《Python 程序设计综合训练》

实验报告

班级：软工 2104

姓名：凌晨

学号：2214414320

目录

一 实验

1 实验内容简介

利用 python 综合训练学习到的部分知识尝试多种简易的西安二手房估价模型。主要内容如下：

1. 利用爬虫爬取西安二手房数据，爬取网站为

https://xa.lianjia.com/ershoufang/pg2co32/

2. 利用办公自动化清洗数据，对定类数据进行编码，方便下一步模型建立分析。

3. 利用办公自动化生成相关词云，利用可视化生成各种图表，直观简单地展示数据特点。

4. 利用科学计算相关知识，尝试建立多种简易较为有效的西安二手房估价模型。

5. 对建立的模型进行分析，给出优点缺点。

2 爬取数据

2 西安二手房数据爬取

2.1 网站的确立

一个好的网站不仅数据全面，而且利于 python 进行爬取。

该任务为爬取西安二手房价数据，首先确定需要爬取的网站和数据，这是十分重要的。

经过筛选，选取了链家网站的数据，搜索结果如下（部分截图）：

该网站有如下优点：

1. 该网站涵盖的数据量巨大，一共含有 111774 个数据，尽管爬取部分数据，其全面可靠也足够满足建模分析。

2. 该网站对于房子的各个特征归纳总结到位，便于分析。

3. 该网站便于数据爬取。

2.2 爬取方式的确定

该网站已经整理完毕，并且分类完毕，因此可以选取部分所需数据爬取。

1. 进行网站构成分析：

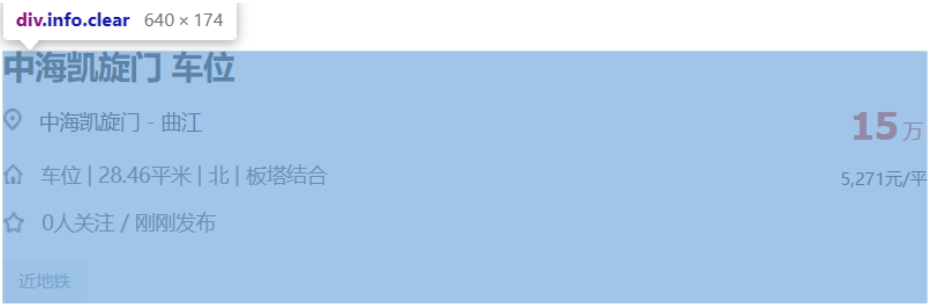
一共有 100 页数据，其中网页链接构成如下：

https://xa.lianjia.com/ershoufang/pg2co32/

构成十分有规律，当进行翻页时，pg 后面的数字加 1 即可，十分利于爬取，利用一个变量表示页数即可，细节不再赘述。

2. 对网页构成分析：发现包含数据的网页不是动态网页，而是静态网页，直接调用 request 库即可进行访问就可以得到所有所需数据了。

3. 对数据包含分析：调用 F12 查看元素可以发现：



所需的数据主要包含在*div.info.clear*中，可以确定以下几种爬取方式：

1. 使用 re 正则匹配式进行查找匹配。该方法直接有效但是容易爬取错误。

2. 对于 html 语法树进行匹配，找到对应容器，提取文字即可。该方法不仅简单有效，而且由于该网站十分规范，不会出现爬取错误数据。

2.3 实战

综上所述，采取了使用**lxml包中html语法树**进行匹配爬取数据的方式。

核心代码如下：

```
1.  hl = html.etree.HTML(resp.text)
2.  con_1 = hl.xpath("/html/body/div[4]/div[1]/ul/li")
3.  for t in con_1:
4.      name = t.xpath("./div[1]/div[2]/div/a[1]/text()")
5.      local = t.xpath("./div[1]/div[2]/div/a[2]/text()")
6.      detail = t.xpath("./div[1]/div[3]/div/text()")
7.      price = t.xpath("./div[1]/div[6]/div[2]/span/text()")
```

