# 项目名称：python成都租房信息分析设计与实现

## 1．项目简介

### 立项的背景和意义

现在很多的年轻人、大学毕业生都喜欢搬到大城市里去生活，希望能够自己在大城市中能够有一套属于自己的房子，能够在大城市有一个一个温馨的小家，但是很多的人却被那个高昂的房价的给堵在了门外，因为这房价实在是太高了，尤其是北上广的房价实在是让人只能仰望的存在，根本就买不起，所以说随着时间的推移，现在很多的年轻人都选择租房而不是买房，因为租房同样是住，买房也同样住。

随着租房市场的高涨，近年来北京、上海、广州、深圳、天津、武汉、重庆、南京、杭州和成都十大城市租金环比均有所上涨。然而现在大城市的租金究竟有多高？我以目前市场占有率最高的房屋中介公司为目标，来获取所在地成都这座大城市的租房信息并简单分析。

同时随着租房的火热，各类房租中介与APP鱼龙混杂，让大量租房者获取不到关键有效信息，于是我准备简单利用高德地图的开发API加载爬取下来的数据，让房源展示在地图上。这样不仅能最直观的感受房源分布，同时能最直观的感受房源与工作地点的交通通行方式与通行时间。

### 项目功能需求

1. .使用BeautifulSoup4爬取链家成都租房信息，并存储成csv格式文件。
2. .使用pandas处理数据，使用matplotlib做数据可视化，展示成都各个区平均房租，成都各个区平均每平米房租单价，成都各个区平均房租面积。
3. 使用线性回归分析面积、所属区与价钱的关系。
4. 利用高德地图开发API加载成都市房源，展示在html页面上。

## 2 实验简介

### 2.1实验内容与知识点

#### 2.1.1爬虫

## 需要爬取的网页分析:

爬取的是链家的数据，网页地址:[https://cd.lianjia.com/zufang/jinjiang/](https://cd.lianjia.com/zufang/jinjiang/" \t "https://blog.csdn.net/qq_40351478/article/details/_blank)  
先要分析链家如何实现成都市各个区以及页数的拼接：

IMG_256

比如青羊区的第二页，地址是：https://cd.lianjia.com/zufang/qingyang/pg2

也就是说url=https://cd.lianjia.com/zufang+各个区的拼音+页数。

分析网页发现我们要爬取的数据都在class="content\_\_list–item"的div中，我们可以先获取所有class="content\_\_list–item"的div在，依次遍历获取每个div中的数据。

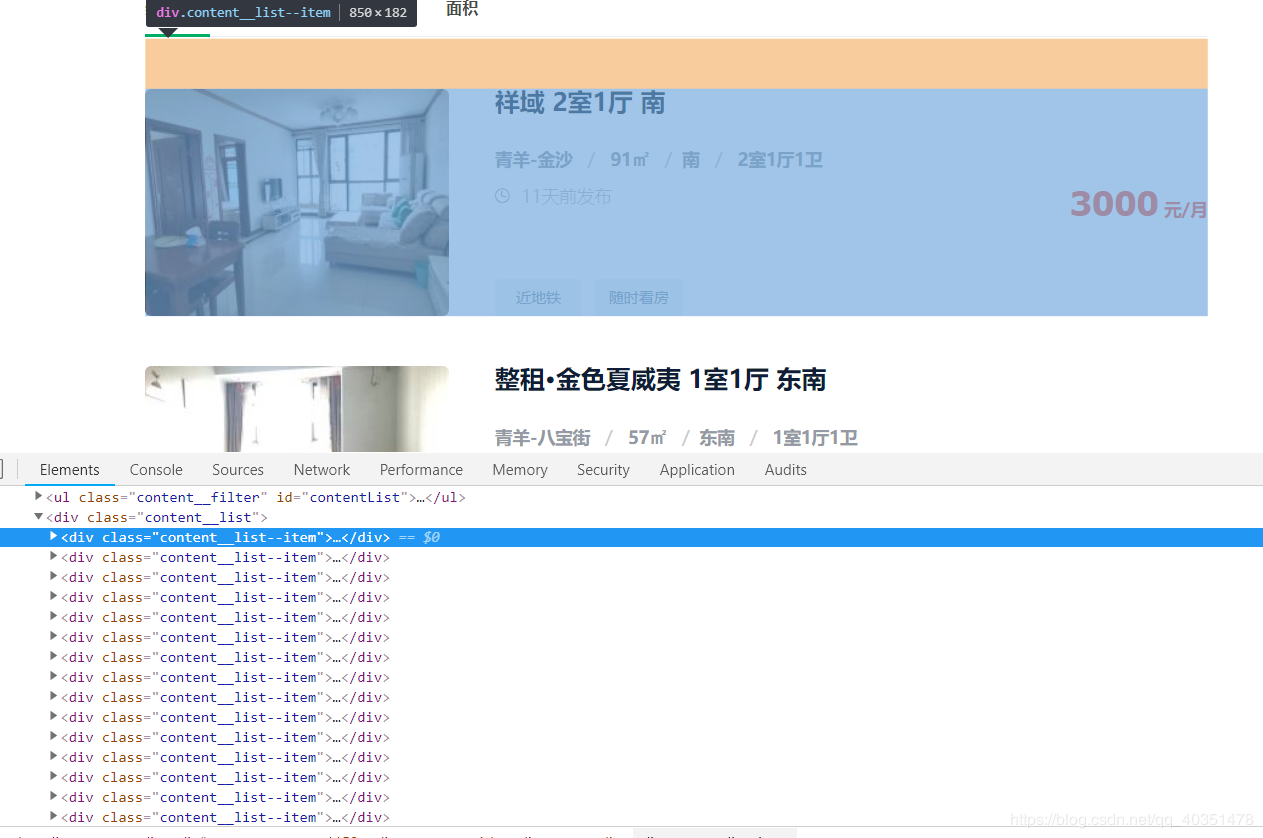


图1

爬虫使用的是bs4,简单来说bs4的使用：

第一步：导入from bs4 import BeautifulSoup 这个模块。

from bs4 import BeautifulSoup

第二步：创建 Beautiful Soup 对象 soup = BeautifulSoup(html) ，至于如何获取headers，按F12，User-Agent便是我们需要的。

