# 摘要

随着中国经济的发展，房地产行业也是一路走高，中国的房价一次次增加，尽管政府也进行了宏观的调控，但是房价还是依旧高居不下。而且我国每年的大学毕业生有800多万人，大学生作为刚刚踏入社会的一个群体，暂时没有太高收入，因此租房成为了这些群体的第一选择。

但是由于没有太高的收入，因此要想租到心仪的房子是需要花费相当时间的，虽然说由于互联网的发展，可以很方便的在网上找到，但是，互联网上的信息有很多是假的，甚至是专门用来骗人的。而且正是由于互联网上的信息很多，也让人难以找到目标。

本文便是针对这种问题，利用python爬虫的技术，爬取链家网上的房源，并对这些房源进行分析，可视化，希望能够为租房的人提供一个非常直观的参考，寻找到性价比最高的房子，既节省了花费，还能住得舒服。本系统的实现需要四大部分，数据爬取部分，数据分析部分，数据可视化部分，系统界面实现部分。使用的技术包括但不限于python，scrapy框架，pandas库，numpy库，matplotlib库，Tkinter 等。

**关键词：**租房；python；数据分析；可视化；GUI

# ABSTRACT

With the development of China's economy, the real estate industry is also on the way up, and the housing price in China has been increasing again and again. Despite the macro-control of the government, the housing price is still high.In addition, there are more than 8 million college graduates in China every year. As a group just entering the society, college students do not have too high income for the time being, so renting becomes the first choice for these groups.

But because do not have too high income, so want to rent the house that admires in the heart to need to spend quite time, although say because of the development of Internet, can very convenient find on the net, but, the information on the Internet has a lot of false, it is used technically even cheat people.And because there's so much information on the Internet, it's hard to find targets.

This paper is aimed at this problem, using python crawler technology, climb the HOME LINK online housing sources, and the analysis of these housing sources, visualization, hoping to provide a very intuitive reference for people who rent, looking for the house with the highest cost performance, not only save the cost, but also can live comfortably.The realization of this system needs four parts: data crawling part, data analysis part, data visualization part and system interface realization part.Techniques used include, but are not limited to, python, scrapy frameworks, pandas, the numpy library, the matplotlib library, Tkinter, and more.

**Key words:** rent;python;data analysis;visualization;GUI

目 录

[1.前言 1](#_Toc216708842)

[1.1秘密共享体制的研究背景及意义 1](#_Toc216708843)

[1.1.1研究背景 1](#_Toc216708848)

[1.1.2研究意义 2](#_Toc216708849)

[致谢 3](#_Toc216708906)

[参考文献 5](#_Toc216708904)

[附录 7](#_Toc216708905)

# 1.前言（每章开始在奇数页）

## 课题研究的背景及意义

自住房市场化改革以来，我国楼市一项十分突出的特征就是高房价，并逐渐演化成社会问题。为此，政府也采用宏观调控的手段对租房市场进行干预，但其调控结果似乎并不怎么乐观，甚至陷入“越调越涨”的怪圈。

2018年以来，中央和地方从不同角度为住房租赁市场制定多项政策，住房租赁市场发展作为住房制度改革的核心内容，相关配套政策进一步细化并出台，大力推动住房租赁市场发展。

我国租赁人口基数庞大，约为1.9 亿人。其中，外出务工人员和刚步入工作的大学毕业生是租赁市场的主力军。 我国目前每年新增大学毕业生约800 万人，异地就业主要集中在一二线城市。大学生毕业生在步入工作的5 年之内，主要依靠租赁解决居住需求。

在如此背景下，租房也就会越来越火热，但是随着住房租赁市场的快速发展，一些问题也会随之出现。租金贷乱象、热点城市房租不合理暴涨、长租公寓企业违法违规操作等。

那么如何租到心仪的房子，就需要多方面参考，可是这是一项不小的工作。好在我们现在生活的社会是一个信息爆炸的社会，因此我们只需要得到这些数据，然后分析它，得到我们想要的结果。

## 开发工具及技术

本设计是使用python3.7语言进行的设计，需要的开发技术包括：scrapy框架，numpy库，pandas库，matplotlib库，Tkinter库等。全部工程是在Windows系统下使用pycharm，spyder，mysql工具进行开发。

### 1.2.1开发技术介绍

（1）python语言

1、python的产生

Python是由创始人贵铎·范·罗萨姆（Guido van Rossum）在阿姆斯特丹于1989年圣诞节期间，为了打发圣诞节的无趣，开发的一个新的解释型脚本语言。

Python的第一个公开发行版发行于1991年，它是纯粹的自由软件， 源代码和解释器（CPython）都遵循 GPL(GNU General Public License)协议。

1. python的设计思想

Python被设计成是 “符合大脑思维习惯” 的，采用极简主义的设计理念，加以统一规范的交互模式。Python 开发者的哲学是 “用一种方法，最好是只有一种方法来做一件事”。

1. python的特点

Python是一种解释型语言，目前Python的标准实现方式是将源代码的语句编译（转换）为字节码格式，然后通过解释器将字节码解释出来。

Python是完全面向对象的编程语言，函数、模块、数字、字符串等内置类型都是对象。

1. python的优缺点

Python秉承了简洁、清晰的语法，以及高度一致的编程模式。始终如一的设计风格，可以保证开发出相当规范的代码。

Python往往只要几十行代码就可以开发出需要几百行C代码的功能。开发速度非常快。

python有着非常丰富的第三方库，覆盖了各大方向：web，数据分析，科学计算，网络编程，网络爬虫等。

但是正是由于Python解释型语言的特点，其程序运行较慢，而且代码不能加密。

1. python的种类

Cpython：

执行过程：程序—>（c解释器）—>（字节码）—>（机器码）—>cpu

Python的官方版本，这是Python初始的以及维护得最好的实现，使用C编写。新的语言特性一般会最先在这里出现。CPython实现会将源文件（py文件）转换成字节码文件（pyc文件），然后运行在Python虚拟机上。

Jyhton：

执行过程：程序—>（java解释器）—>（字节码）—>（机器码）—>cpu

Python的Java实现，此实现可用作Java应用程序的脚本语言，或可用于使用Java类库创建应用程序，它也经常用于为Java库创建测试。Jython会将Python代码动态编译成Java字节码，然后在JVM上运行。

PyPy：

执行过程：程序—>（字节码）—>（机器码）—>cpu

完全用Python编写的Python实现。它支持在其他实现中没有找到的几个高级功能，如stackless支持和一个Just in Time编译器。PyPy将Python的源码翻编译成字节码再编译成机器码。