不再赘述，核心思想就是找到容器，爬取文本。

2.4 爬取的改进

1. 加入线程池加速爬取速度

可以知道，如果我们要爬取 100 页，一页一页爬取速度会很慢，因此使用线程池可以大大加速爬取过程，核心代码如下：

```
1.  with ThreadPoolExecutor(5) as t:
2.          for i in range(1, 101):
3.              t.submit(get_one_page, f"https://xa.lianjia.com/ershou
    fang/pg{i}/")
```

加入线程池后，爬取速度大幅加快，不过出现了 ip 被网站标记，禁止登陆爬取的问题，最后只爬取到了不到 200 条数据。

2. 想办法绕过 ip 封锁

不过这肯定难不倒我，解决办法如下：

访问网页加入 header。

使用*time.sleep()*缓解了一下我和网页的关系。

代码不再给出，详情可见附录。

2.5 爬取结果及展示

最后通过优化，爬取到了足足 1500 条以上的数据，下面展示部分数据：

名称	地址	房型	面积（㎡）	朝向	装修	楼层	版型	单价(元/㎡)
水榭花都	高陵	1室1厅	22.21	北	简装	15	板楼	8,015
高新枫尚	太白南路	1室0厅	23.29	东	简装	24	塔楼	22,757
爱尚泾渭	高陵	1室1厅	26.11	西北	毛坯	16	板塔结合	7,201
斯惟小区	经开北	1室1厅	29.16	北	简装	18	塔楼	10,460
御笔华庭	城西	1室0厅	29.59	东	简装	32	板塔结合	11,998
翠屏湾二期	矿山路	1室1厅	29.95	北	简装	20	板塔结合	17,363
翠屏湾二期	矿山路	1室1厅	29.95	北	简装	20	板塔结合	17,363
水榭花都	高陵	2室1厅	32.01	南	精装	15	板楼	9,654
东方罗马花园	矿山路	2室1厅	32.18	北	精装	16	塔楼	17,092
罗马嘉园	广泰门	2室1厅	32.5	南	精装	16	塔楼	23,047

可以看到数据是非常喜人的。

3 可视化实验

3.1 词云

词云有助于我们对文字形成直观清晰的认知，一方面有利于我们进行定量转换，另一方面有助于我们进行判断数据好坏程度。

3.1.1 数据的选择

理论上所有数据都可以生成词云，但是我选择了二手房地址数据生成词云，原因如下：

1. 地址不好定量转换，最先讨论较好。
2. 地址是纯文字，生成词云效果更佳。

3.1.2 代码编写

代码与课堂教学并无过多出入，不再展示，可见附录。

3.1.3 词云展示

可以看到，重点分布在住宅区得多，下面是部分统计分析。

城西	10.17%
经开北	7.90%
城南	7.84%
曲江	6.46%
长安	6.19%
城北	5.91%
高陵	5.91%
经开南	5.09%
城东	4.95%
浐河西路	4.05%
高新六路	3.09%
安装四处	2.89%
城内	2.13%
西咸	2.06%

不难得出结论：房子地址也是一个需要考虑的要素，不同地址房子数量分布不同，进而导致房价波动。

3.2 可视化表格

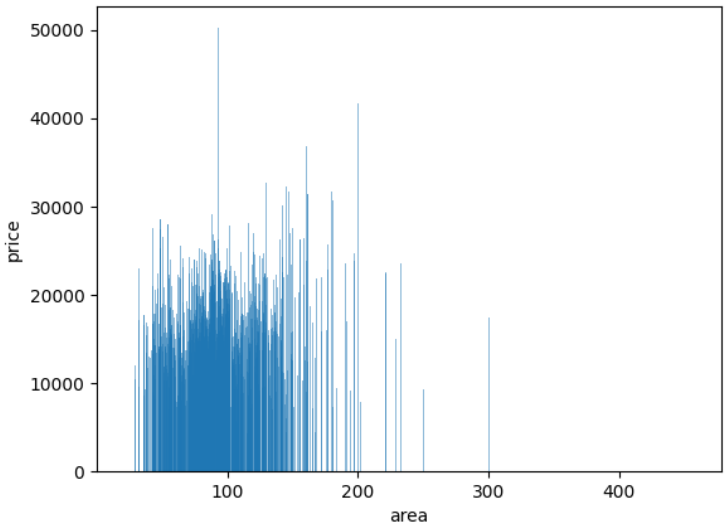
干巴巴的数字是没有生命力的，但是如果利用各种表格，比如柱状图，折线图，不仅可以让使用者对数据有清楚的认知，还可以快速找到需要数据的范围。因此我展开了该实验。