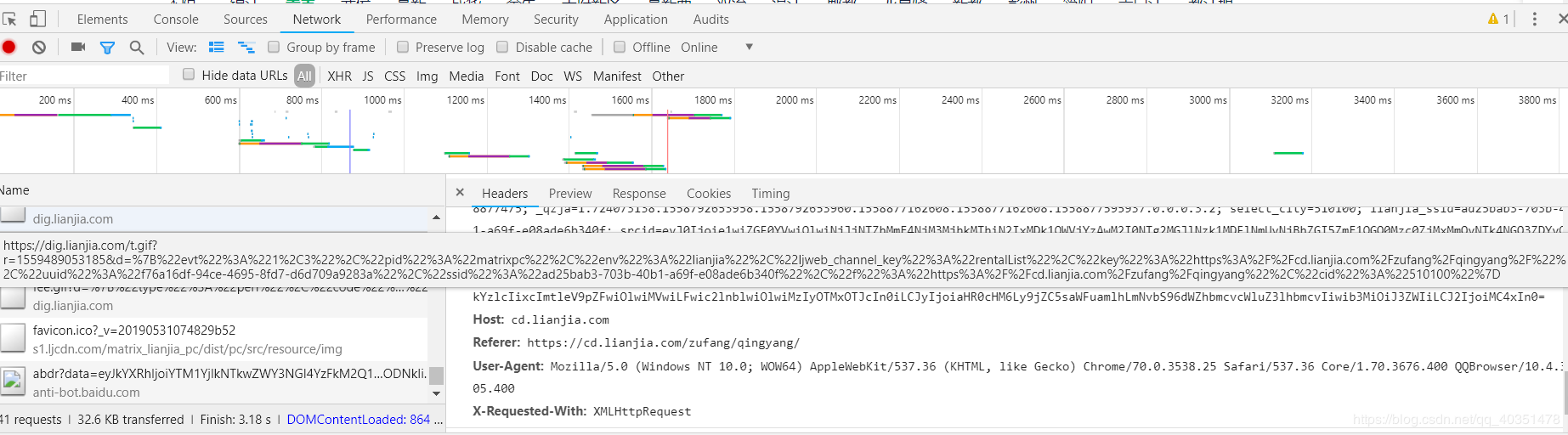


图2

### response=requests.get(url,headers=headers)#,headers=headers

### response.encoding=response.apparent\_encoding

### soup=BeautifulSoup(response.text,'html.parser')# BeautifulSoup解析

### 第三步使用各种选择器解析提取数据。比如提取div中数据如下。

totaldivlist=soup.find\_all("div", attrs={"class","content\_\_list--item"})

第四步使用csv存储数据。

csv\_file = open("cdlianjia.csv", "w", newline='')

csv\_writer = csv.writer(csv\_file, delimiter=',')

csv\_writer.writerow([house\_name, house\_layout,house\_direction, house\_area,area, address,price])

#### 2.1.2数据可视化:

第1步:使用pandas分别获取成都市各个区房租数据。

*# 获取所属区*areas = list(Date.groupby(**'所属区'**).size().index)  
*#每个区房租平均价钱*area\_mean\_price=[]  
*#每个区房屋平均面积*area\_mean\_house\_area=[]  
*#每个区平均每平米房租价钱*area\_mean\_perhouseareaprice=[]

第2步:计算成都各个区房租相关数据。

for i in range(len(price\_num\_total)):  
 price\_num\_all=price\_num\_all+price\_num\_total[i]  
*#当前区房租平均价钱*price\_mean=price\_num\_all/len(price\_num\_total)  
*#存入房租平均价钱*area\_mean\_price.append(price\_mean)

第3步:使用matplotlib展示相关数据。

#### 2.1.3线性回归:

数据集一共有3w数据，每一组观测值对应一个房租情况。

步骤1:首先使用pandas库导入数据:

import pandas as pd

Date = pd.read\_csv('csv/completed.csv')

步骤2:由于读入的数据结构是Pandas的数据帧(data frame),而Scikit-learn要求的自变量X是一个特征矩阵，因变量y是一个NumPy向量。因此需要从原先的数据帧中转还数据，代码如下：

X=Date.loc[:,['面积','所属区']]

y=Date.loc[:,['价钱']]

步骤3:在这个数据集中，由于有3w个观测值，需要构建训练集与测试集，同样使用Scikit-learn中的model\_selection库对数据进行划分，代码如下:

from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,y\_train,y\_test=train\_test\_split(X,y,test\_size=.9,random\_state=0)

上面代码将原数据集的90%划分为训练集，原数据集的10%为测试集。  
步骤4:使用多元回归模型:

from sklearn.linear\_model import LinearRegression

linreg=LinearRegression()

model=linreg.fit(X\_train,y\_train)

步骤5:模型训练结束后，使用该模型进行预测:

y\_pred=linreg.predict(X\_test)

步骤6:数据可视化比较测试数据与预测数据的关系,这里为了更方便观察，只使用了少量数据进行画图:

plt.figure()

plt.plot(range(len(y\_pred[1:130])),y\_pred[1:130],'b',label="price\_predict")

plt.plot(range(len(y\_pred[1:130])),y\_test[1:130],'r',label="price\_test")

plt.legend(loc="upper right")

plt.show()