1. 爬虫结技术

首先，什么是爬虫？通俗的定义：爬虫是一个模拟人类请求网站行为的程序。可以自动请求网页、并数据抓取下来，然后使用一定的规则提取有价值的数据。

通用爬虫：通用爬虫是搜索引擎抓取系统（百度、谷歌、搜狗等）的重要组成部分。主要是将互联网上的网页下载到本地，形成一个互联网内容的镜像备份。

聚焦爬虫：是面向特定需求的一种网络爬虫程序，他与通用爬虫的区别在于：聚焦爬虫在实施网页抓取的时候会对内容进行筛选和处理，尽量保证只抓取与需求相关的网页信息。

因为我们请求网站都是通过浏览器进行的，因此就必须了解浏览器是如何工作的。在此之前先来看看什么是HTTP和HTTPS：

HTTP:超文本传输协议（英文：HyperText Transfer Protocol，缩写：HTTP）是一种用于分布式、协作式和超媒体信息系统的应用层协议[1]。HTTP是万维网的数据通信的基础。设计HTTP最初的目的是为了提供一种发布和接收HTML页面的方法。

HTTPS：超文本传输安全协议（英语：Hypertext Transfer Protocol Secure，缩写：HTTPS，常称为HTTP over TLS，HTTP over SSL或HTTP Secure）是一种通过计算机网络进行安全通信的传输协议。

在浏览器中发送一个http请求的过程：

1.当用户在浏览器的地址栏中输入一个URL并按回车键之后，浏览器会向HTTP服务器发送HTTP请求。HTTP请求主要分为“Get”和“Post”两种方法。

2.当我们在浏览器输入URL http://www.baidu.com 的时候，浏览器发送一个Request请求去获取 http://www.baidu.com 的html文件，服务器把Response文件对象发送回给浏览器。

3.浏览器分析Response中的 HTML，发现其中引用了很多其他文件，比如Images文件，CSS文件，JS文件。 浏览器会自动再次发送Request去获取图片，CSS文件，或者JS文件。

4.当所有的文件都下载成功后，网页会根据HTML语法结构，完整的显示出来了。

爬虫就是模拟浏览器的这个过程，以使程序能够自动完成大量网络请求和数据下载。

1. scrapy框架

Scrapy是一个为了爬取网站数据，提取结构性数据而编写的应用框架。可以应用于包括数据挖掘，信息处理或存储历史数据等一系列的程序中。

其最初是为了页面抓取（更确切来说，网络抓取）所设计的，也可以应用在获取API所返回的数据（例如Amazon Associates Web Services）或者通用的网络爬虫.

学习和理解scrapy框架，那么是怎么也离不开它的架构的：



图1-2-1 Scrapy架构图

Scrapy主要包括了以下组件：

**引擎（Scrapy）：**用来处理整个系统的数据流处理，触发事务（框架核心）

**调度器（Scheduler）：**用来接受引擎发过来的请求，压入队列中，并在引擎再次请求的时候返回。可以想像成一个URL（抓取网页的网址或者说是链接）的优先队列，由它来决定下一个要抓取的网址是什么，同时去除重复的网址

**下载器（Downloader）：**用于下载网页内容，并将网页内容返回给蜘蛛（Scrapy下载器是建立在twisted这个高效的异步模型上的）

**爬虫（Spiders）：**爬虫是主要干活的，用于从特定的网页中提取自己需要的信息，即所谓的实体（Item）。用户也可以从中提取出链接，让Scrapy继续抓取下一个页面

**项目管道（Pipeline）：**负责处理爬虫从网页中抽取的实体，主要的功能是持久化实体，验证实体的有效性，清除不需要的信息。当页面被爬虫解析后，将被发送到项目管道，并经过几个特定的​​次序处理数据。

**下载器中间件（Downloader Middlewares）：**位于Scrapy引擎和下载器之间的框架，主要是处理Scrapy引擎与下载器之间的请求及响应。

**爬虫中间件（Spider Middlewares）：**介于Scrapy引擎和爬虫之间的框架，主要工作是处理蜘蛛的响应输入和请求输出。

**调度中间件（Scheduler Middewares）：**介绍Scrapy引擎和调度之间的中间件，从Scrapy引擎发送到调度的请求和响应。

scrapy的运行流程大致如下：

首先，引擎从调度器中取出一个链接（URL）用于接下来的抓取；

引擎把URL封装成一个请求（请求）传给下载器，下载器把资源下载下来，并封装成应答包（响应）；

然后，爬虫解析响应；

若是解析出实体（物品），则交给实体管道进行进一步的处理；

若是解析出的是链接（URL），则把URL交给调度等待抓取。

1. Numpy库

NumPy（Numerical Python的缩写）是一个开源的Python科学计算库。使用NumPy，就可以很自然地使用数组和矩阵。NumPy包含很多实用的数学函数，涵盖线性代数运算、傅里叶变换和随机数生成等功能。

NumPy中数组的存储效率和输入输出性能均远远优于Python中等价的基本数据结构（如嵌套的list容器）。其能够提升的性能是与数组中元素的数目成比例的。对于大型数组的运算，使用NumPy很有优势。

但是也有其局限性，大部分NumPy模块是用C语言实现的，使用其他类的编程语言，如jython。

1. Pandas库

pandas是一个专门用于数据分析的开源python库。目前，所有使用python语言研究和数据分析的专业人士，在工作中都会使用到pandas库。pandas能够以最简单的方式提供数据处理、数据抽取和数据操作所需的全部工具。而又由于pandas是在numpy的基础上进行的设计，因此，它不仅可以和其他大多数模块兼容，而且哈能借助numpy模块杂计算方面性能高的优势。

pandas主要使用两种数据结构：

Series：一种类似于一维数组的对象，它是由一组数据（各种Numpy数据类型）以及一组与之相关的数据标签（即索引）组成。仅由一组数据即可产生简单的Series。

DataFrame：一个表格型的数据结构，含有一组有序的列，每列可以是不同的值类型（数值、字符串、布尔值等），DataFrame既有行索引也有列索引，可以被看做是由Series组成的字典。

1. Matplotlib库

Matplotlib是一个用于在Python中制作数组的2D绘图的库。尽管它起源于模拟MATLAB® [1]图形命令，但它独立于MATLAB，可以以Pythonic，面向对象的方式使用。尽管Matplotlib主要是用纯Python编写的，但它大量使用NumPy和其他扩展代码，即使对于大型数组也能提供良好的性能。

Matplotlib的设计理念是，您只需几个命令即可创建简单的绘图，或只需一个！Matplotlib试图让简单的事情更简单，困难的事情成为可能。你只需几行代码即可生成绘图，直方图，功率谱，条形图，误差图，散点图等。

