

**基于Python的广州二手房信息爬取数据分析**



组员：15051054区志瑜

15051056苏冠为

15051081曾瑞斌

15051019郭易胜

15051078郑伟东

**目录**

[**第1章 引言** **3**](#_Toc22765_WPSOffice_Level1)

[1.1 背景介绍 3](#_Toc23906_WPSOffice_Level2)

[1.2 技术工具介绍 3](#_Toc19594_WPSOffice_Level2)

[1.3 项目分工 3](#_Toc25869_WPSOffice_Level2)

[**第2章 需求分析** **4**](#_Toc23906_WPSOffice_Level1)

[2.1基本需求 4](#_Toc11525_WPSOffice_Level2)

[2.2开发软件 4](#_Toc5704_WPSOffice_Level2)

[2.3运行环境 5](#_Toc30190_WPSOffice_Level2)

[2.4库和模块 5](#_Toc10207_WPSOffice_Level2)

[2.5性能需求 6](#_Toc30162_WPSOffice_Level2)

[2.6功能需求 6](#_Toc7508_WPSOffice_Level2)

[2.7机器学习算法介绍与应用 6](#_Toc31921_WPSOffice_Level2)

[**第3章 系统设计** **7**](#_Toc19594_WPSOffice_Level1)

[3.1 Scrapy框架介绍 7](#_Toc32543_WPSOffice_Level2)

[3.2 数据爬虫设计实现 9](#_Toc28591_WPSOffice_Level2)

[3.3功能框图 10](#_Toc27173_WPSOffice_Level2)

[Scrapy架构图 10](#_Toc15173_WPSOffice_Level2)

[代理池框图 10](#_Toc22414_WPSOffice_Level2)

[Cookie池框图 11](#_Toc8233_WPSOffice_Level2)

[**第4章 系统实现** **13**](#_Toc25869_WPSOffice_Level1)

[Item代码 14](#_Toc18595_WPSOffice_Level2)

[Cookie中间件代码 14](#_Toc30789_WPSOffice_Level2)

[**第5章 系统测试** **17**](#_Toc11525_WPSOffice_Level1)

[5.1 预测效果图 17](#_Toc7780_WPSOffice_Level2)

[广州二手房数据 17](#_Toc15462_WPSOffice_Level2)

[户型数量分布—以市区划分 18](#_Toc25325_WPSOffice_Level2)

[户型数量分布 18](#_Toc6303_WPSOffice_Level2)

[面积分布 19](#_Toc15079_WPSOffice_Level2)

[户型和关注人数分布 19](#_Toc3175_WPSOffice_Level2)

[小区数量分布 20](#_Toc10131_WPSOffice_Level2)

[MongoDB数据库 20](#_Toc15608_WPSOffice_Level2)

[Redis数据库(cookie池) 21](#_Toc17979_WPSOffice_Level2)

[Redis数据库(代理池) 21](#_Toc5002_WPSOffice_Level2)

[5.2 项目运行图 21](#_Toc24582_WPSOffice_Level2)

[**第6章 展望与总结** **24**](#_Toc5704_WPSOffice_Level1)

[6.1 总结 24](#_Toc20044_WPSOffice_Level2)

[6.2 展望 24](#_Toc30076_WPSOffice_Level2)

# 第1章 引言

## 背景介绍

由于课程需要，本小组确立题目为基于Python的广州二手房信息数据爬取分析项目的设计与实现。爬虫功能主要的确定功能为：对广州二手房的房价基础信息继续宁爬取和数据可视化分析。

## 技术工具介绍

**IDLE**：Pycharm --PyCharm是一种Python IDE，其带有一整套可以帮助用户在使用Python语言开发时提高其效率的工具，比如， 调试、语法高亮、Project管理、代码跳转、智能提示、自动完成、单元测试、版本控制等等。

**框架**：Scrapy

**非关系型数据库**：mongodb，redis

NoSQL，全称 Not Only SQL，意为不仅仅是 SQL，泛指非关系型的数据库。NoSQL 是基于键值对的，而且不需要经过 SQL 层的解析，数据之间没有耦合性，性能非常高。对于爬虫的数据存储来说，一条数据可能存在某些字段提取失败而缺失的情况，而且数据可能随时调整，另外数据之间能还存在嵌套关系。如果我们使用了关系型数据库存储，一是需要提前建表，二是如果存在数据嵌套关系的话需要进行序列化操作才可以存储，比较不方便。如果用了非关系数据库就可以避免一些麻烦，简单高效。

**MongoDB存储**

MongoDB 是由 C++ 语言编写的非关系型数据库，是一个基于分布式文件存储的开源数据库系统，其内容存储形式类似 Json 对象，它的字段值可以包含其他文档，数组及文档数组，非常灵活

**Redis存储**

Redis 是一个基于内存的高效的键值型非关系型数据库，存取效率极高，而且支持多种存储数据结构

**Coffee**：Execl

## 项目分工

|  |  |
| --- | --- |
| **队员** | **分工** |
| 区志瑜 | 系统框架的确定和项目整体优化 |
| 苏冠为 | 需求分析和系统设计 |
| 曾瑞斌 | 爬虫系统的实现和测试 |
| 郑伟东 | PPT和数据收集 |
| 郭易胜 | PPT和数据库的优化 |