最终结果如下:

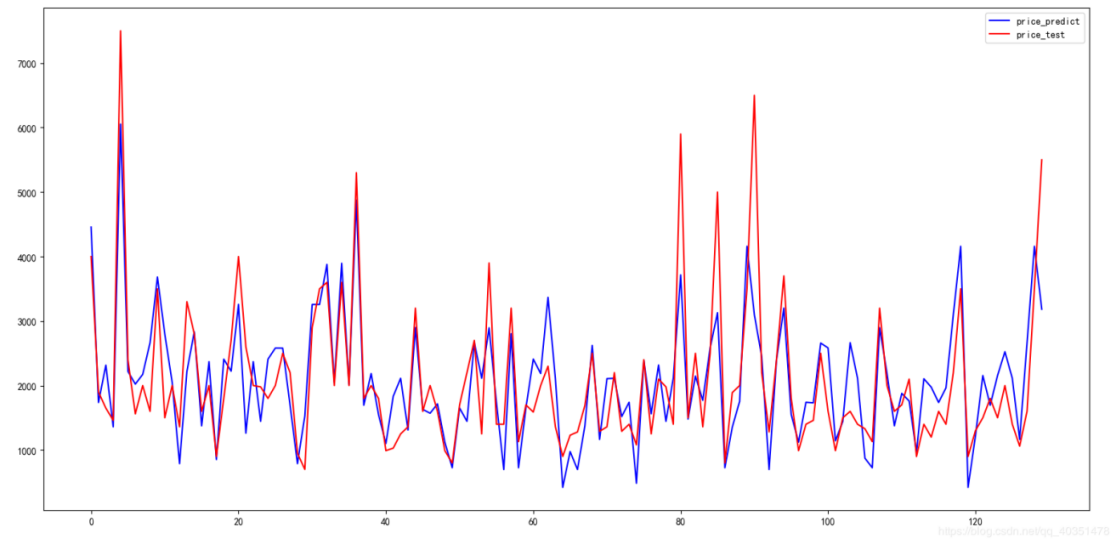


图3

#### 2.1.4随机森林:

步骤1、2、3参考上面线性回归。

步骤4使用随机森林模型:

try:  
 clf = joblib.load('model/random.h5') *# 加载API*except:  
 clf = RandomForestClassifier()  
 clf.fit(X\_train, y\_train)  
 joblib.dump(clf, "model/random.h5") *# 保存*y\_pred2 = clf.predict(X\_test)

步骤5、6参考线性回归。

最终结果如下:

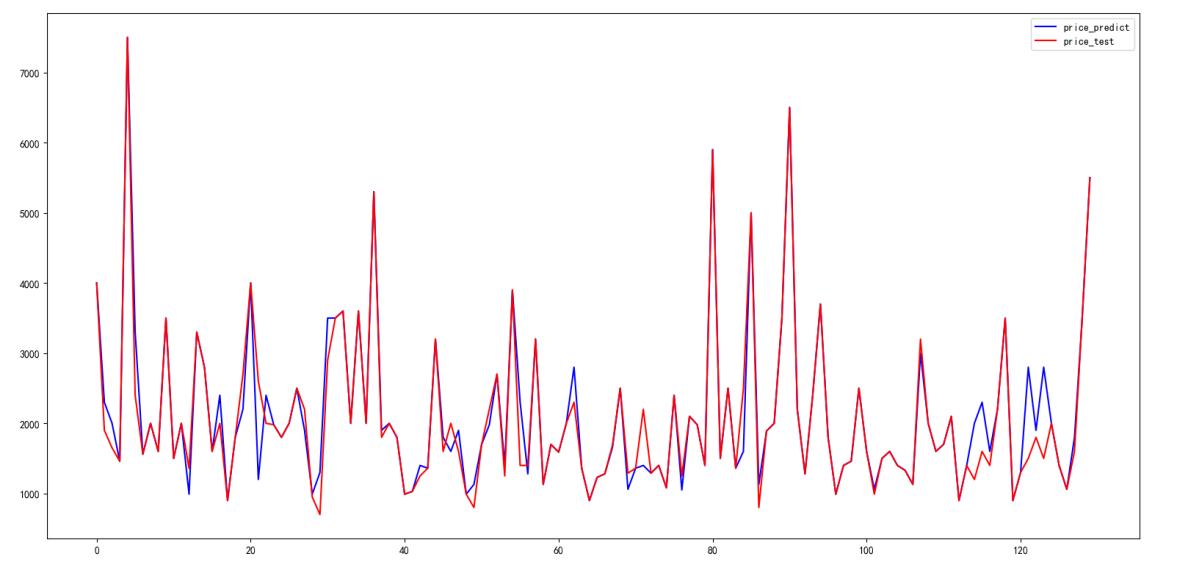


图4

#### 2.1.5朴素贝叶斯:

步骤1、2、3参考上面线性回归。

步骤4使用朴素贝叶斯模型:

try:  
 mnb = joblib.load('model/mnb.pkl')  
except:  
 mnb = MultinomialNB() *# 使用默认配置初始化朴素贝叶斯* mnb.fit(X\_train,y\_train) *# 利用训练数据对模型参数进行估计*y\_pred3 = mnb.predict(X\_test) *# 对参数进行预测*

步骤5、6参考线性回归。

最终结果如下:

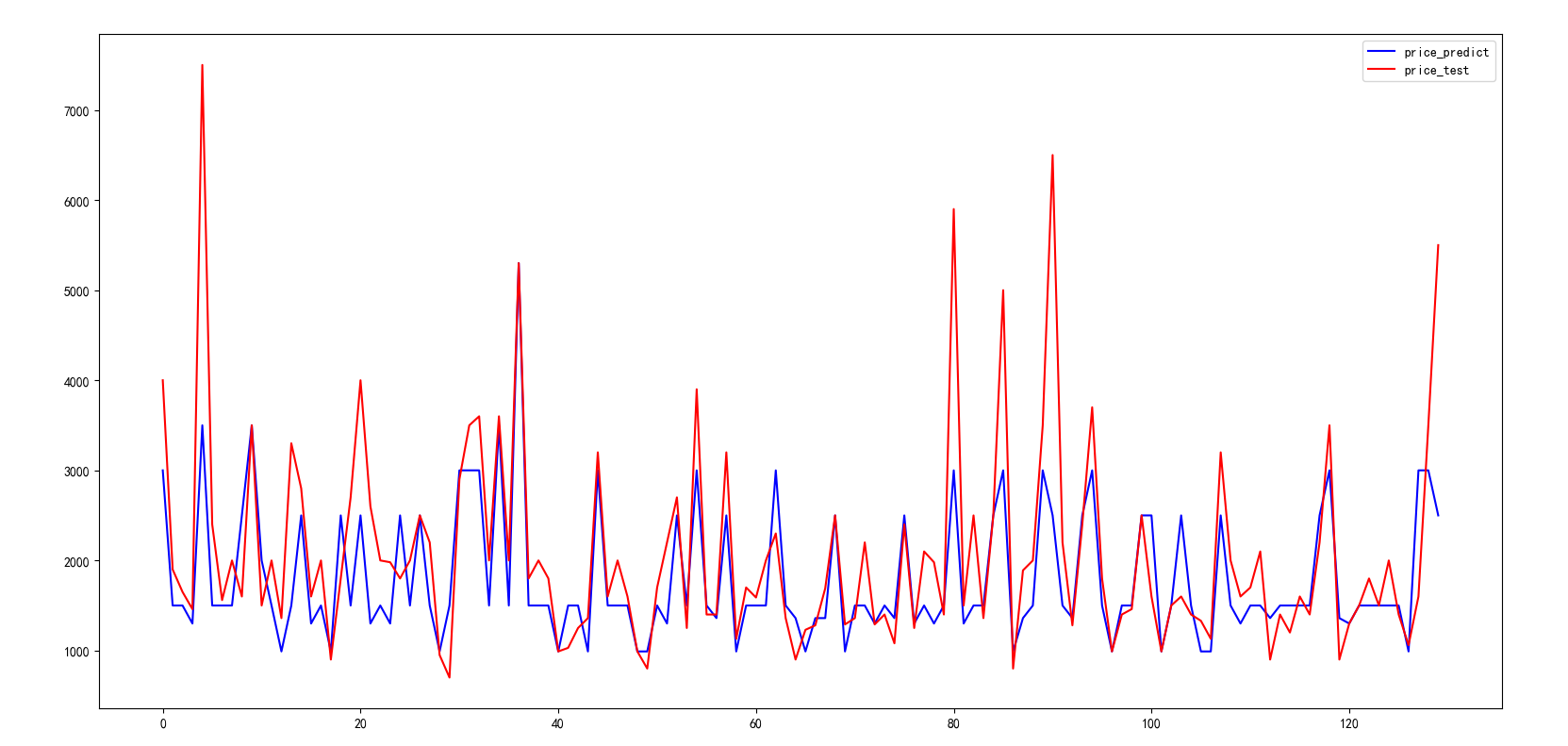


图5

#### 2.1.6高德地图加载房源:

步骤1: key 这个参数，你需要注册注册高德的开发者用户，创建应用才能得到key值，然后在你的项目中导入：

<script src="http://webapi.amap.com/maps?v=1.3&**key=22d3816e107f199992666d6412fa0691&plugin**=AMap.ArrivalRange,AMap.Scale,AMap.Geocoder,AMap.Transfer,AMap.Autocomplete"></script>

步骤2：载入编写代码时可能用到的 API 插件

<script src="http://webapi.amap.com/maps?v=1.3&key=22d3816e107f199992666d6412fa0691&plugin=**AMap.ArrivalRange,AMap.Scale,AMap.Geocoder,AMap.Transfer,AMap.Autocomplete**"></script>

这些插件包括:

ArrivalRange：公交到达圈

Scale：标尺

Geocoder：正向地理编码（地址-坐标）

Transfer：路径规划

Autocomplete：地址自动补全

步骤3:JQUERY读取csv文件

*//载入房源文件*function *importRentInfo*(fileInfo) {  
 var file = fileInfo.files[0].name;  
 *loadRentLocationByFile*(file);  
}

步骤4:利用高德地图进行各种开发操作，篇幅有限这里只列举了部分全部代码

*//公交到达圈对象*var *arrivalRange* = new AMap.ArrivalRange();  
*//经度，纬度，时间（用不到），通勤方式（默认是地铁＋公交）*var *x*, *y*, *t*, *vehicle* = "SUBWAY,BUS";  
*//工作地点，工作标记*var *workAddress*, *workMarker*;  
*//房源标记队列*var *rentMarkerArray* = [];  
*//多边形队列，存储公交到达的计算结果*var *polygonArray* = [];  
*//路径规划*var *amapTransfer*;  
*//信息窗体对象,点击房源点后出现*var *infoWindow* = new AMap.InfoWindow({  
 offset: new AMap.Pixel(0, -30)  
});

### 2.2 实验环境及技术路线

实验环境:

开发语言:Python3.6.5

开发软件:JetBrains PyCharm 2018.3.5 x64

技术路线:

Python:熟练使用python基础知识。

爬虫:requests、BeautifulSoup、csv 等库的简单使用。  
数据处理及可视化:matplotlib，pandas，numpy等库的使用。

机器学习型:线性回归，随机森林，朴素贝叶斯。  
地图显示:html,css,,jquery,高德地图API。

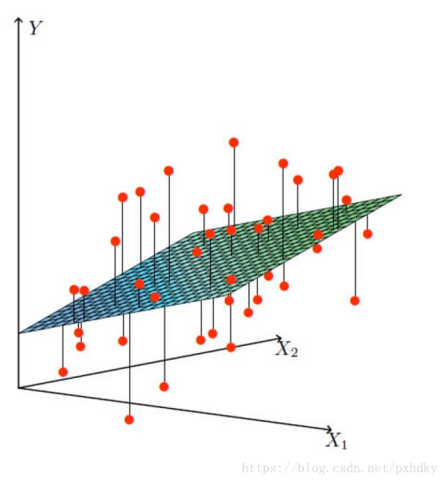
### 3 实验原理

### 3.1 算法原理

#### 3.1.1线性回归

# ****1. 概念****

线性回归（Linear Regression）是一种通过属性的线性组合来进行预测的线性模型，其目的是找到一条直线或者一个平面或者更高维的超平面，使得预测值与真实值之间的误差最小化。



# ****2. 特点****

1. 优点：结果具有很好的可解释性（w直观表达了各属性在预测中的重要性），计算熵不复杂。
2. 缺点：对非线性数据拟合不好
3. 适用数据类型：数值型和标称型数据

# ****3. 原理与推导****

****1.****给定数据集IMG_256，其中IMG_256，IMG_256（线性回归的输出空间是整个实数空间）。IMG_257是样本数，IMG_258是属性维度。

线性回归试图学得：IMG_256（1），使得IMG_256。

为便于讨论，使IMG_256，其中IMG_257。此时，IMG_258就成为了IMG_259，IMG_260就成为了IMG_261，期望学得的函数为IMG_262。

****2.****预测值和真实值之间都肯定存在差异IMG_263，对于每个样本：

IMG_256 （2）

假设误差IMG_256是独立同分布的，并且服从高斯分布。即：

IMG_256  （3）

将（2）代入（3）中，得到在已知参数IMG_256和数据IMG_257的情况下，预测值为IMG_258的条件概率：

IMG_256  （4）

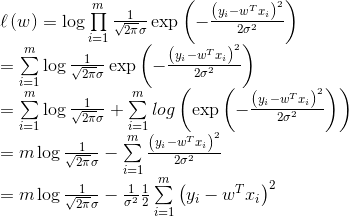
****3.****将（4）连乘得到在已知参数IMG_256和数据IMG_257的情况下，预测值为IMG_258的条件概率，这个条件概率在数值上等于，likelihood（w|x,y），也就是在已知现有数据的条件下，w是真正参数的概率，即似然函数（5）：

IMG_256（5）

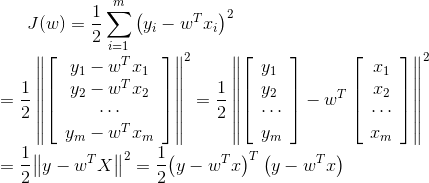
**为什么要引入似然函数：为了根据样本估计参数值。**

**为什么要对似然函数进行log变换：由于乘法难解，通过对数可以将乘法转换为加法，简化计算。**

对数似然函数：

（6）

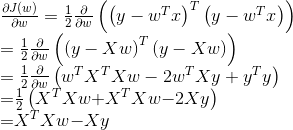
得到目标函数：

（7）（最小二乘法）

**为什么要让目标函数越小越好：似然函数表示样本成为真实的概率，似然函数越大越好，也就是目标函数**IMG_256**越小越好。**

****4.****目标函数是凸函数，只要找到一阶导数为0的位置，就找到了最优解。

因此求偏导：

 （8）

****5.****令偏导等于0：

IMG_256  （9）

得到：IMG_256 （10）

情况一：IMG_256可逆，唯一解。令公式（10）为零可得最优解为：

IMG_256 （11）

 学得的线性回归模型为:

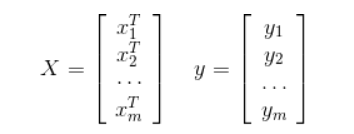
IMG_256 （12）

情况二：IMG_256不可逆，可能有多个解。选择哪一个解作为输出，将有学习算法的偏好决定，常见的做法是增加IMG_257扰动。

IMG_256 （13）

## ****4. 算法描述****

****1.****从数据集D出发，构建输入矩阵X和输出向量y。



****2.****计算伪逆（pseudo-inverse）IMG_256。

****3.**** 返回IMG_257，学得的线性回归模型为IMG_258。

#### 3.1.2随机森林

**1.随机森林原理：**

随机森林由Leo Breiman（2001）提出的一种分类算法，它通过自助法（bootstrap）重采样技术，从原始训练样本集N中有放回地重复随机抽取n个样本生成新的训练样本集合训练决策树，然后按以上步骤生成m棵决策树组成随机森林，新数据的分类结果按分类树投票多少形成的分数而定。其实质是对决策树算法的一种改进，将多个决策树合并在一起，每棵树的建立依赖于独立抽取的样本。

