true
                } else {
                    $("ol").append("<iframe src=\"analyse/?taobao_id="+result+"\" frameborder=\"0\" s
                    console.log("生产iframe成功！");
                    return true
                }
            }
        }
    )}
```

图 5-36 发送 AJAX 请求

后端接收到数据后，会再进行一次校验，排除有人绕过前端和校验或者网络出现问题的情况，通过查询 URL 地址所对应的数据，若该地址对应的商品 ID 存在相应的数据，已经进行过评论的爬取和分析，将会返回商品 ID，由前端再向图表显示模块发送获取分析结果的请求。若是不存在相应数据，将会向调用爬虫模块进行相关评论数据的爬取，并向前端返回数据开始爬取分析的提示消息。若是同时前端返回了邮箱地址，则会在开始爬取时将商品 ID 和邮箱地址存入相应的数据库，以便在分析结束后向用户发送提醒邮件

```
if_new_taobao_id = Taobao.objects.filter(taobao_id=taobao_id[0])
if if_new_taobao_id:  # 数据库中存在相应信息，直接返回

    # 取结果
    ctx = taobao_id[0]
    return HttpResponse(ctx)

    start_thread_spider(taobao_id[0])  # 防止意外存入taobao未存入spider
    return HttpResponse(ctx)
else:  # 不存在，加入数据库，加入爬取池
    taobao_add = Taobao()
    taobao_add.taobao_price_now = taobao_price_now[0]
    taobao_add.taobao_shop_name = taobao_shop_name[0]