# 第2章 需求分析

## 2.1基本需求

笔记本电脑，基本设计软件工具，设计计划等文档

## 2.2开发软件

Python3.5、Scrapy、pycharm、Redis、MongoDB

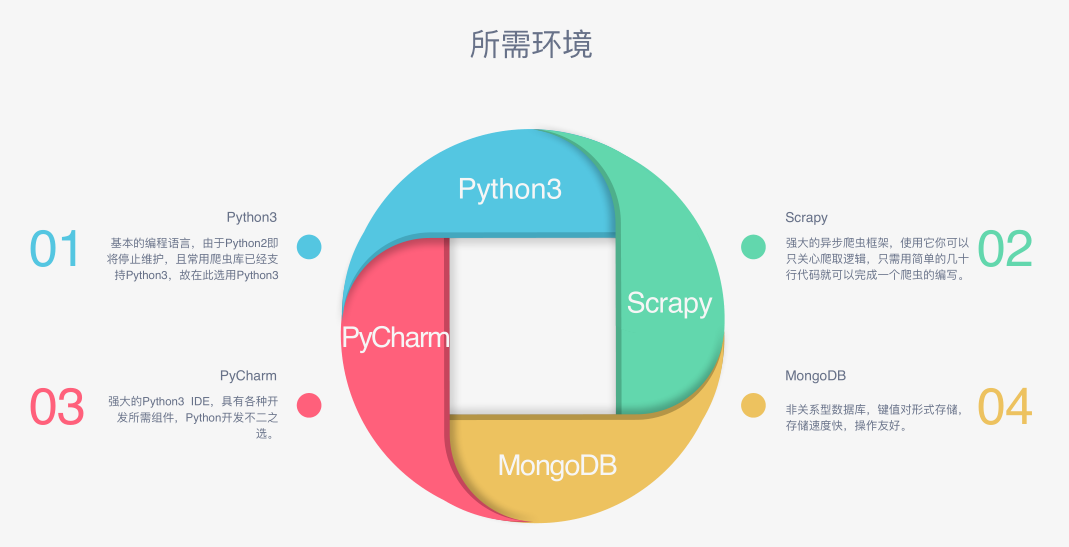


图2-1 所需环境

## 2.3运行环境

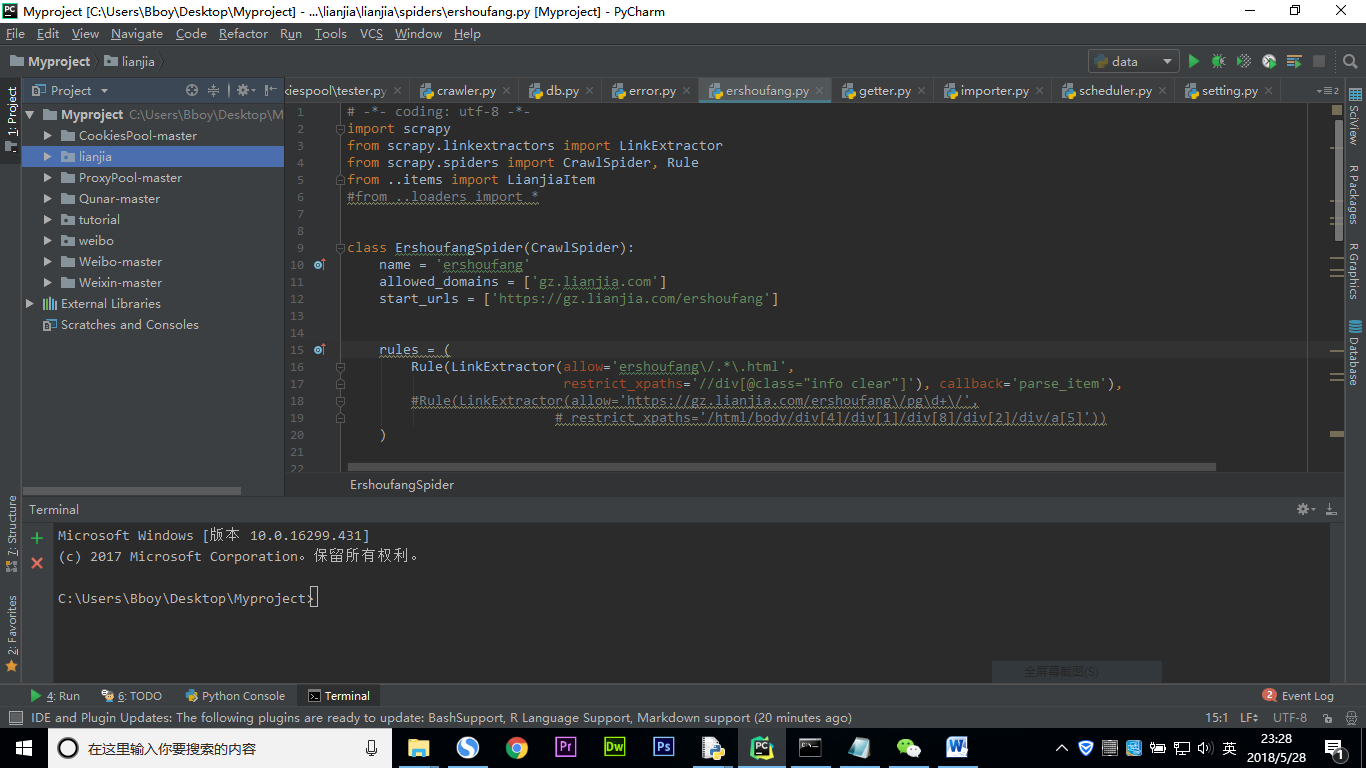


图2-2 运行环境

## 

## 2.4库和模块

**所用到的包和模块**

1、Matplotlib

2、pandas

3、sklearn

4、json

5、requests

6、logging

7、pymongo

8、redis

9、selenium

10、multiprocessing

11、pyquery

12、re

## 2.5性能需求

实时、高效、操作简便、数据在Excel表格中简单明了，一目了然、并且能够看出房价中的差价、对比

## 2.6功能需求

从网页中把数据收集到Excel表格当中，用视图效果显示出当地房价。能够直观的看出房价的差距，各个区的差距。也能够在Excel中查询某个小区的二手房价。

## 2.7机器学习算法介绍与应用

学习系统的基本结构。环境向系统的学习部分提供某些信息，学习部分利用这些信息修改知识库，以增进系统执行部分完成任务的效能，执行部分根据知识库完成任务，同时把获得的信息反馈给学习部分

(1)表达能力强。

(2)易于推理。

(3)容易修改知识库。

(4)知识表示易于扩展。

# 第3章 系统设计

## 3.1 Scrapy框架介绍

Scrapy是一个基于Twisted的异步处理框架，是纯Python实现的爬虫框架，其架构清晰，模块之间的耦合程度低，可扩展性极强，可以灵活完成各种需求。我们只需要定制开发几个模块就可以轻松实现一个爬虫。

它可以分为如下的几个部分：

Engine：引擎，处理整个系统的数据流处理、触发事务，是整个框架的核心。Item。 项目，它定义了爬取结果的数据结构，爬取的数据会被赋值成该Item对 象。

Scheduler：调度器，接受引擎发过来的请求并将其加入队列中，在引擎再次请求 的时候将请求提供给引擎。

Downloader：下载器，下载网页内容，并将网页内容返回给蜘蛛。Spiders。蜘蛛， 其内定义了爬取的逻辑和网页的解析规则，它主要负责解析响应 并生成提取结果和新的请求。

Item Pipeline：项目管道，负责处理由蜘蛛从网页中抽取的项目，它的主要任务 是清洗、验证和存储数据。

Downloader Middlewares：下载器中间件，位于引擎和下载器之间的钩子框架， 主要处理引擎与下载器之间的请求及响应。