单棵树的分类能力可能很小，但在随机产生大量的决策树后，一个测试样本可以通过每一棵树的分类结果经统计后选择最可能的分类。

随机森林大致过程如下：

1）从样本集中有放回随机采样选出n个样本；

2）从所有特征中随机选择k个特征，对选出的样本利用这些特征建立决策 树（一般是CART，也可是别的或混合）；

3）重复以上两步m次，即生成m棵决策树，形成随机森林；

4）对于新数据，经过每棵树决策，最后投票确认分到哪一类。

**2.随机森林特点：**

随机森林有很多优点：

1） 每棵树都选择部分样本及部分特征，一定程度避免过拟合；

2） 每棵树随机选择样本并随机选择特征，使得具有很好的抗噪能力，性能稳定；

3） 能处理很高维度的数据，并且不用做特征选择；

4） 适合并行计算；

5） 实现比较简单；

缺点：

1） 参数较复杂；

2） 模型训练和预测都比较慢。

**3.使用：**

随机森林算法在大部分数据处理软件中都有实现，使用时可以直接调用，只需指定所需参数。

随机森林模型训练前要设置的参数较多，按PAI平台的实现有如下几个：

o 算法类型：（可选）可供选择的算法类型有id3算法、cart算法、c4.5算法以及默认情况下的将上述三种算法均分的混合算法

o 树的数目：森林中树的个数, 范围(0, 1000]

o 随机属性个数：（可选）单颗树在生成时，每次选择最优特征，随机的特征个数。可供选择的类型有logN，N/3，sqrtN，N四种类型，其中N为属性总数

o 树最大深度：（可选）单颗树的最大深度，范围[1, ∞)，-1表示完全生长。

o 叶子节点最少记录数：（可选）叶节点数据的最小个数。最小个数为2

o 叶子节点最少记录百分比：（可选）叶节点数据个数占父节点的最小比例，范围[0,100]，-1表示无限制。默认-1

o 每棵树最大记录数：（可选）森林中单颗树输入的随机数据的个数。范围为(1000, 1000000]

4.模型评估：

算法模型建立后需要进行评估，以判断模型的优劣。一般使用训练集 (training set) 建立模型，使用测试集 (test set) 来评估模型。对于分类算法评估指标有分类准确度、召回率、虚警率和精确度等。而这些指标都是基于混淆矩阵 (confusion matrix) 进行计算的。

混淆矩阵用来评价监督式学习模型的精确性，矩阵的每一列代表一个类的实例预测，而每一行表示一个实际的类的实例。以二分类问题为例，如下表所示：



图6

其中

P (Positive Sample)：正例的样本数量。

N (Negative Sample)：负例的样本数量。

TP (True Positive)：正确预测到的正例的数量。

FP (False Positive)：把负例预测成正例的数量。

FN (False Negative)：把正例预测成负例的数量。

TN (True Negative)：正确预测到的负例的数量。

根据混淆矩阵可以得到评价分类模型的指标有以下几种。

分类准确度，就是正负样本分别被正确分类的概率，计算公式为：IMG_256

召回率，就是正样本被识别出的概率，计算公式为：IMG_256

虚警率，就是负样本被错误分为正样本的概率，计算公式为：IMG_256

精确度，就是分类结果为正样本的情况真实性程度，计算公式为：IMG_256

评估方法有保留法、随机二次抽样、交叉验证和自助法等。

保留法 (holdout) 是评估分类模型性能的最基本的一种方法。将被标记的原始数据集分成训练集和检验集两份，训练集用于训练分类模型，检验集用于评估分类模型性能。但此方法不适用样本较小的情况，模型可能高度依赖训练集和检验集的构成。

随机二次抽样 (random subsampling) 是指多次重复使用保留方法来改进分类器评估方法。同样此方法也不适用训练集数量不足的情况，而且也可能造成有些数据未被用于训练集。

交叉验证 (cross-validation) 是指把数据分成数量相同的 k 份，每次使用数据进行分类时，选择其中一份作为检验集，剩下的 k-1 份为训练集，重复 k 次，正好使得每一份数据都被用于一次检验集 k-1 次训练集。该方法的优点是尽可能多的数据作为训练集数据，每一次训练集数据和检验集数据都是相互独立的，并且完全覆盖了整个数据集。也存在一个缺点，就是分类模型运行了 K 次，计算开销较大。

自助法 (bootstrap) 是指在其方法中，训练集数据采用的是有放回的抽样，即已经选取为训练集的数据又被放回原来的数据集中，使得该数据有机会能被再一次抽取。用于样本数不多的情况下，效果很好。

#### 3.1.3朴素贝叶斯

**1.贝叶斯定理**

首先，要明白贝叶斯统计方式与统计学中的频率概念是不同，从频率的角度出发，即假定数据遵循某种分布，我们的目标是确定该分布的几个参数，在某个固定的环境一下做模型。而贝叶斯则是根据实际的推理方式来建模。我们拿到的数据，来更新模型对某事件即将发生的可能性的预测结果。在贝叶斯统计学中，我们使用数据来描述模型，而不是使用模型来描述数据。

贝叶斯定理旨在计算P(A|B)的值，也就是在已知B发生的条件下，A发生的概率是多少。大多数情况下，B是被观察事件，比如“昨天下雨了”，A为预测结果“今天会下雨”。对数据挖掘来说，B通常是观察样本个体，A为被预测个体所属类别。所以，说简单一点，贝叶斯就是计算的是：B是A类别的概率。

贝叶斯公式：

IMG_256

举例说明，我们想计算含有单词drugs的邮件为垃圾邮件的概率。

在这里，A为“这是封垃圾邮件”。我们先来计算P(A)，它也被称为先验概率，计算方法是，统计训练中的垃圾邮件的比例，如果我们的数据集每100封邮件有30封垃圾邮件，P(A)为30/100=0.3。

B表示“该封邮件含有单词drugs”。类似地，我们可以通过计算数据集中含有单词drugs的邮件数P(B)。如果每100封邮件有10封包含有drugs，那么P(B)就为10/100=0.1。