但是Matplotlib的缺点也很明星，高度依赖其他包，如Numpy；只适用于Python：很难或不可能在Python以外的语言中使用。

1. Tkinter库

Tkinter是python的默认GUI库。它基于Tk工具包，该工具包最初是为工具命令语言设计的。Tk普及后被移植到包括python在内的很多其他脚本语言。

公认的是Tkinter开发的界面比较丑，不灵活，扩展功能差，模块少等。这些是Tkinter的缺点，这是由他的特性决定的。它的最大优势就是，它是python内置模块，不需要进行额外的安装，对于刚开始接触GUI编程的人来说很友好。同样的，因为其内置模块的属性，使得其在程序打包为EXE或其他可执行文件的时候，打包出来的程序文件不会特别的大,这样对程序的传播还是很有用处的。

### 1.2.2开发工具介绍

（1）pycharm

Pycharm是由jetbrains开发的优秀的python IDE。正如所有其它 JetBrains 集成开发环境一样，PyCharm 具有智能代码编辑器，能理解 Python 的特性并提供卓越的生产力推进工具：自动代码格式化、代码完成、重构、自动导入和一键代码导航等。这些功能在先进代码分析程序的支持下，使 PyCharm 成为 Python 专业开发人员和刚起步人员使用的有力工具。

* PyCharm 提供以下重要功能：
* 针对 Django 开发的高级支持，包括智能代码完成、检查、重构和特定代码导航。
* 综合 Python 调试器和图形单元测试工具。
* 支持先进 Python 开发工具链，包括 virtualenv 和 buildout。
* 版本控制系统整合——针对 Mercurial、Git、SVN、Perforce 和 CVS 等大多数版本控制系统的统一用户界面 (UI)。
* Google App Engine 支持，能到 App Engine 服务器上运行和部署用户的应用，同时确保代码满足沙盒环境需求。
* REPL 和 Django 控制台，具备胜过标准控制台的众多优势：运行中语法检查;括号和引号匹配和自动配对;以及代码完成。
* Pycharm集合PEP规范，几乎不再用担心代码规范问题。

1. Spyder

Spyder（前身为Pydee）是一个使用Python语言的开放源代码跨平台科学运算集成开发环境。Spyder集成了NumPy，SciPy，Matplotlib与IPython，以及其他开源软件。

与其他科学数值分析专用IDE（如Matlab或RStudio）相比，Spyder有下列特色：开放源代码，以Python编写并且可以兼容于非自由软件许可协议。Spyder可以使用附加组件扩展，内置交互式工具以处理数据。跨平台的特性使得它可以通过Anaconda，Winpython和Python（x,y）（Windows平台）。此外在主流的Linux发行版本例如Ubuntu、Debian、Fedora、OpenSUSE等等中都有它。

1. mysql

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB 公司开发，目前属于 Oracle 旗下产品。MySQL是一种关系数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。MySQL 是最流行的关系型数据库管理系统之一。原因在于具备以下这些优点：

* 基于开源许可发布，无需付费即可使用。
* 自身的功能非常强大，足以匹敌绝大多数功能强大但却价格昂贵的数据库软件。
* 使用业内所熟悉的标准SQL数据库语言。
* 可运行于多个操作系统，支持多种语言，包括 PHP、PERL、C、C++ 、python及 Java 等语言。
* 非常迅速，即使面对大型数据集也毫无滞涩。
* 支持大型数据库，最高可在一个表中容纳 5千多万行。每张表的默认文件大小限制为 4GB，不过如果操作系统支持，你可以将其理论限制增加到 800 万 TB。
* 可以自定义。开源 GPL 许可保证了程序员可以自由修改 MySQL，以便适应各自特殊的开发环境。

# 2.可行性分析和需求分析

## 2.1可行性分析

### 2.1.1技术可行性

技术上使用python语言进行编程，开发工具使用pycharm，spyder，然后还需要使用一些其他的第三方库：numpy，pandas，matplotlib，Tkinter等。在之前自己就已经学习过python有一段时间了，并且也爬取过一些网站，有一定的爬虫经验。而对于numpy，pandas，matplotlib，Tkinter等库也在学习过程中，因此在技术上可以实现该工程。

### 2.1.2经济可行性

python是一门开源的语言，而其开发工具pycharm有Community版本，这时候完全免费的，而且同样能够完成本次开发。而Spyder本身就是一款开源的软件，所以本次开发所需要的开发工具都是免费的。

### 2.1.3操作可行性

由于本功能的独特性，其价值不在软件设计，界面设计等，在于数据分析与可视化，所以该软件的最终实现仅仅只是对分析过后的可视化图形进行展示，所以在操作上很简单。

### 2.1.4法律可行性

该软件的功能仅仅只是帮助人们对于杭州市的租房市场有一个全面的认识，爬取的是公开的数据，而且也不是用于商业上面，也没有用于个人盈利，所以没有任何违背法律的情况。

## **2.2需求分析**

人们在租房的时候需要考虑价格，大小，房间数量，位置，等等。因此，需要将杭州整体的情况展现出来，还要将各个城区，以及各个城区的各小区的具体情况进行展示和对比。

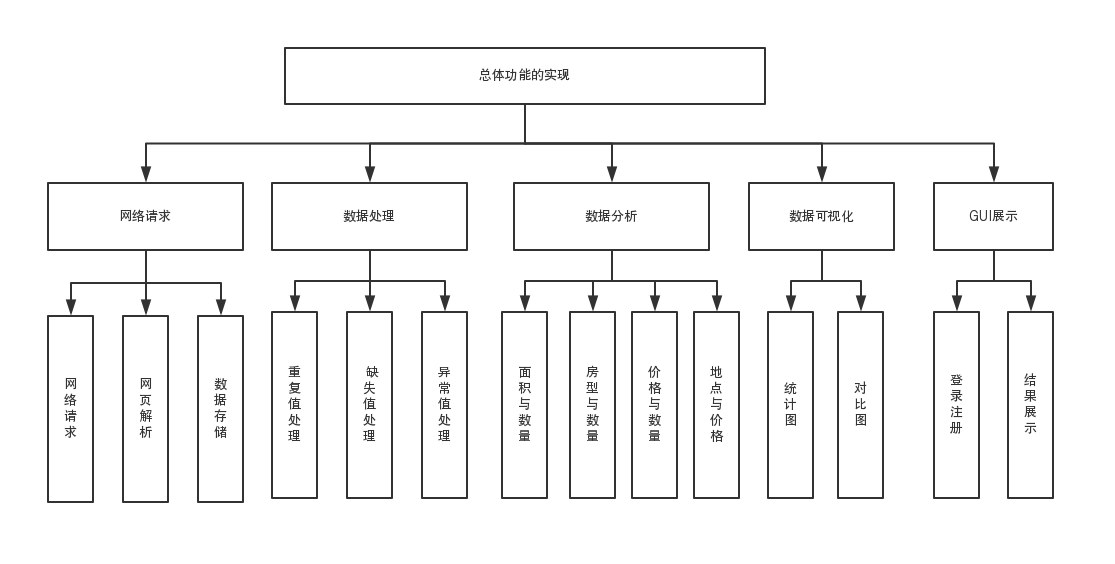
统计图：不同价格区间内房屋数量的统计，不同面积区间内房屋数量的统计，不同房型的房屋数量统计。需要进行三个层次的统计：杭州市整体的统计，各城区内的整体统计，各城区内的小区统计。三个层次的统计，能够让人们对于杭州市的房屋数量分布有一个清晰的了解。