Spider Middlewares：蜘蛛中间件，位于引擎和蜘蛛之间的钩子框架，主要处理蜘 蛛输入的响应和输出的结果及新的请求。

Scrapy中的数据流由引擎控制，数据流的过程如下：

* Engine首先打开一个网站，找到处理该网站的Spider，并向该Spider请求第一个要爬取的URL。
* Engine从Spider中获取到第一个要爬取的URL，并通过Scheduler以Request的形式调度。
* Engine向Scheduler请求下一个要爬取的URL。
* Scheduler返回下一个要爬取的URL给Engine，Engine将URL通过Downloader Middlewares转发给Downloader下载。
* 一旦页面下载完毕，Downloader生成该页面的Response，并将其通过Downloader Middlewares发送给Engine。
* Engine从下载器中接收到Response，并将其通过Spider Middlewares发送给Spider处理。
* Spider处理Response，并返回爬取到的Item及新的Request给Engine。
* Engine将Spider返回的Item给Item Pipeline，将新的Request给Scheduler。
* 重复第二步到最后一步，直到Scheduler中没有更多的Request，Engine关闭该网站，爬取结束。

通过多个组件的相互协作、不同组件完成工作的不同、组件对异步处理的支持，Scrapy最大限度地利用了网络带宽，大大提高了数据爬取和处理的效率。

Scrapy框架和pyspider不同，它是通过命令行来创建项目的，代码的编写还是需要IDE。项目创建之后，项目文件结构如下所示：

scrapy.cfg

project/

\_\_init\_\_.py

items.py

pipelines.py

settings.py

middlewares.py

spiders/

\_\_init\_\_.py

spider1.py

spider2.py

...这里各个文件的功能描述如下：

scrapy.cfg：它是Scrapy项目的配置文件，其内定义了项目的配置文件路径、部 署相关信息等内容。

items.py：它定义Item数据结构，所有的Item的定义都可以放这里。

pipelines.py：它定义Item Pipeline的实现，所有的Item Pipeline的实现都可以放 这里。

settings.py：它定义项目的全局配置。

middlewares.py：它定义Spider Middlewares和Downloader Middlewares的实现。spiders：其内包含一个个Spider的实现，每个Spider都有一个文件。