P(B|A)指的是垃圾邮件中含有的单词drugs的概率，计算起来也很容易，如果30封邮件中有6封含有drugs，那么P(B|A)的概率为6/30=0.2。

现在，就可以根据贝叶斯定理计算出P(A|B)，得到含有drugs的邮件为垃圾邮件的概率。把上面的每一项带入前面的贝叶斯公式，得到结果为0.6。这表明如果邮件中含有drugs这个词，那么该邮件为垃圾邮件的概率为60%。

**2.朴素贝叶斯**

其实，通过上面的例子我们可以知道它能计算个体从属于给定类别的概率。因此，他能用来分类。

我们用C表示某种类别，用D代表数据集中的一篇文档，来计算贝叶斯公式所要用到的各种统计量，对于不好计算的，做出朴素假设，简化计算。

P(C)为某一类别的概率，可以从训练集中计算得到。

P(D)为某一文档的概率，它牵扯到很多特征，计算很难，但是，可以这样理解，当在计算文档属于哪一类别时，对于所有类别来说，每一篇文档都是独立重复事件，P(D)相同，因此根本不用计算它。稍后看怎样处理它。

P(D|C)为文档D属于C类的概率，由于D包含很多特征，计算起来很难，这时朴素贝叶斯就派上用场了，我们朴素地假定各个特征是互相独立的，分别计算每个特征（D1、D2、D3等）在给定类别的概率，再求他们的积。

上式右侧对于二值特征相对比较容易计算。直接在数据集中进行统计，就能得到所有特征的概率值。

相反，如果我们不做朴素的假设，就要计算每个类别不同特征之间的相关性。这些计算很难完成，如果没有大量的数据或足够的语言分析模型是不可能完成的。

到这里，算法就很明确了。对于每个类别，我们都要计算P(C|D)，忽略P(D)项。概率较高的那个类别即为分类结果。

### 3.2 数据获取或导入

[数据获取:参考2.1.1爬虫](#_2.1.1爬虫)

数据导入:打开csv文件

Date = pd.read\_csv(**'csv/cdlianjia.csv'**)