对比图：不同地方的房屋数量对比，同等面积大小，不同地点的房屋数量对比；同样的房型，不同地点的房屋数量对比；同等价格区间，不同地点的数量对比；不同地方的平均价格对比，最高价对比，最低价对比等。对比图也需要不同层次的内的对比：各城区之间进行对比，城区内各小区之间的对比。

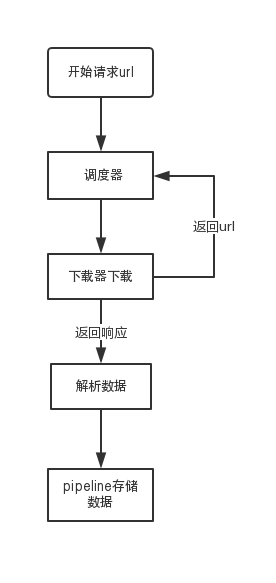
# 系统设计

## 3.1系统总体功能设计

整个系统功能的实现主要分为五大部分：数据爬取部分，数据处理部分，数据分析部分，数据可视化部分和GUI展示部分。这几部分是顺序执行的，必须是在上一部分完成的基础上进行下一部分的实现。



## 3.2爬虫功能实现流程图

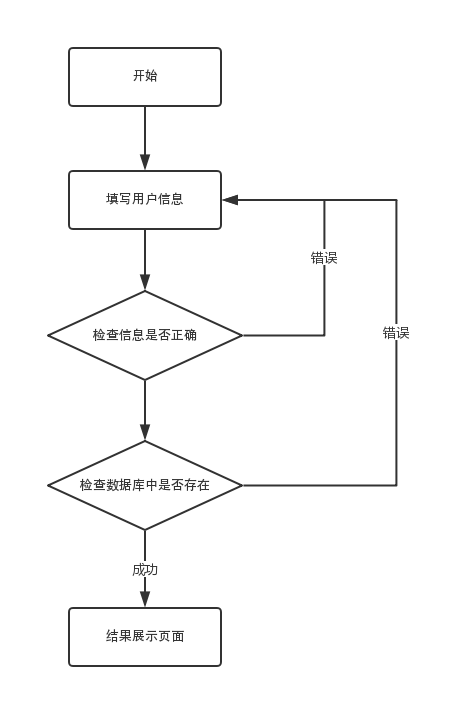


## 3.3GUI设计

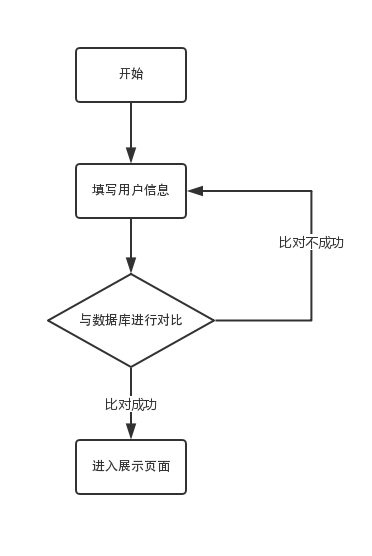
该系统的GUI设计仅仅只是用于对数据可视化后的图形的展示，所以使用的是python的内置的库Tkinter。因此也就没有精美的UI设计，只是简功能的实现。

### 3.3.1登录注册

本系统设置了登录界面，需要用户输入用户名和密码才能进入到可视化展示页面。所以需要先进行注册：



登录流程：



### 3.3.2结果展示

## 3.4数据库设计

# 各部分功能实现

## 4.1数据爬取部分

项目的第一步就是进行数据爬取，因为所有的事情都是建立在对数据进行操作的前提下的，没有数据一切都无从谈起。

### 4.1.1网络请求

网络请求中常用的请求方法是get请求和post请求：

get请求：一般情况下，只从服务器获取数据下来，并不会对服务器资源产生任何影响的时候会使用get请求。

post请求：向服务器发送数据（登录）、上传文件等，会对服务器资源产生影响的时候会使用post请求。

主要请求头参数：

在http协议中，向服务器发送一个请求，数据分为三部分，第一个是把数据放在url中，第二个是把数据放在body中（在post请求中），第三个就是把数据放在head中。这里介绍在网络爬虫中经常会用到的一些请求头参数：

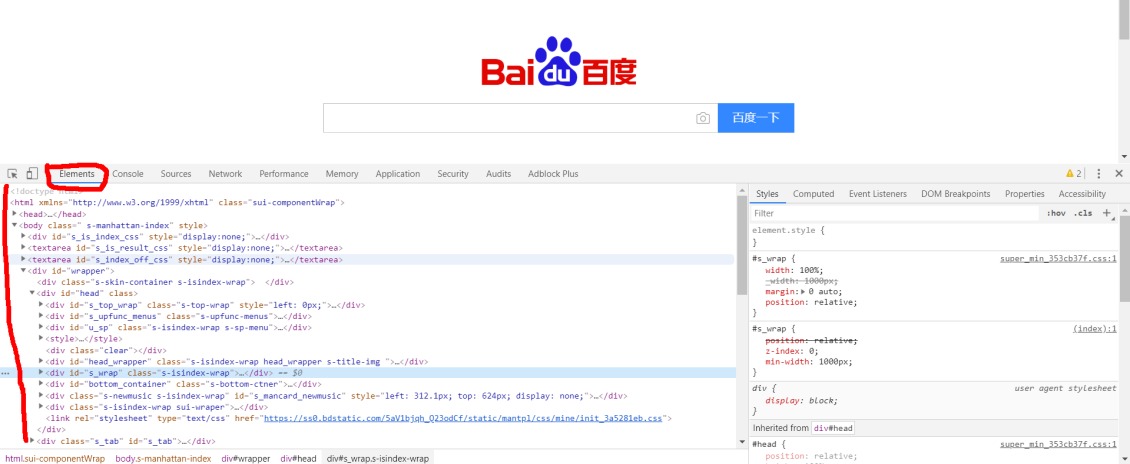
User-Agent：浏览器名称。这个在网络爬虫中经常会被使用到。请求一个网页的时候，服务器通过这个参数就可以知道这个请求是由哪种浏览器发送的。