## 3.2 数据爬虫设计实现

利用Scrapy框架(cookie池和代理池)爬取广州二手房基本信息，并存到Mongodb数据库和生成Execl表格，并对数据进行可视化操作。

**爬取的内容：**区域、小区名、总价、房型、面积、单价、朝向、楼层位置、装修情况、建筑时间、是否有电梯、产权类型、住宅类型、发布日期

**信息保存：**

1.存到Mongodb数据库和生成Execl表格

2.能在Excel中查询广州某地区的二手房房价，能够进行对比和比较

3.能够用视图直观的看到广州市的房价分布

## 3.3功能框图



图3-1 Scrapy架构图

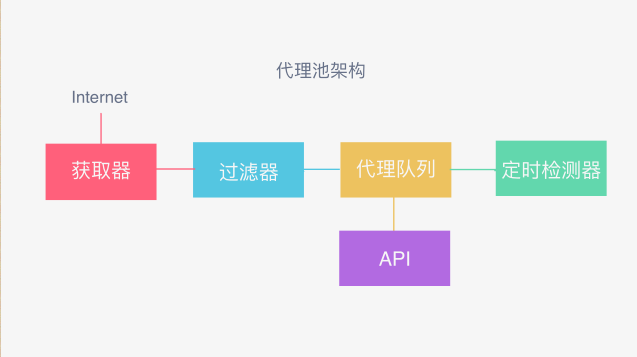


图3-2 代理池框图

为什什么要用代理池？

许多网站有专门的反爬虫措施，可能遇到封IP等问题。互联网上公开了了大量量免费代理，利用好资源。通过定时的检测维护同样可以得到多个可用代理理。

代理理池的要求

多站抓取，异步检测，定时筛选，持续更新提供接口，易于提取

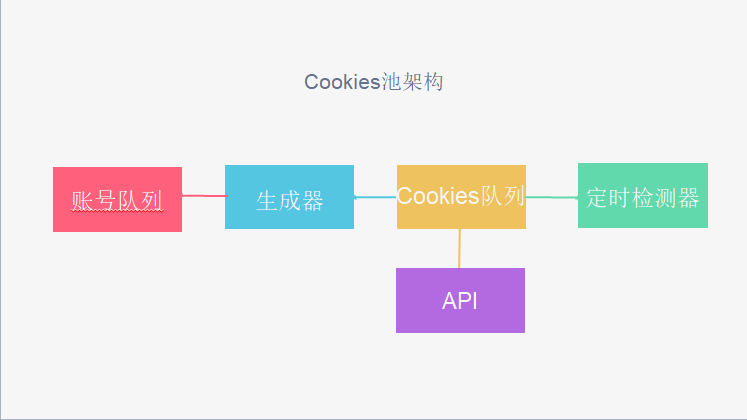


图3-3 Cookie池框图

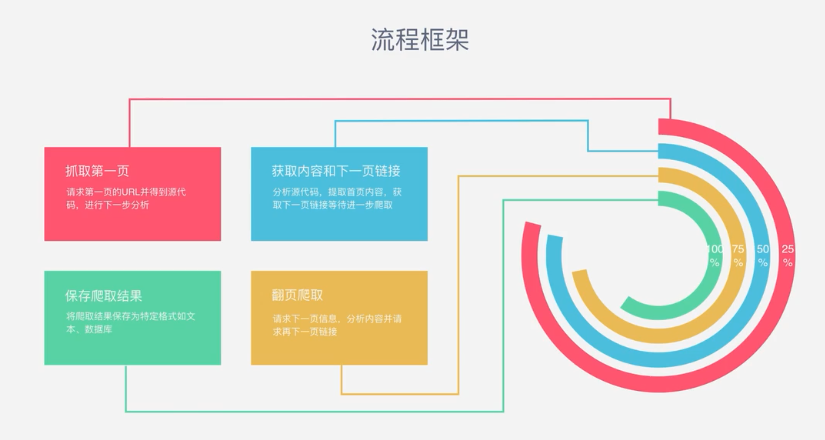
为什什么要⽤用Cookies池？

网站需要登录才可爬取，例如新浪微博爬取过程中，如果频率过高会导致封号，需要维护多个账号的Cookies池，实现大规模爬取。

Cookies池的要求

自动登录更新，定时验证筛选，提供外部接口

3.4设计流程图图3-4 Scrapy流程框架



# 第4章 系统实现

## 4.1 功能代码实现

图4-1 开发者选项

通过开发者工具查看html源代码,确定每个标题所在的链接，在a标签里面，通过如下代码进行url的捕获，代码如下：

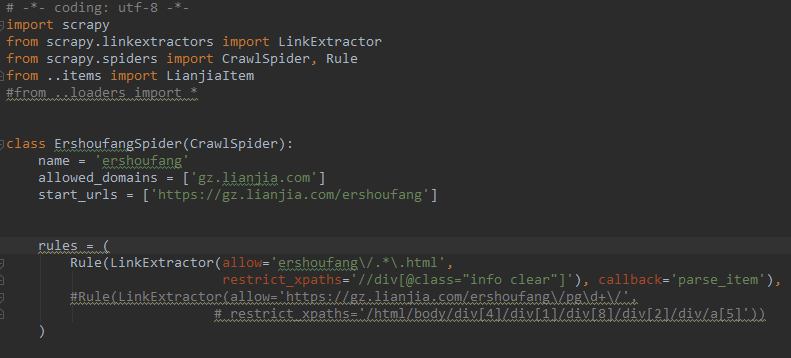


图4-2 Spider代码(1)