    taobao_add.taobao_url = taobao_url
    taobao_add.taobao_id = taobao_id[0]
    taobao_add.taobao_name = taobao_name[0]
    taobao_add.save()

    # 加入线程池
    start_thread_spider(taobao_id[0])
    # 加入邮箱
    email_result = save_email(taobao_id[0], email)
    if email_result:
        ctx = "未找到结果，等待分析，请及时查看邮箱"
    else:
        ctx = "未找到结果，等待分析，请稍后再试，若想及时得到信息，请勾选并输入邮箱"

    return HttpResponse(ctx)
```

图 5-37 处理请求参数

5.6 邮件发送模块

读取设置好的邮箱地址和密码，以此作为发件人，向需要发送提醒的用户发送邮件进行提醒，告知用户评论分析已完成。

以下为邮件发送模块的流程图：

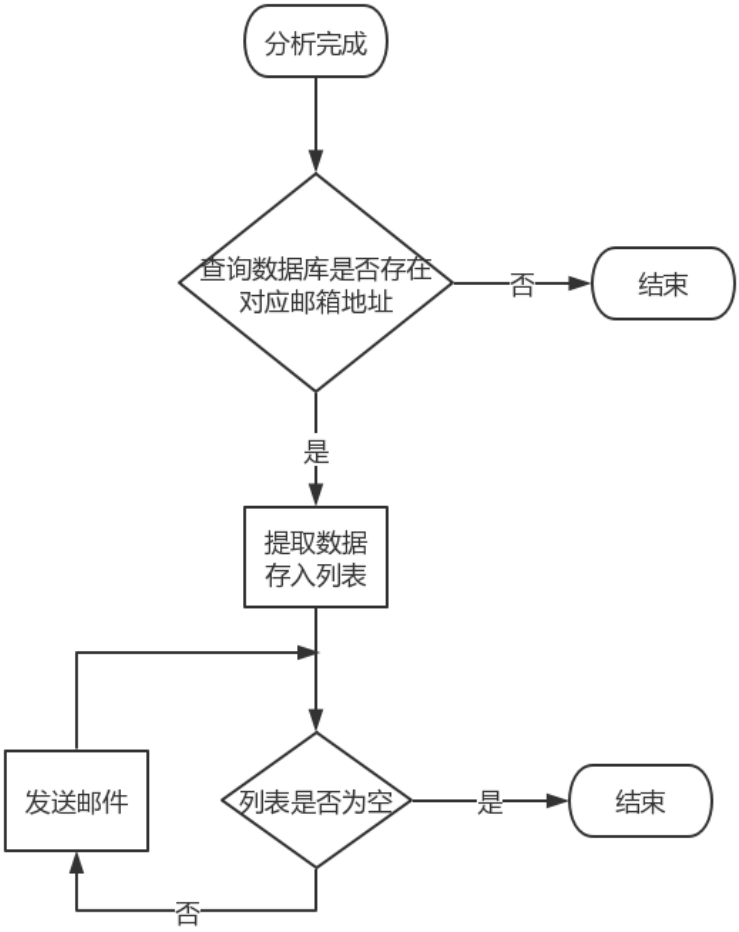

图 5-38 邮件发送模块流程图

邮箱发送模块需要在当评论数据分析完成后调用时，通过传递的商品 ID 查询邮箱地址数据表，查看是否存在相应需要发送提醒邮件的邮箱地址，若不存在，直接返回。若存在，则首先读取需要发送邮件的邮箱地址并使用传递的商品 ID 产生拼接后的可以查看分析结果的 URL 地址，地址经过邮件发送给用户，在提醒用户分析完成的情况下同时发送相应的 URL 地址，方便用户的查看。这里使用 python 内置的 mail 库进行邮件的发送。

```python
def mail():
    ret = True
    try:
        text = "您的分析已完成，请点击链接以查看结果：http://127.0.0.1:8000/analyse/?taobao_id=" + taobao_id
        msg = MIMEText(text, 'plain', 'utf-8')