Referer（引用）：表明当前这个请求是从哪个url过来的。

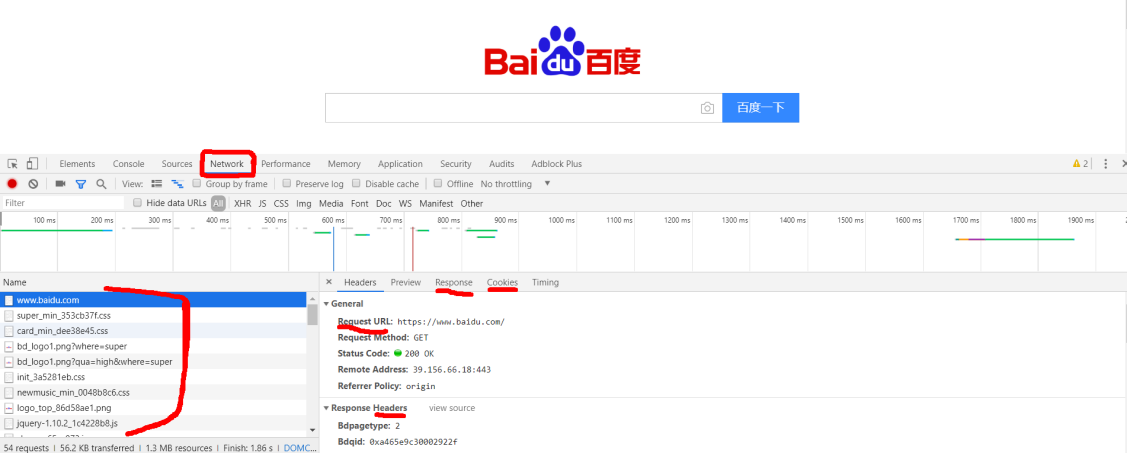
Cookie：http协议是无状态的。也就是同一个人发送了两次请求，服务器没有能力知道这两个请求是否来自同一个人。因此这时候就用cookie来做标识。

Chrome抓包工具：

Chrome浏览器是一个非常亲近开发者的浏览器。可以方便的查看网络请求以及发送的参数。



从上图可以看到，红色划线部分就是百度首页的网页源代码。



上图的Network选项中左边显示的是浏览器加载百度首页时，浏览器发出的所有请求，右边可以看到上文提到的请求头，cookies，其中response选项中是浏览器向百度的服务器发送请求后，服务器返回来的内容。而爬虫程序最主要的就是模拟浏览器的行为请求服务器，获得服务器返回的内容，其中包括需要分析的数据。

请求链家网站：

访问链家杭州租房的网页，同过点击第二页，第三页，可以发现网页url的规律：

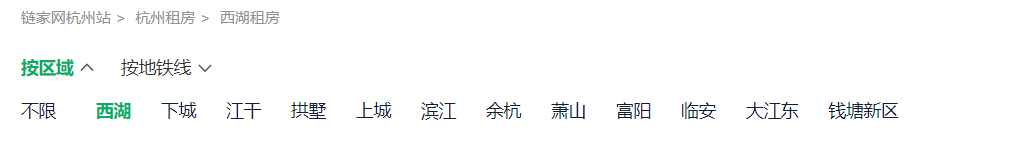
[https://hz.lianjia.com/zufang/xihu/pg1/#contentList](https://hz.lianjia.com/zufang/xihu/pg1/" \l "contentList)

[https://hz.lianjia.com/zufang/xihu/pg2/#contentList](https://hz.lianjia.com/zufang/xihu/pg2/" \l "contentList)

[https://hz.lianjia.com/zufang/xihu/pg3/#contentList](https://hz.lianjia.com/zufang/xihu/pg3/" \l "contentList)

通过测试#及其之后的类容可以省略，因此可以看到，网页url有三部分组成：基地址，城区，页数。所以如果想访问不同的城区可以修改对应参数，访问所有页面，同样可以修改对应页数。

通过网站可以看到杭州市有如下12个区组成：



点击每个区，可以看到每个区的房屋页数也不相同，因此，要想通过修改url对应的参数来访问杭州市所有的租房页面，就需要解决如何正确的写出参数改变方法，并且尽可能的简单。

最方便的方法就是使用crawlspider类爬虫，因为该类的爬虫可以自定义爬取的url规则，通过上面的分析可以看到，url有着明显的规则，只需要使用正则表达式表示出该规则就可以爬取整站的杭州租房页面。但是通过分析发现，该网站上的翻页功能是由js技术动态实现的，无法使用crawlspider类爬虫。

所以只能使用较为粗暴的方法，使用for循环来改变页数。每个城区写一个for循环，这样需要12个for循环，好在不需要太多代码，每个for循环仅仅只有两句一句是得到新的url，一句就是访问该url：

for page in range(2, 101):

url= 'https://hz.lianjia.com/zufang/xihu/pg%s/#contentList'%page

yield scrapy.Request(url=url, callback=self.parse\_fang)

### 4.1.2网页解析

当发送网络请求后，scrapy框架会将请求交给调度器（Scheduler），然后引擎从中提取请求，交给下载器进行下载，下载器将网页内容下载下来，并返回一个响应，这时就需要对响应中的数据进行解析，得到我们想要的数据。那么如何解析呢？其实可以通过浏览器的开发者工具查看网页源代码，下载器下载的就是这些内容。



通过上图可以看到，当点击下方源代码时，在上方网页中就会同时显示对应的网页内容，因此我们可以通过这种方式很方便的弄清楚网页的结构，知道应该怎么解析。

那么这么多的数据，哪些是我们需要用到的呢？



上图画红线的部分就是我们需要的信息，包括房屋链接，地址，大小，房屋类型，价格，发布时间，是否近地铁等。点击该房屋链接进入详情页面可以看到该房屋的房源编号和房屋链接中的后部分字母数字是一样的，因此也可以通过房屋链接解析出房源的编号，这是房屋的唯一识别。通过房源编号可以确定房屋是否重复。

那么如何获取上面网页中显示的信息呢，有三种方法可以实现：

1、使用Beautiful Soup 4库进行解析

2、使用xpath工具进行解析

3、使用正则表达式进行解析

其中正则表达式速度最快，但却最难，Beautiful Soup 4最简单，却比较慢，而xpath简单，速度也相对较快，个人使用又较为熟练，因此选择xpath进行解析。

fang\_addr = **"-"**.join(fanginfo\_div.xpath(**".//p[@class='content\_\_list--item--des']//a/text()"**).getall())

fang\_price = int(fanginfo\_div.xpath(**".//span[@class='content\_\_list--item-price']/em/text()"**).get())