图4-3 Spider代码(2)

## 创建Item

接下来我们解析用户的基本信息并生成Item。这里我们先定义几个Item，如

标题,小区,价格,地区的item，如下所示：

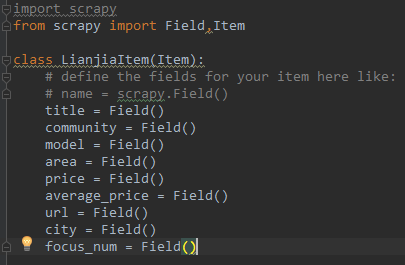
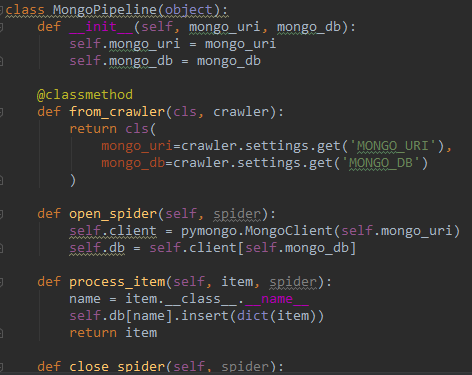


图4-4 Item代码

## 数据存储

数据清洗完毕之后，我们就要将数据保存到MongoDB数据库。我们在这里实现MongoPipeline类，如下所示：

图4-5 保存数据库到mongodb代码

open\_spider()方法里添加了Collection的索引，在这里为两个Item都添加了索引，索引的字段是id。由于我们这次是大规模爬取，爬取过程涉及数据的更新问题，所以我们为每个Collection建立了索引，这样可以大大提高检索效率。

在process\_item()方法里存储使用的是update()方法，第一个参数是查询条件，第二个参数是爬取的Item。这里我们使用了$set操作符，如果爬取到重复的数据即可对数据进行更新，同时不会删除已存在的字段。如果这里不加$set操作符，那么会直接进行item替换，这样可能会导致已存在的字段如价格和题目列表清空。第三个参数设置为True，如果数据不存在，则插入数据。这样我们就可以做到数据存在即更新、数据不存在即插入，从而获得去重的效果。

对于关注人数和地区列表，我们使用了一个新的操作符，叫作$addToSet，这个操作符可以向列表类型的字段插入数据同时去重。它的值就是需要操作的字段名称。这里利用了$each操作符对需要插入的列表数据进行了遍历，以逐条插入关注人数或地区数据到指定的字段。

## Cookies池对接

链家的反爬能力非常强，我们需要做一些防范反爬虫的措施才可以顺利完成数据爬取。

如果没有登录而直接请求微博的API接口，这非常容易导致403状态码。这个情况我们在Cookies池一节也提过。所以在这里我们实现一个Middleware，为每个Request添加随机的Cookies。

我们先开启Cookies池，使API模块正常运行。例如在本地运行5000端口，访问：http://localhost:5000/weibo/random，即可获取随机的Cookies。当然也可以将Cookies池部署到远程的服务器，这样只需要更改访问的链接。

我们在本地启动Cookies池，实现一个Middleware，如下所示：

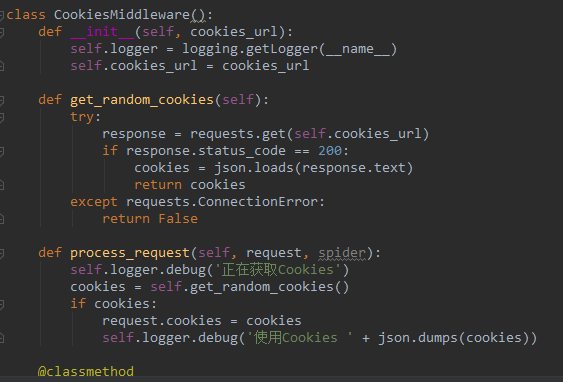


图4-6 Cookie中间件代码

我们首先利用from\_crawler()方法获取了COOKIES\_URL变量，它定义在settings.py里，这就是刚才我们所说的接口。接下来实现get\_random\_cookies()方法，这个方法主要就是请求此Cookies池接口并获取接口返回的随机Cookies。如果成功获取，则返回Cookies；否则返回False。

接下来，在process\_request()方法里，我们给request对象的cookies属性赋值，其值就是获取的随机Cookies，这样我们就成功地为每一次请求赋值Cookies了。