3.2.1 确定图像

想要初步直观观察各个数据的关系，柱状图是比较好的选择。

3.2.2 绘图

与课件代码类似，详情见附录，图像如下：



面积/房价柱状图

3.2.3 图像分析

通过观察图像可以得到以下结论：

1.大部分房子的面积分布在 50~100 平米之间。

2.90 平左右的房子单价高于附近面积的房子

4 数据清洗及编码

4.1 数据清洗

利用 python 办公自动化，进行了如下情况清洗：

1.装修为“其他”

2.版型为“暂无数据”

3.楼层为“空”

4.单价为“空”

5.其它不符合标准的数据

代码机械重复性工作较多，不再给出，附录也不再给出。

4.2 清洗结果

无法全部展示，但是通过 excel 自带工具可以看到全为有效数据。

4.3 数据编码

进行了如下编码：

1.对装修情况进行编码，编码为 1，2，3…

2.对房型进行编码，编码为 1，2，3…

3.对版型进行编码，编码为 1，2，3…

4.对楼型进行三分位数编码，编码为 1，2，3…

5.对地址进行编码，编码为 1，2，3…

6.对朝向进行编码，编码为 1，2，3…

得到数据部分展示：

房型_数字编码	版型_数字编码	楼层_三分位数分组	装修_数字编码	名称	地址	房型	面积（m^2）	朝向	装修	楼层	版型	单价(元/m^2)
1	1	1	2	1	1	1	22.21	1	1	15	1	8015
2	2	2	2	2	2	2	23.29	2	1	24	2	22757
1	3	1	1	3	1	1	26.11	3	2	16	3	7201
1	2	2	2	4	3	1	29.16	1	1	18	2	10460
2	3	3	2	5	4	2	29.59	2	1	32	3	11998
1	3	2	2	6	5	1	29.95	1	1	20	3	17363
1	3	2	2	6	5	1	29.95	1	1	20	3	17363
3	1	1	3	1	1	3	32.01	4	3	15	1	9654
3	2	1	3	7	5	3	32.18	1	3	16	2	17092
3	2	1	3	8	6	3	32.5	4	3	16	2	23047

4.4 编码优点及缺点

几乎是不假思索的对所有定量进行了数值化编码，带来了建立模型和分析方便的好处，但是没有更进一步地思索数据的内在关联，甚至只是粗暴地通过整数定义。希望未来学习完数学建模可以更进一步地进行改善。

5 基于线性回归模型_尝试 1

5.1 模型的简述

把每个变量都是为线性影响房价，然后根据最小二乘拟合法可以建立起来一个简单的线性模型。代码主要是根据上课所学。

5.2 符号,公式说明

由于地址变量还是不好量化，暂且不考虑。

定义各个变量：

y	单价

x_i	各个变量

公式如下：

$$Y = \sum K_i X_i + b$$

5.3 代码

与 PPT 上的相差无几，就是多录入几个变量。核心代码如下：

```
1. def residuals(p):
2.     k0, k1, k2, k3, k4, k5, b = p
3.     return y -
    (k0 * x[0] + k1 * x[1] + k2 * x[2] + k3 * x[3] + k4 * x[4] + k4
   * x[5] + b)
4. r = optimize.leastsq(residuals, [1, 1, 1, 1, 1, 1, 0])
5. print(r[0])
```

得到的参数如下：

$$[5.98234546e+02\ \ 2.03494731e+02\ \ 1.44985594e+03\ \ 2.12034725e+03$$
$$-1.28752421e+00\ \ 1.00000000e+00\ \ 3.81468992e+03]$$