上面是通过xpath解析得到房屋的地址和价格，可以看到，xpath就是通过网页信息在源码中的路劲得到该信息。因此可以和浏览器的开发者工具结合，十分方便的进行解析。

### 4.1.3数据存储

通过xpath解析得到的数据是如何进行存储的？在前面就已经讲过，scrapy框架对于爬虫的分工是十分明确的，每个部分该干什么就干什么。那么就是应该在pipelines.py文件中进行数据存储，可是spider.py中解析得到的数据怎么送到pipeline也就是管道中的？

在这里需要说到scrapy框架是异步进行的，也就是说，它从调度器中提取的url不一定是按照原来的顺序进行的。这样在进行数据存储的时候就需要注意，怎么保证将数据正确的传过去。

好在scrapy框架设计了一个items.py，将需要传递的信息写在该文件中，在上一步中一条数据解析完成后，将该条所有数据通过items中的类，将这些数据一起传给管道（pipeline），这样就可以保证数据的完整性和正确性。

item = HangzhouZufangItem(fang\_id=fang\_id, fang\_url=fang\_url, fang\_addr=fang\_addr,fang\_area=fang\_area,fang\_homes=fang\_homes,

fang\_date=fang\_date,fang\_benefit=fang\_benefit, fang\_price=fang\_price)

yield item

在pipelines.py中需要将数据存储为json格式的文件中，以及数据库中。存入json格式中是为了将其转换为csv文件，在数据处理时更方便，快捷，而从数据库中提取数据相对较为慢些。之所以没有直接存储为csv格式是因为可以使用JsonLinesItemExporter类进行存储。存入数据库是为了可以更好的保存数据。

存入json格式文件也是有不同的方法的：

就是直接使用一般的写入json文件的方法。

使用JsonLinesItemExporter类进行写入，使用这个类的好处是每次没传入一次数据就直接存储到硬盘中，这样不会耗内存，数据也较为安全。当然坏处就是每个字典时一行，整个文件不是一个满足json格式的文件，但是可以通过在打开文件和关闭文件时添加一句话就可以解决：

self.fang\_fp.write(b"[")

self.fang\_fp.close()

存入数据库也有两种方式，这里选择的是mysql数据库：

同样就是一般的方法，该方法是采用同步的方式进行存储。

另一种是采用异步的方式进行存储，使用scrapy框架自带的异步功能，通过TwistedPipeline中间件，可以异步的将数据存入数据库中，提高了爬虫数据存储方面的速度。

self.\_sql="insert into article(id,title,content,author,avatar,pub\_time," \ "origin\_url,article\_id,read\_count,like\_count,word\_count,subjects,comment\_count) values (null,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"

上面的是需要执行的sql语句。在调用该语句时添加参数可以动态的插入数据。

## 4.2数据处理

### 4.2.1将json数据转换为csv格式

将json格式的数据转为csv格式是十分方便也十分快捷的，核心的代码如下：

data\_list = json.load(json\_fp)

sheet\_title = data\_list[0].keys() # 表头

print(sheet\_title)

sheet\_data = []

for data in data\_list: # 表内容

sheet\_data.append(data.values())

### 4.2.2使用pandas读取数据

对于数据的处理使用的是numpy和pandas这两个库，这两个库处理数据实在是太方便了。特别是pandas的两大数据结构：Series和DataFrame对象。可以很好的处理csv这样的表格数据。

首先pandas对于读取和写入各种格式的文件都有对应的接口，只需要一句话就可以快速的读取文件内容或者写入数据到文件，以csv文件为例：

读取csv文件：

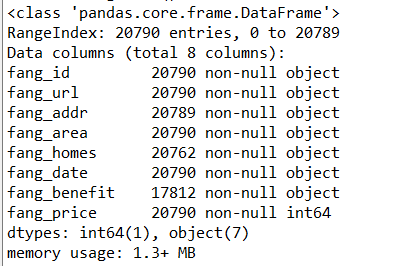
pd.read\_csv('G:/python\_study/hangzhou\_zufang\_1/csv\_file/hangzhou\_zufang.csv')

将处理后的数据写入csv格式的文件：

fang\_drop\_dup\_dropna.to\_csv('G:/python\_study/hangzhou\_zufang\_1/csv\_file/fang\_dealed\_all.csv', index=False)

在读取原始数据之后，通过调用两个函数来查看该数据的一些基本信息：

fang.info() ：显示数据的详细的统计信息



上图的结果能够让我们对于如何处理和分析数据有一个较为清晰的方向。

### 4.2.3处理重复值

通过调用函数：len(fang.fang\_id.unique())，可以查看该数据是否有重复值，如果返回的数值小于原始数据的行数，则说明有重复值。需要说明的是，在这里是以房屋的编号也就是fang\_id作为唯一判断标准，当fang\_id有重复的时候，变认定有重复值。

使用函数：fang\_drop\_dup = fang.drop\_duplicates(subset="fang\_id", keep='first')

删除重复值，同样需要指定以fang\_id为基准进行删除。

通过函数：len(fang\_drop\_dup)，可以检验到重复值已经删除。

### 4.2.4处理缺失值

通过函数fang\_drop\_dup.isnull().sum()，可以看到数据的每一列是否有缺失值，有多少缺失值，该函数都进行了详细的统计。

使用语句：fang\_drop\_dup[fang\_drop\_dup.isnull().values==True]，还可以查看具体的缺失值对应的详细数据信息。

使用函数：

fang\_drop\_dup\_dropna=fang\_drop\_dup.dropna(subset=['fang\_addr','fang\_homes'])，可以删除缺失值所对应的数据，有些缺失值可以进行填充，但有些缺失值无法进行有用的填充，所以需要删除。同样也需要指定处理的是哪一列的缺失值，在这里需要处理的是房屋对应的地址的问题。将有地址缺失的数据进行删除。

### 4.2.5处理异常值

在进行数据分析的时候，有些数据之间需要进行数字运算，此时就需要该数据类型为int类型。通过fang.describe()函数可以看到，需要数字运算的有两列：面积（fang\_area）以及价格（fang\_price），其中fang\_price列是int类型，但是fang\_area列并不是，通过查看，可以发现是因为它的数值后面带了单位。因此现在需要做的是去掉单位。

def str\_to\_int(data, str):

if str in data:

return float(data[0: data.find(str)])