如果启用了该Middleware，每个请求都会被赋值随机的Cookies。这样我们就可以模拟登录之后的请求，403状态码基本就不会出现。

## 代理池对接

链家还有一个反爬措施就是，检测到同一IP请求量过大时就会出现414状态码。如果遇到这样的情况可以切换代理。例如，在本地5555端口运行，获取随机可用代理的地址为：http://localhost:5555/random，访问这个接口即可获取一个随机可用代理。接下来我们再实现一个Middleware，代码如下所示：



图4-7 代理中间件代码

同样的原理，我们实现了一个get\_random\_proxy()方法用于请求代理池的接口获取随机代理。如果获取成功，则返回改代理，否则返回False。在process\_request()方法中，我们给request对象的meta属性赋值一个proxy字段，该字段的值就是代理。

另外，赋值代理的判断条件是当前retry\_times不为空，也就是说第一次请求失败之后才启用代理，因为使用代理后访问速度会慢一些。所以我们在这里设置了只有重试的时候才启用代理，否则直接请求。这样就可以保证在没有被封禁的情况下直接爬取，保证了爬取速度。

## 启用Middleware

接下来，我们在配置文件中启用这两个Middleware，修改settings.py如下所示：

注意这里的优先级设置，前文提到了Scrapy的默认Downloader Middleware的设置如下：

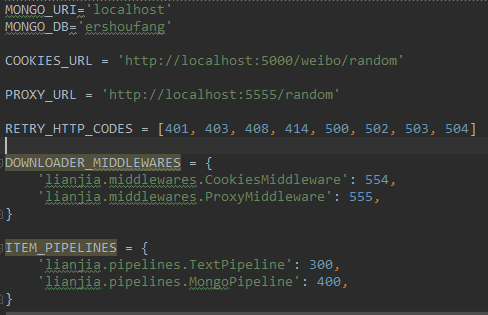


图4-8 Setting代码

要使得我们自定义的CookiesMiddleware生效，它在内置的CookiesMiddleware之前调用。内置的CookiesMiddleware的优先级为700，所以这里我们设置一个比700小的数字即可。

要使得我们自定义的ProxyMiddleware生效，它在内置的HttpProxyMiddleware之前调用。内置的HttpProxyMiddleware的优先级为750，所以这里我们设置一个比750小的数字即可。

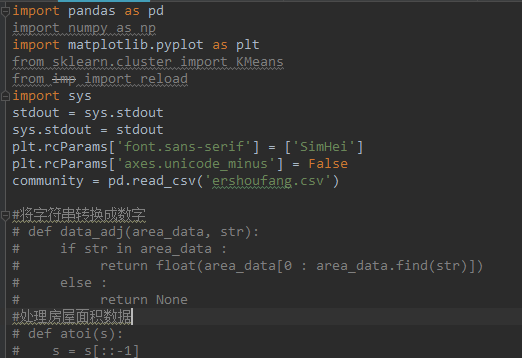


图4-9 数据可视化分析部分代码

导入panda，numpy，matplotlib和sys模块读取ershoufang.csv文件下的数据，进行户型数量分布—以市区划分，户型数量分布，面积分布，户型和关注人数分布，小区数量的数据可视化分析。