从左到右分别代表$\boldsymbol{k0 - k5}$和$\boldsymbol{b}$

简易的线性回归模型建立完毕。

5.4 分析及评估

由初始值和最终参数可知，前四个变量即楼型，版型，楼层，装修提升了数量级，而面积是负相关，朝向的影响微乎其微。

得到以下结论：

1. 房价对楼型，版型，楼层，装修敏感。
2. 房价对朝向不敏感
3. 房价与面积呈现反相关。

优点：

1. 模型简单有效地评估西安二手房价与其它几个变量之间的线性关系。
2. 可以反映房价的平均值。

缺点：

1. 现实是房价与几个变量不一定呈现线性关系，因此模型并不适用房价。
2. 模型与我的编码有极大关系，比如朝向通过编码无法体现线性关系。
3. 模型还未考虑地址问题。
4. 存在多重共线性问题。

6 基于 adaboost 回归模型_尝试 2

6.1 模型简述

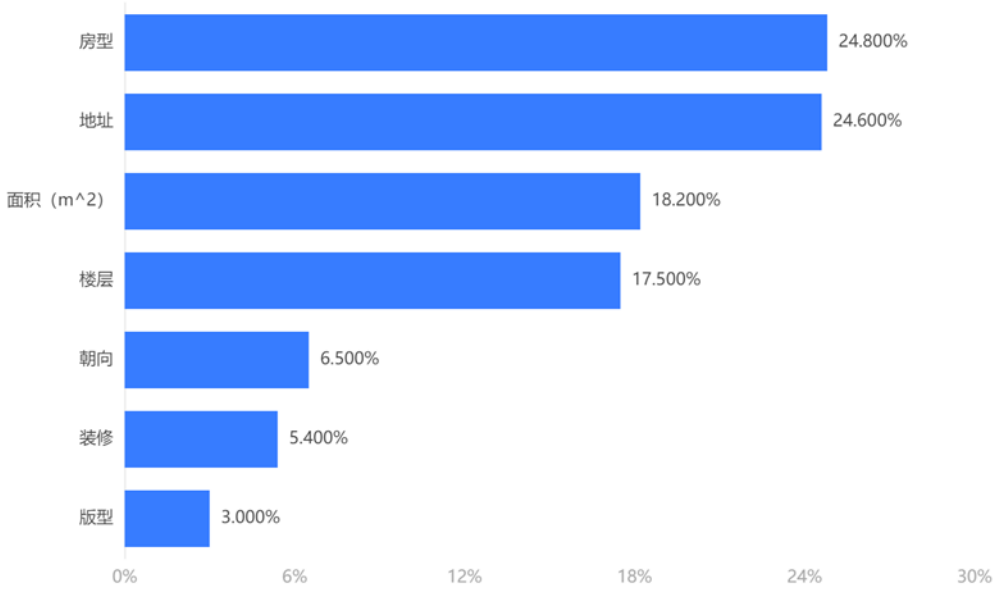
我在网上寻找了其它有效的好用的房价预测模型，发现了机器回归模型中的*adaboost 回归模型*。