该函数是对fang\_area列的数据进行处理，返回去掉单位的float类型的数值。

pandas有一个函数可以快速的让上面的函数处理一些数据，而不需要通过for循环等方式进行：

fang\_drop\_dup\_dropna['fang\_area'].apply(str\_to\_int, str='㎡')

pandas.apply()这个函数很强大，它可以让pandas数据去执行传入的函数，最重要的是传入的函数可以自己制定，也就是说，想怎么处理这些数据就自己写什么样的函数，然后只需要同apply.()函数就可以对数据进行处理。

还有一列的数据需要处理，通过csv文件的查看，以及我们需要分析的数据，发现地址这一列是以“城区—小区/地段”的方式给出的。在后面的分析中，我们还需要对小区/地段进行分析，所以需要将这一列进行拆分。

fang\_drop\_dup\_dropna['fang\_urban']=fang\_drop\_dup\_dropna['fang\_addr'].str.split('-').apply(lambda x: x[0])

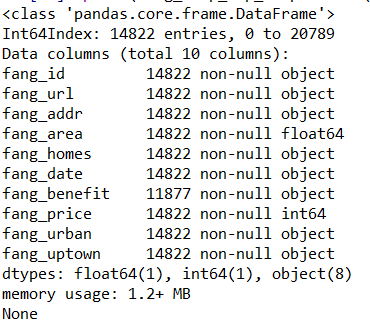
fang\_drop\_dup\_dropna['fang\_uptown']=fang\_drop\_dup\_dropna['fang\_addr'].str.split('-').apply(lambda x: x[1])

通过上面两句话就可以将地址列拆分为两列，而原来的地址列可以删除了，因为后面的分析不需要这一列了。

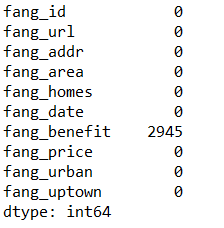
### 4.2.6检验处理结果

至此，数据的处理就结束了，处理完后还需要进行检验，看看是否真的全部处理完了：

print(fang\_drop\_dup\_dropna.info())



print(fang\_drop\_dup\_dropna.isnull().sum())



可以看到除了fang\_benefit这一列外，其他列都没有缺失值了，而fang\_benefit这一列暂时用不上，所以可以不进行处理。

## 4.3数据分析部分

当数据处理完成后，需要进行的就是数据分析了，这是最为关键也是最为复杂的部分。那么怎样进行分析？这取决于我们想要实现什么样的功能。本设计的目的是为了让人们对杭州市的租房市场有一个全面的，详细的了解。那么就需要先对杭州市的租房数量进行统计，大概有多少房子，又分布在什么地方，因此还要统计各个城区的数量。而我们最终租房子还是要确定在哪个小区或者路段，因此还要对该层次的房屋进行统计。

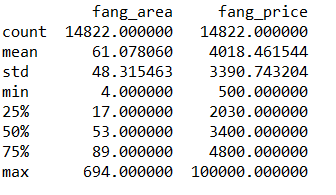
统计是为了对杭州市的租房市场有全局的掌握，那么对比图就是为我们选择房子提供参考了。

### 4.3.1统计图

杭州市整体统计

该统计是在整体上了解杭州市的租房市场，了解杭州市的租房都是分布在什么价位，主要都是多大的面积，以什么房型为主。

通过函数：fang\_dealed\_all.describe()，可以看到在面积和价格方面的一些数学数值。



通过结果可以将杭州市的租房市场的价格和面积区间进行分段：

将价格分为12个区间：

['<1000', '1000-2000', '2000-3000', '3000-4000', '4000-5000', '5000-6000', '6000-7000', '7000-8000','8000-9000', '9000-10000', '10000-15000', '>15000']

面积分为12个区间：

['<10', '10-20', '20-30', '30-40', '40-50', '50-70', '70-90', '90-110', '110-150', '150-200', '>200']

而同过对房型的数量统计，可以分析数量在120间往上的，太小数量的没有太大参考意义，恰好也是12种房型：

['2室1厅1卫', '1室1厅1卫', '4室1厅2卫', '3室1厅1卫', '3室2厅2卫', '4室1厅1卫', '2室2厅1卫', '3室2厅1卫', '5室1厅2卫', '3室1厅2卫', '4室2厅2卫', '1室0厅1卫']

各城区的统计

对各城区也进行上面几个方向的统计，让人们能够对自己所在的城区有一个清晰的了解，毕竟杭州市整体的了解只是一个大的方向的把握，对整体租房市场的了解，但是实际中还是需要更关心自己所在的城区的详细情况。

因为有12个城区，所以不可能每个城区写一段代码，这样工作量太大，最主要的是代码太过于臃肿，运行速度更是很慢。因此需要对上面统计价格区间，面积区间，房型的方法进行重构，编写成函数，那么在进行城区统计的时候，只需要使用for循环遍历城区列表，调用相关函数，传递相应的参数，就可以快速实现12个城区的统计。

for item in fang\_urban\_list:

fang\_urban\_frame = fang\_dealed\_all[fang\_dealed\_all['fang\_urban'].values == '%s'%item]

X, Y = statistic\_func.price\_interval(fang\_urban\_frame)

上面的代码是统计各城区的价格区间分布的函数调用，其他面积区间和房型的函数调用也是类似的情况。

各城区内各小区的统计

上面对城区的统计让我们对该城区有了较为详细的了解，那么到底应该选择哪个小区呢？因为现在交通比较方便，所以可以更好的考虑性价比的问题，不一定非要找工作地点附近的小区。在做选择之前还是同样的对该城区的每个小区或不同路段的租房情况有所了解，同样的是了解该小区都是什么价位的房子，多大面积，什么房型为主，是否符合自己的期望。

统计方法使用的还是上面的函数，但是相较于各城区的统计，各小区的统计又要多加一个for循环，用来循环该城区内的小区列表，因此代码的逻辑就更深一层：

for item in fang\_urban\_list:

# 得到该城区的小区列表

fang\_urban\_frame = fang\_dealed\_all[fang\_dealed\_all['fang\_urban'].values == '%s'%item] fang\_uptown\_list=pd.value\_counts(fang\_urban\_frame['fang\_uptown']).index.tolist()

for uptown\_item in fang\_uptown\_list: fang\_uptown\_frame=fang\_urban\_frame[fang\_urban\_frame['fang\_uptown'].values == '%s'%uptown\_item]

X, Y = statistic\_func.price\_interval(fang\_uptown\_frame)

上面的代码就是各小区的价格区间分布图，需要先遍历该城区的小区列表，然后再遍历城区列表。虽然这样会让程序的运行时间变长，但是却可以很方便的得到各个城区内的各个小区的统计图。要知道总共有100多个小区，只有使用这种方法才是最快速的。