        msg['From'] = formataddr(["陈怡", my_sender])  # 括号里的对应发件人邮箱昵称、发件人邮箱账号
        msg['To'] = formataddr(["everyone", my_user])  # 括号里的对应收件人邮箱昵称、收件人邮箱账号
        msg['Subject'] = "分析结果-提醒"  # 邮件的主题，也可以说是标题

        server = smtplib.SMTP_SSL("smtp.qq.com", 465)  # 发件人邮箱中的SMTP服务器，端口是465
        server.login(my_sender, my_pass)  # 括号中对应的是发件人邮箱账号、邮箱密码
        server.sendmail(my_sender, [my_user, ], msg.as_string())  # 括号中对应的是发件人邮箱账号、收件人邮箱
        server.quit()  # 关闭连接
    except Exception:  # 如果 try 中的语句没有执行，则会执行下面的 ret=False
        ret = False
    return ret
```

图 5-39 发送邮件

6 编码设计

6.1 编码设计风格

本系统编码设计风格应遵循以下基本规范：

1) 关于系统中出现的变量名、函数名、数据库表名、字段名等应采用能体现其功能或特点的英文单词组合或缩写命名。这样编码的好处是方便理解和记忆，提高代码的可读性。方便一段时间后再次对系统进行迭代时不需要耗费时间去理解变量名、函数名、表名、字段名等，可以快速地投入编码设计中。
2) 程序中复杂过程及关键功能的实现都应在代码中有详细注释，相关功能函数的实现都应在函数定义时有功能说明注释和调用说明注释。这样的好处是便于进行代码的编写、调试和后续的版本迭代。
3) 系统的交互要趋于简单清晰，减少用户的学习成本，方便用户使用的同时要注意有相关的提示，增强用户的体验感受。