# 第5章 系统测试

## 5.1 **预测效果图**

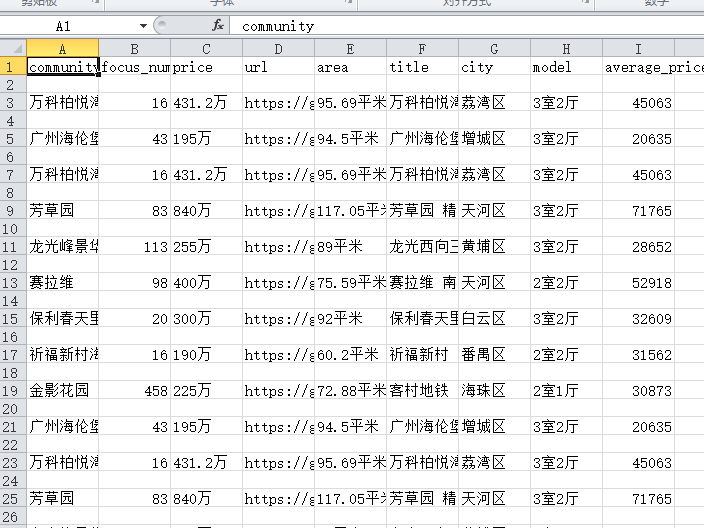


图5-1 广州二手房数据

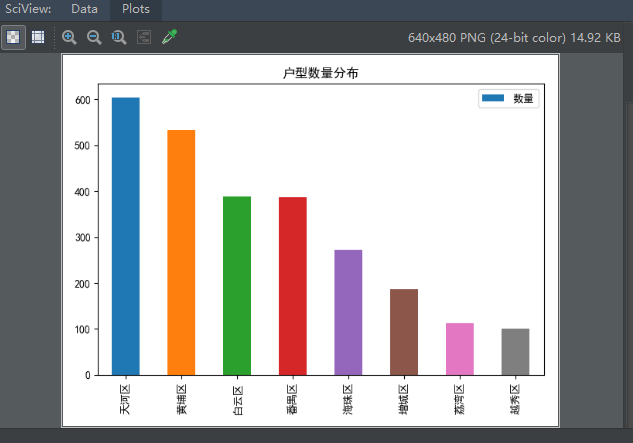


图5-2 户型数量分布—以市区划分

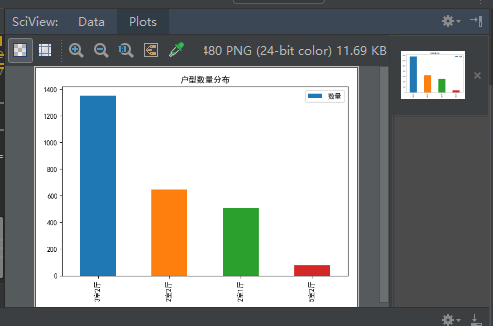


图5-3户型数量分布

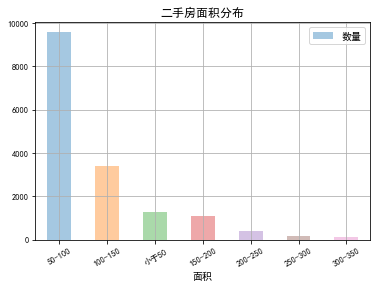


图5-4面积分布

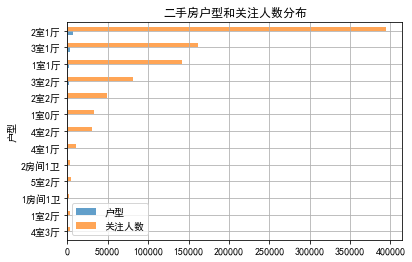


图5-5户型和关注人数分布

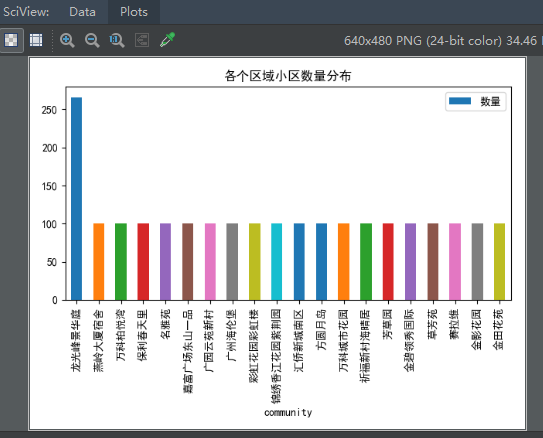


图5-6小区数量分布

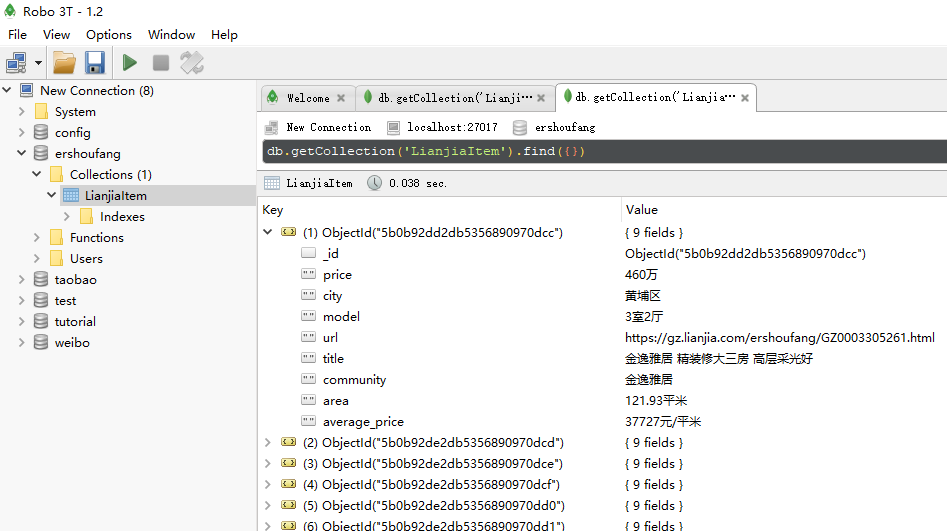


图5-7 MongoDB数据库

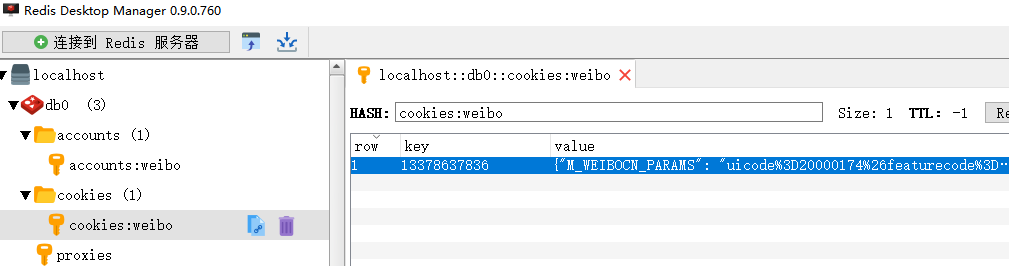


图5-8 Redis数据库(cookie池)