关于模型的介绍不再赘述，因为我也尚未完全掌握。不过，利用该模型我可以进行定量和定性的分析。

6.2 模型结果

得到的结果如下：

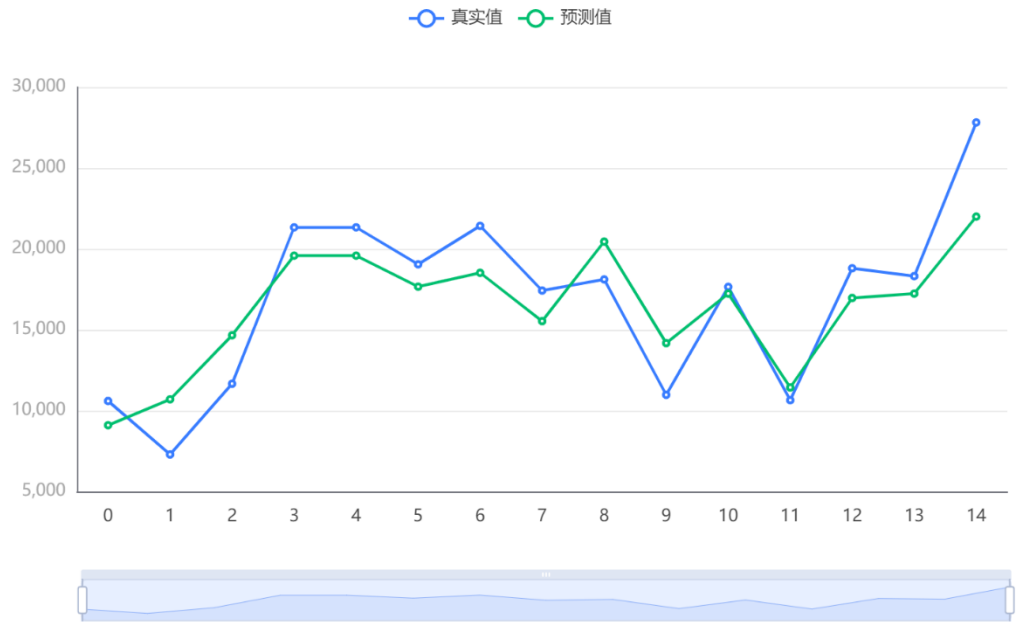
1. 特征值重要性



2. 模型评估结果

	MSE	RMSE	MAE	MAPE	R^2
训练集	31619.944	177.82	45.727	0.427	0.999
交叉验证集	16682142.1	3994.761	2561.437	18.03	0.281
测试集	73113930.9	8550.668	4949.89	23.185	-1.09

3. 预测图



6.3 模型分析

通过上述分析，可以得到以下结论：

1. 模型在训练集中表现较好，但测试集中，预测房价的精准度有待提高。

2. 影响房价较大的因素表现为房型，地址，面积和楼层。这里可以看到，地址的作用体现出来了。而在线性模型中较为重要的装修，版型在该模型中不再被“重视”。

3. 在图中可以看到，虽然预测存在误差，但总体趋势保持一致，与线性模

型相比算是巨大进步。

二、实验过程中遇到的问题和解决过程

1 各种包安装的问题

1.1 包安装速度过慢

利用国内镜像源解决，代码如下：

$$pip\ install\ -i\ https://pypi.tuna.tsinghua.edu.cn/simple\ some-package$$

1.2 包安装完毕后无法使用

经过我漫长地探寻和长时间百度，终于发现了问题，问题如下：

因为我安装了多个版本的 python，有 3.9 也有 3.10，每个版本安装的盘不一样，然后电脑系统以我最新安装的 python 为系统环境了，所有的包安装到了 D 盘但是我使用的编译器是 C 盘里面的，它只能调用 C 盘里面的包，我只需要改变电脑关于 python 编译器的系统环境就好了，具体解决办法如上。

1.3 包的调用出问题

是因为我下载的是最新版本，而原有的版本书写规则跟不上了，学习了新的书写规则就好了。

2 模型构建遇到的问题

线性模型的构建没有遇到说什么问题，但是在验证上出现了极大的问题，因此急需寻找一个好的有效的模型。

在找到*adaboost回归模型* 前，我还尝试了*决策树模型*，*xgboost模型*，效果均没有*adaboost回归模型* 好，因此最后采用了该模型，但是该模型还是存在不少缺陷的，希望未来学习了数学建模的相关知识后可以进行改进。

3 爬虫遇到的问题

其实一开始想简单地按照课件上的代码进行爬取，但是，后面学习了新的方法和包，感觉非常有趣，对于爬虫有了最基础的了解。同时在反爬这件事情上也学习到了一定技巧

三、其他（感悟、建议等）

1 感悟

python 综合训练算是为我打下了一个良好的科研基础吧，学习到了 python 爬虫，可以用来搜集大量数据。学习到了图表数据化，可以不用再用干巴巴的文字描述问题了。学习到了简单的 numpy 包，为科学计算打下了基础。学习到了查阅文献的能力。学习到了如何配置环境，解决实际问题。总而言之，非常高兴学习这门课程。