### 4.3.2对比图

统计图是为了在大方面有所了解，整个杭州市的各个层次的租房市场有一定的全面的了解，这样对整体的把握之后，经过对比就可以有一定的判断结果出来。那么对比哪些内容呢？

就是根据之前的统计结果，在相同的价格，面积，房型下，各个地点的房屋数量对比，之所以是数量的对比是为了可以与之前各个地点的统计量进行参考，之前的统计量可以知道该地的价格，面积，房型都集中在什么区间，那么在这同样的区间下，各个地方的数量对比可以让人根据自己的喜好进行选择。

比如对于价格的要求在1000-2000之间，那么就可以对比在该区间内哪个地方的房子数量较多，因为同等区间它的数量最多，那么可供选择的余地就相对比较多。

各城区之间进行对比

通过各个城区之间的对比，可以看到哪个城区更适合自己的选择，这种对比适合那些工作地点距离附近的几个城区都不是特别远的地方。毕竟我们选择租房首选的肯定还是尽可能的的距离工作地点近的地方。

在之前的分析中将价格，面积，房型都分成了12个区间，因此，在对比时就需要12区间的遍历。虽然是对比图，但是本质上还是统计。因此需要对代码进行分装，写成函数的形式，然后通过遍历12个区间，调用函数来达到快速的到12个区间各自的对比结果。

for min\_value, max\_value in zip(min\_price, max\_price):

X,Y=compare\_func.urban\_same\_price\_graph(fang\_dealed\_all,

min\_value,max\_value)

上面的代码就是通过循环遍历12个价格区间，调用函数来快速得到在这12个价格区间内各城区的统计量。

各城区内的各小区之间的对比

与统计图中的一样，因为需要遍历各个城区内的各个小区，所以相较于城区之间的对比，又要添加一层for循环嵌套。而原本的统计函数中也有一层嵌套，所以该功能的实现有三层for循环，运行可能较慢。但是却是很方便的就得到所有的结果，在程序设计上节省了大量的时间和代码量。

在此处也不需要对代码进行一些讲解，因为在逻辑上和之前的一样，需要注意的是不能和统计图中与城区的统计函数也是一样的。在这里不是的，因为参考值不同，所以需要另外的函数，但也是对城区之间进行对比的调用函数进行简单的修改，但这却是必须的。

## 4.4数据可视化部分

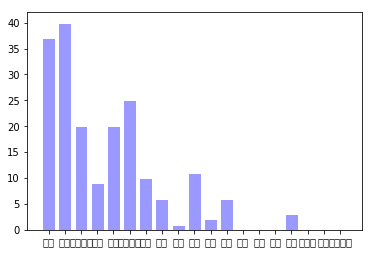
数据的可视化使用matplotlib库进行绘制的，前文介绍的该库的一般认知。而对于本设计由于需要以最直观的方式将结果展现给用户，所以采用了最为直观的柱状图来进行展示。

柱状图的实现其实只是两句话的事情：

plt.bar(X, Y, facecolor='#9999ff', edgecolor='white')

plt.show()

只需要给该函数提供x轴坐标，y轴坐标，柱状图颜色，背景色等基本信息，通过plt.show()函数进行展示。一个柱状图就展现出来了。



这就是只是 使用上面两句代码实现的一张图，可以看到在x轴的标签中无法显示汉字，没有标题，每个bar对应的也没有详细的数据显示，从图中也只能看到一个大概的数字。

所以还需要对上面的代码进行较大的改进。首先是要保证能够正常的显示中文，因此需要添加代码：

plt.rcParams['font.sans-serif'] = ['SimHei']

plt.rcParams['axes.unicode\_minus'] = False

这两行代码实现了汉字的正常的显示。下一步为整个图形添加标题，还要为x轴和y轴添加标题，以使用户能够一眼便知道改图的具体意义：

plt.title('价格区间统计图', fontsize=20)

plt.xlabel('价格区间(元)', fontsize=20)

plt.ylabel('数量', fontsize=20)

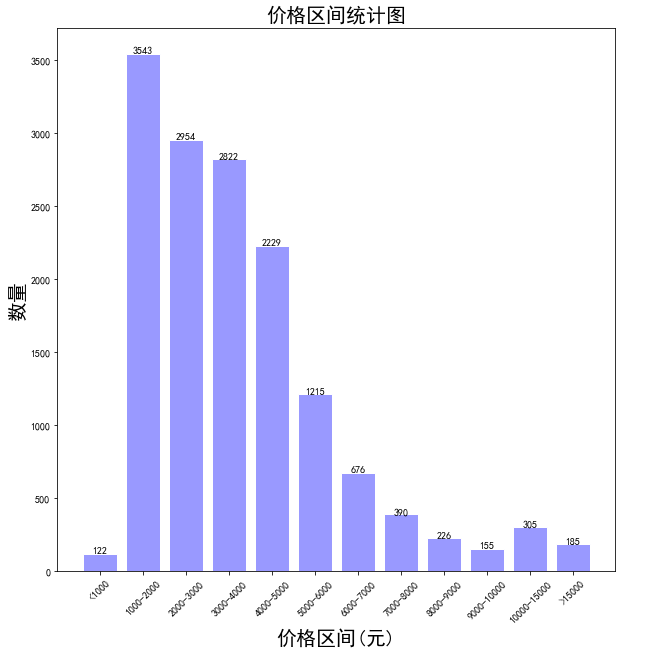
为每一个bar添加具体数据说明：

for x,y in zip(uptown\_list,counts):

plt.text(x , y + 3, '%d'%y, ha='center', va='top')

因为每个图形肯定都有多个bar，因此需要遍历这些数字，为每个bar添加数据说明。

除了上面的主要的改进之外，生成的图形可以发现还是有一些其他的问题，比如标题显示不全，x轴标签的文字叠加等。因此还需要为该图形设置一个画布用来装图形的容器，可以设置其大小。通过一些参数的设置是x轴的标签旋转45度，可以让x轴标签斜着显示，刚好错开文字，不会产生重叠。



可以看到上图就是一个元素较为完全的柱状图。

# 致谢

在此我谨向我的指导教师\*\*\*教授表示最衷心的感谢！在我学习期间,李老师为我们创造了良好的学习环境，他渊博的学识、严谨的治学态度、对科学敏锐的洞察力和极强的预见性，使我受益匪浅。

# 参考文献

[1] Shamir A.How to share a secret[C].Communications of the ACM,1979.24(11):612-613

[2] 卢开澄.《计算机密码学》[M].北京:清华大学出版社.1998.56-190

\*\*\*

# 附录