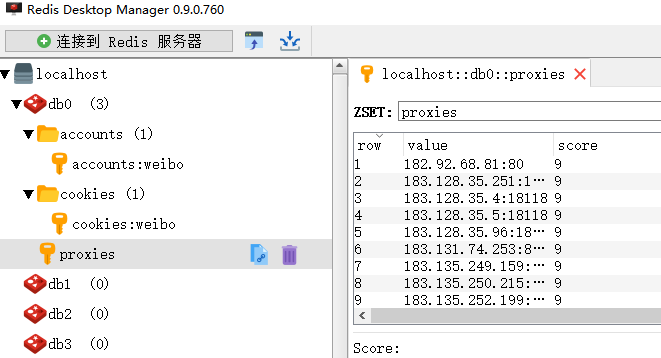


图5-9 Redis数据库(代理池)

## 

## 5.2 项目运行图

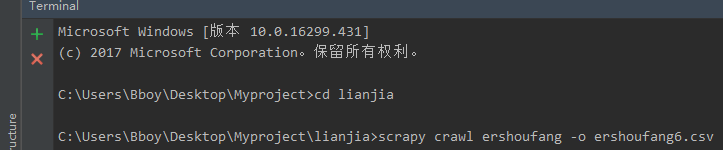
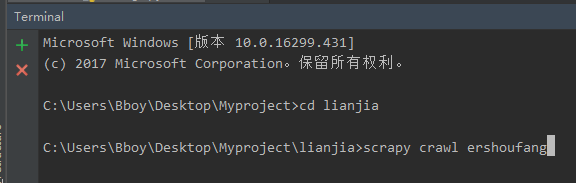


图5-10 保存为csv格式命令



5-11 运行程序命令

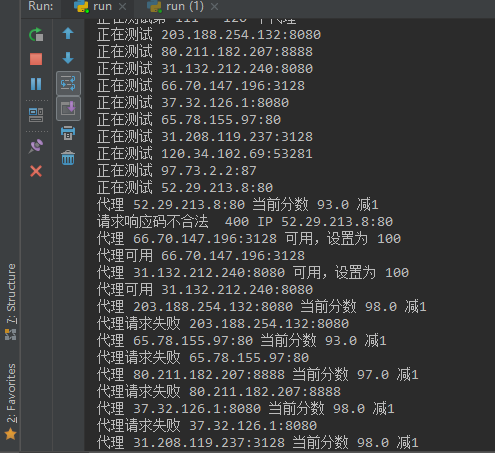


图5-12 代理池开启过程(对代理的质量进行评分)

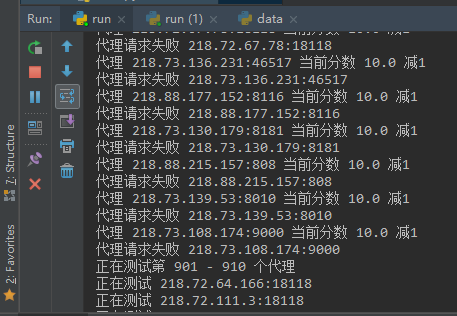


图5-13代理池开启过程(对代理的质量进行评分)

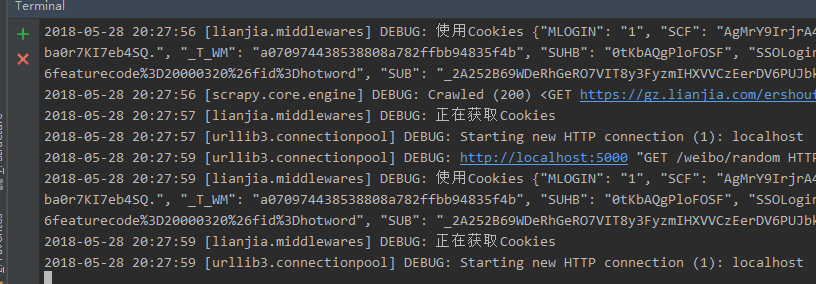


图5-14收集数据过程

图5-15收集数据过程



# 第6章 展望与总结

## 6.1 总结

该广州二手房房价数据爬取分析的爬虫系统主要是在学完Python课程后，对已有的课程知识加以运用，并学习新的技术有机地结合起来，里面运用了BScrapy、代理池、Cookie池等技术，爬取到了关于广州二手房的户型数量分布—以市区划分、户型数量分布、面积分布、户型和关注人数分布、小区数量分布等信息，并对数据加以可视化分析。

## 6.2 展望

爬虫系统的设计与实现涉及多方面的理论、方法和技术，本系统也还有许多新的问题需要解决，需要在实际应用中不断积累和完善，还需要做进一步的研究和开发。并且，我们小组也对爬虫系统进行一个未来的功能展望：

怎样把爬虫系统和人工智能结合起来，使得人工智能能够自动在网上爬取庞大的网络数据，从而实现不断巨量的自我深度学习呢？如果能够实现这个功能，那只需要时间的积累，这个人工智能也将变得无比的智能与强大。