2 建议

希望未来这门课程的知识容量可以再大一点，比如教一下 git 等等基础科研知识，非常期待了属于是。

四、附录

爬取数据

```python
import time
import requests
from lxml import html
import csv
from concurrent.futures import ThreadPoolExecutor

f = open("西安二手房数据.csv", mode="w", encoding="utf-8")
csvwriter = csv.writer(f)
header = ["名称", "地址", "细节", "单价(元/m^2)"]
csvwriter.writerow(header)


def get_one_page(url):
    # 获取网页
    header = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) "
                      "Chrome/103.0.5060.66 Safari/537.36 Edg/103.0.1264.44 "
    }
    resp = requests.get(url=url, headers=header)

    # 解析语法树
    hl = html.etree.HTML(resp.text)
    con_1 = hl.xpath("/html/body/div[4]/div[1]/ul/li")
    for t in con_1:
        name = t.xpath("./div[1]/div[2]/div/a[1]/text()")
        local = t.xpath("./div[1]/div[2]/div/a[2]/text()")
        detail = t.xpath("./div[1]/div[3]/div/text()")
        price = t.xpath("./div[1]/div[6]/div[2]/span/text()")
        # 存入数据
        csvwriter.writerows(zip(name, local, detail, price))
    time.sleep(1)


if __name__ == '__main__':
    with ThreadPoolExecutor(5) as t:
        for i in range(1, 101):
            t.submit(get_one_page, f"https://xa.lianjia.com/ershoufang/pg{i}/")
```

词云

```python
from wordcloud import WordCloud
```

```
from openpyxl import load_workbook
import matplotlib.pyplot as plt


# 打开 xlsx
from wordcloud import WordCloud

workbook = load_workbook('西安二手房_清洗版.xlsx')
worksheet = workbook['Sheet 1']

'''''
# 把地址写入 txt 文件中
data = []
f = open("地址.txt", 'w', encoding="gbk")
for row in range(2, worksheet.max_row+1):
    data.append(worksheet['B'+str(row)].value)
f.write(str(data))
f.close()

# 读取
f = open("地址.txt", 'r', encoding="gbk")
outstr = f.readline().strip()
'''

# 生成词云图
outstr = ""
for row in range(2, worksheet.max_row+1):
    outstr += worksheet['B'+str(row)].value+" "
font = r'C:\Windows\Fonts\simsun.ttc'
wc = WordCloud(font_path=font, width=2400, height=1200, max_words=100
)
wc.generate(outstr)
wc.to_file('词云.jpg')
plt.figure(dpi=100)
plt.imshow(wc, interpolation='catrom')
plt.axis('off')
plt.show()
plt.close()
```

柱状图

```
import numpy as np
from matplotlib import pyplot as plt

data = np.loadtxt("area_pri.txt")
```

```
area, pri = [], []
for x, y in data:
    area.append(x)
    pri.append(y)
plt.bar(area, pri, align='center', alpha=0.5)
plt.xlabel("area")
plt.ylabel("price")

plt.show()
```

线性模型

```
import numpy as np
from scipy import optimize
from openpyxl import load_workbook

workbook = load_workbook('西安二手房_清洗版_版本4_数字编码.xlsx')
worksheet = workbook['编码数据_版本4_数字编码']

# 把数据写入xi，y变量中
x = [[] for i in range(6)]
for row in range(2, worksheet.max_row + 1):
    for name in ['A', 'B', 'C', 'D', 'E', 'F']:
        x[ord(name) - ord('A')].append(worksheet[name + str(row)].value)
y = []
for row in range(2, worksheet.max_row + 1):
    y.append(worksheet['G' + str(row)].value)
for i in range(len(x)):
    x[i] = np.array(x[i])
y = np.array(y)


def residuals(p):
    k0, k1, k2, k3, k4, k5, b = p
    return y - (k0 * x[0] + k1 * x[1] + k2 * x[2] + k3 * x[3] + k4 * x[4] + k4 * x[5] + b)


r = optimize.leastsq(residuals, [1, 1, 1, 1, 1, 1, 0])
print(r[0])
```

机器学习模型

```
import numpy
```

```
import pandas
from spsspro.algorithm import supervised_learning

# 生成案例数据
data_x = pandas.DataFrame({
    "A": numpy.random.random(size=100),
    "B": numpy.random.random(size=100)
})
data_y = pandas.Series(data=numpy.random.random(size=100), name="C")

# adaboost 回归，输入参数详细可以光标放置函数括号内按 shift+tab 查看，输出结果参考 spsspro 模板分析报告
result = supervised_learning.adaboost_regression(data_x=data_x, data_y=data_y)
print(result)
```