4) 系统的各个模块应尽量趋向于独立化，实现模块可以脱离系统整体使用。这样设计的好处是代码编写实现时可以分块进行，而不用时时都考虑到系统的整体。在调试代码时也可以分模块进行调试，减少代码调试、bug 查找、修复时候的工作量。并且同时也可以增强系统的可迭代性，当某个模块不完善需要重构或进一步开发迭代时，可以不触及系统的整体，将模块升级即可。

6.2 编码设计思想

本系统实现对在线商品评价数据的爬取、清理、分析都应部署在服务器后台实现，前台只实现商品 URL 地址的传递和评论分析结果的图表显示。在数据发送和接收后，都应及时验证数据的有效性和合法性，减少系统不必要的性能消耗。

对于系统的实现应在编码开始前做好足够的准备，如熟悉 Django 框架的各个部件，熟悉 Python 代码编写语法和规范，了解文本情感倾向分析的原理和编码过程，防止系统编写过程中，编码的凌乱和不顺畅，增大无谓的代码量。

要善于利用开源社区中的资源，在功能完善的情况下使代码尽量的简洁明了。要善于使用网络上的资源，不应浪费时间去编写轮子。

7 运行结果

程序目前运行稳定，无明显 bug，接下来进行运行结果的展示。系统分为用户界面和管理员界面。用户界面进行在线商品的 URL 地址输入和展示，管理员界面进行数据库的查看和相关系统维护。

7.1 用户界面

用户输入淘宝商品的 URL 并点击提交后如果系统之前进行过相应的分析，则会对该商品的分析结果以图表的形式展示，用户如果输入多个 URL 并以回车进行分割，则会并列展示多个商品的分析图表。分别展示商品的好评度分布图、整体评价分布图、好、中、差评分布图以及历史价格折线图。

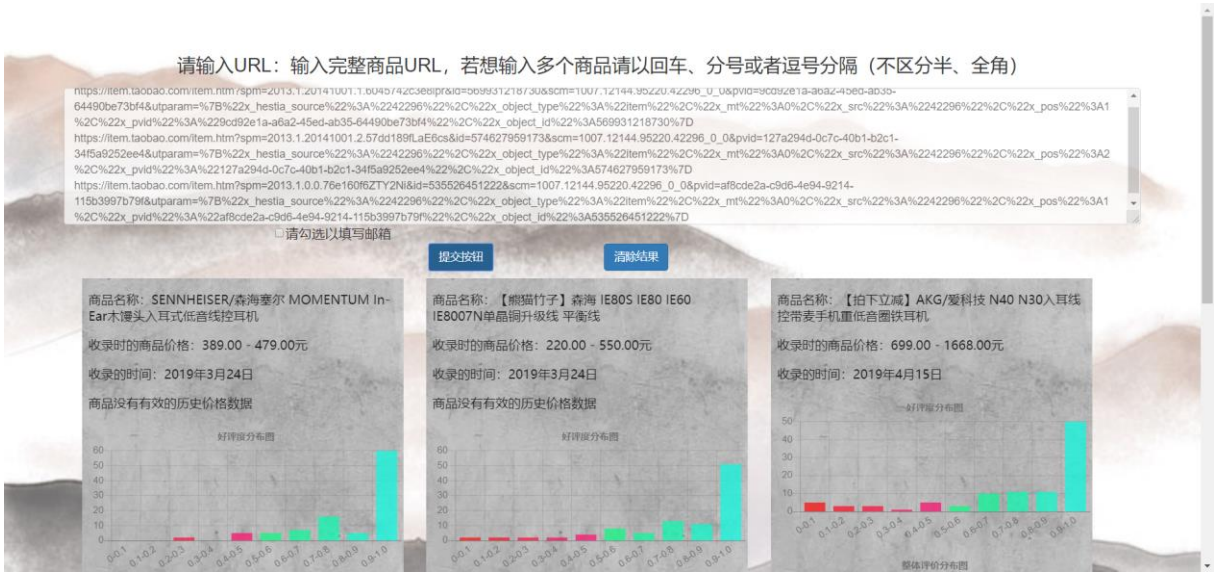

图 7-1 结果查看 1

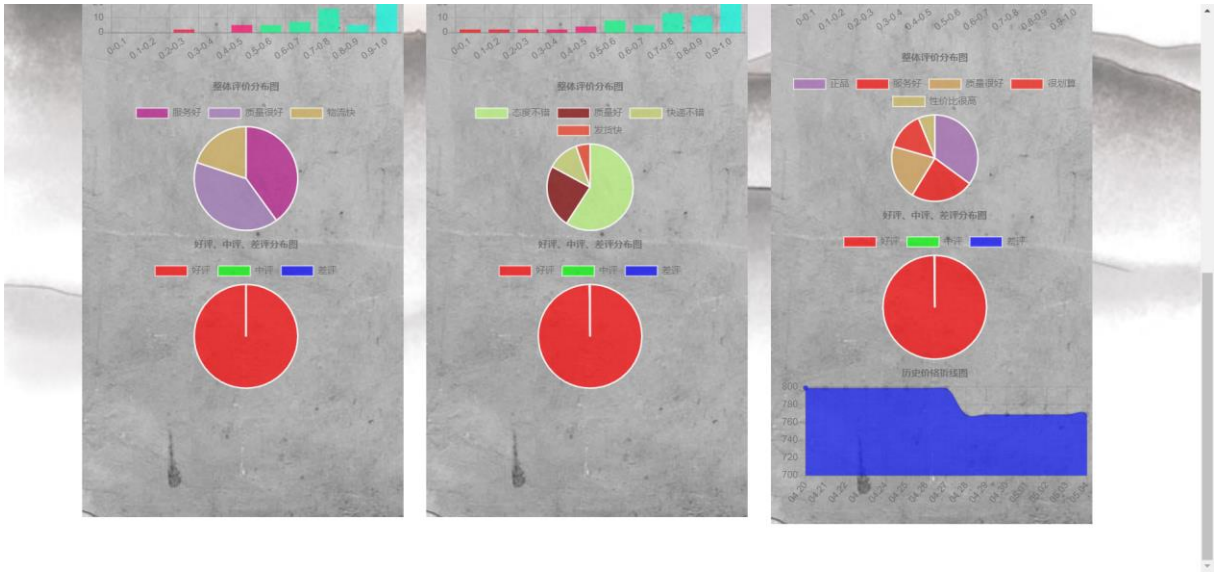

图 7-2 结果查看 2

通过清除按钮可以清除页面上已显示的图表数据，如果系统之前未保存过该商品的数据，将会将在后台进行爬取、分析该商品的信息。

图 7-3 发送新商品

在输入新添加的商品 URL 地址时可以勾选输入邮箱的选框，输入自己的邮箱地址。

图 7-4 勾选邮箱发送

在商品评论数据分析完成后，将会收到提示邮件。

图 7-5 提醒邮件

通过点击邮件中的 url 连接可以查看分析结果。

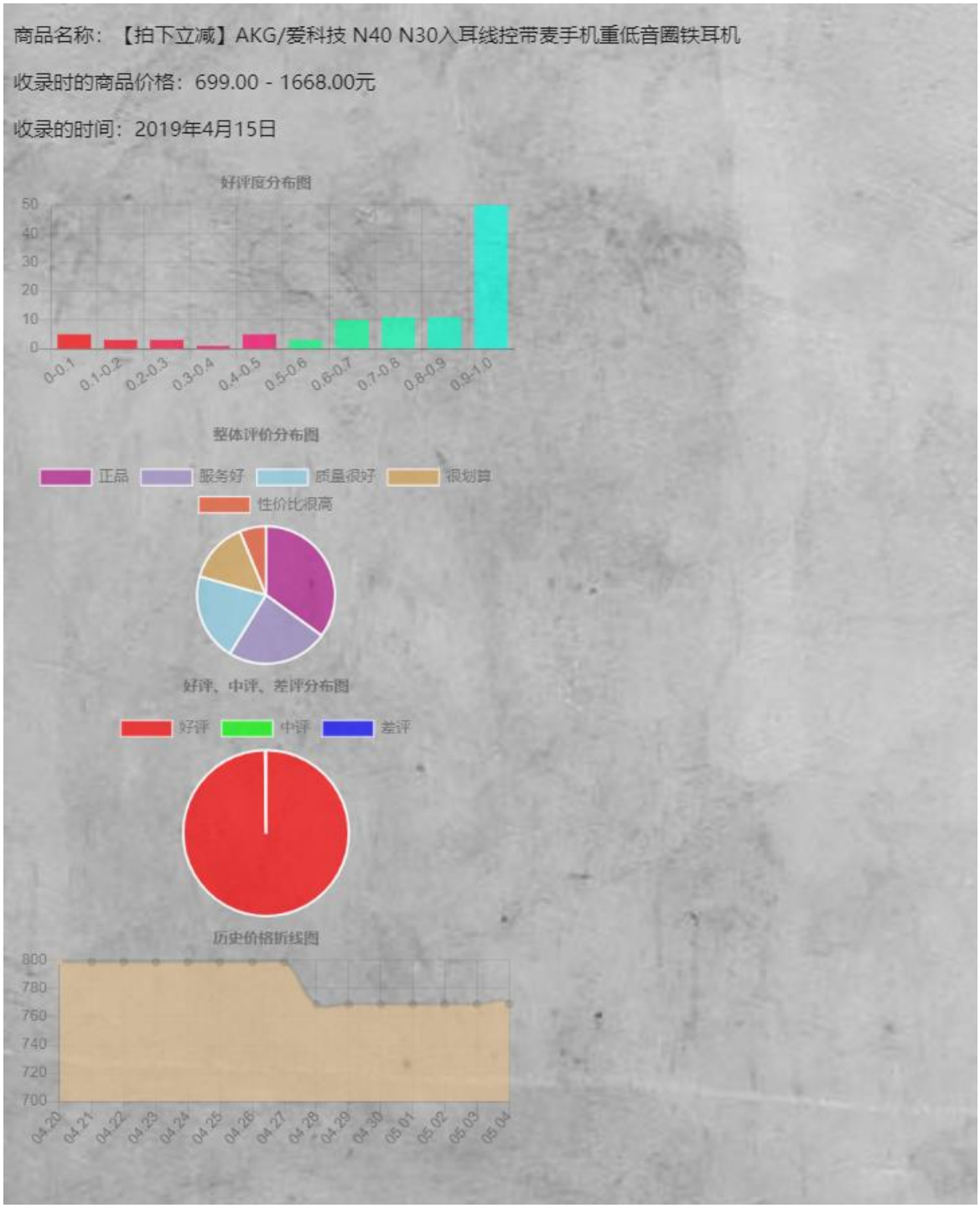

图 7-6 查看分析结果

7.2 管理员界面

管理员界面是基于 django 的 admin 界面上进行设置的，当前提供了对数据库的查看和管理功能。通过登陆/admin 页面后可以查看数据库的信息和对管理人员的创建和授权。

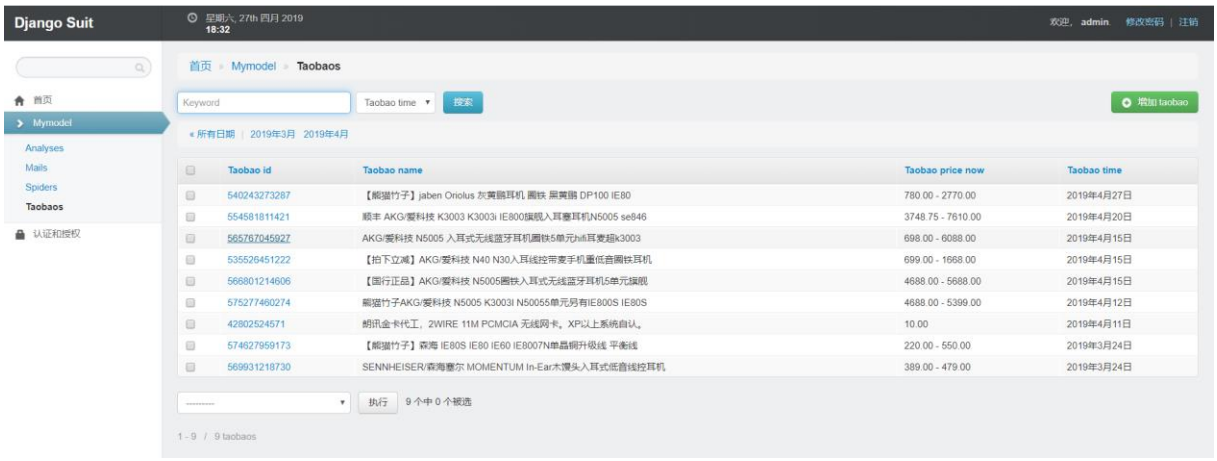

图 7-7 后台查看数据库

图 7-8 后台管理员配置

8 总结

我完成的毕业设计是在线商品评论分析系统，系统基本完成既定目标，已完善系统的相关功能和 bug 的修补，使用上用户体验良好，系统运行流畅。从系统的设计到完成的过程中，我体悟良多，不管是知识的获取能力、应用系统的设计能力、程序代码的编写能力还是项目进度和需求管理能力都感到受益颇深。特别是在文本情感倾向分析模块的设计和编写过程中，对于机器学习、自然语言处理都有了初步的理解和练习。在这次毕业设计的过程中，通过应用系统程序的设计和实现，使我在编程设计的理论和实践上都有了一个提高，尤其是增进了自己分析问题、研究问题、解决问题、进度管理等方面的能力。

通过大学期间的理论学习、每学期的课程设计以及这次的毕业设计的完成，充分的锻炼了自己的理论和实践，这其中除了自己的刻苦学习和认真专研外，也要归功于各位授课教师和指导教师的辛勤教育。感谢各位老师付出的努力和期望。

随着毕业设计的完成，我的大学生涯也面临终点。通过大学四年的学习，我不仅学到了学科里应该掌握的知识而且学会了为人处事的原则和方法。四年大学生涯将会一个美好回忆，而我在大学期间获得的点点滴滴也将受益终身。

参考文献

[1] 于娟,刘强.主题网络爬虫研究综述[J].计算机工程与科学,2015,37(02):231-237.

[2] 魏韡,向阳,陈千.中文文本情感分析综述[J].计算机应用,2011,31(12):3321-3323.

[3] 王洪伟,郑丽娟,尹裴,史伟.在线评论的情感极性分类研究综述[J].情报科学,2012,30(08):1263-1271+1276.

[4] 尹裴,王洪伟.面向产品特征的中文在线评论情感分类:以本体建模为方法[J].系统管理学报,2016,25(01):103-114.

[5] 陈旻,朱凡微,吴明晖,应晶.观点挖掘综述[J].浙江大学学报(工学版),2014,48(08):1461-1472.

[6] [美]韦斯·麦金尼(Wes McKinney).利用 Python 进行数据分析[M].北京:机械工业出版社,2018:102-305.

[7] [美]Steven Bird,[美]Ewan Klein,[美]Edward Loper.Python 自然语言处理[M].北京:人民邮电出版社,2014:98-201.

[8] Ivan Idris.Python 数据分析[M].南京:东南大学出版社,2016:61-89

[9] Kumar Ravi,Vadlamani Ravi. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications[J]. Knowledge-Based Systems,2015,89.

[10] Goyal,Palash.Deep learning for natural language processing :: creating neural networks with Python[M]. Berkeley,CA: Apress,2018:62-103

[11] Harris, Mary Dee.Introduction to natural language processing[M]. Reston, Va: Reston Pub. Co,1985:65-120

致谢

本设计是在指导教师黄雷君老师的亲切关心和细心指定下完成的。黄雷君老师从设计方案的选定、设计计划的安排、设计编码疑难问题的解惑、论文的书写都给予了精心的指导和严格的要求。对我的系统设计开发和论文的结搞都给予了莫大的帮助和关心。在此，我向黄雷君老师表示由衷的感谢和诚挚的谢意。

同时也感谢我的同学和朋友提供的大力支持和帮助。

最后向评审本论文，参加论文答辩的各位老师表示最衷心的感谢。