## 4项目文件结构

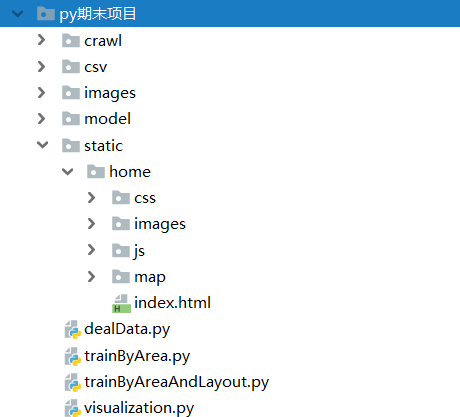


图7

## 5 实验步骤

第一步:爬虫爬取租房数据。

第二步：租房数据分析与可视化。

第三步:预测租房信息。

第四步:使用高德地图加载房源信息。

所有的详细步骤参考[2.1实验内容与知识点](#_2.1实验内容与知识点)。

## 6 测试结果及分析

成都各个区每月平均房租如图8,成都房租以高新区最高一个月3748元，新都区最低1443元。

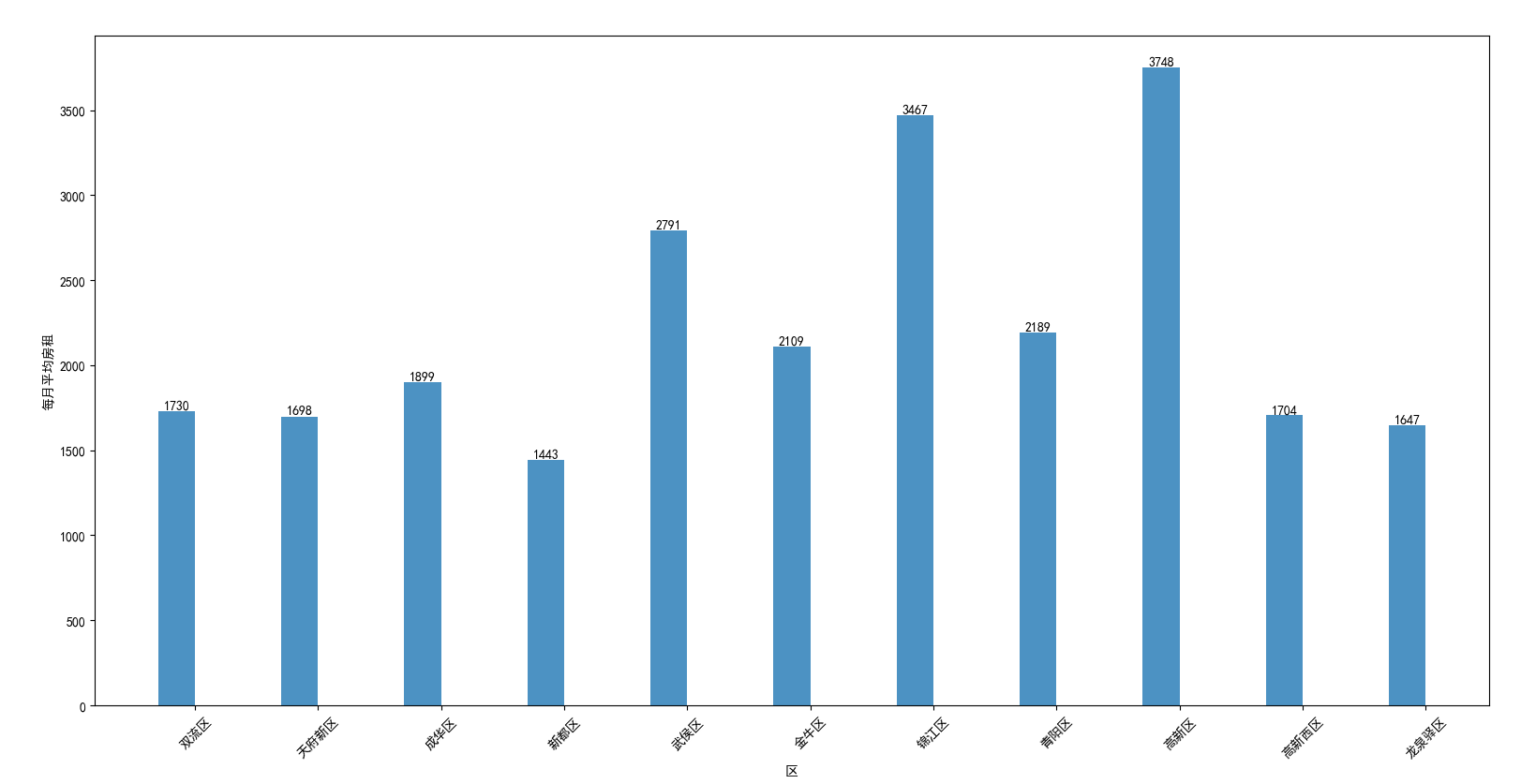


图8

成都各个区平均租房面积如下图9，成都租房面积最高的是双流区达到93平方米，最低的是成华区只有45平方米。

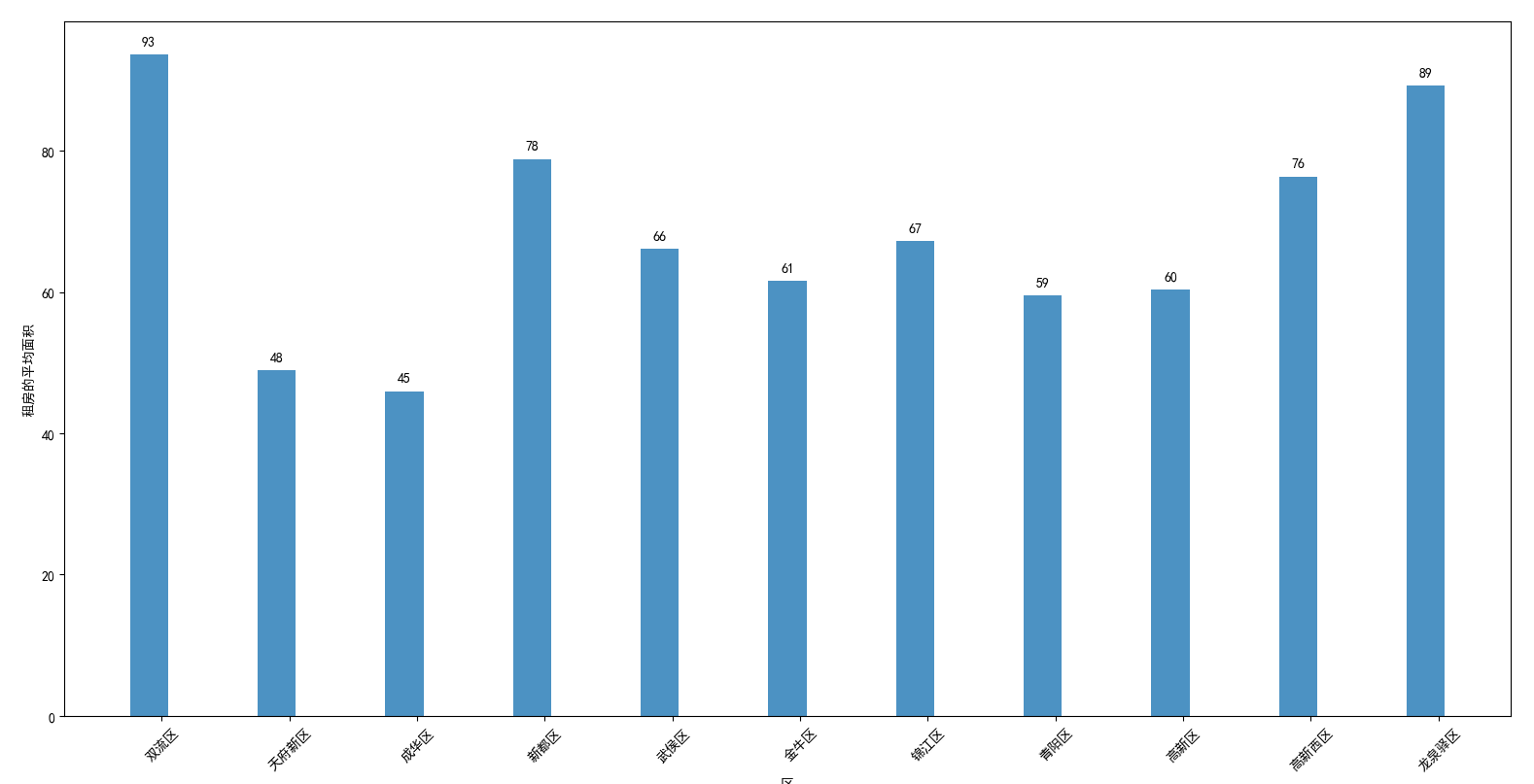


图9

成都各个区每月每平米平均房租如下图10，高新区一骑绝尘，而最低的龙泉驿区只有高新区的四分之一不到。在成都高新区组一个100平米的房子要8000，而在龙泉驿区只需要1900:

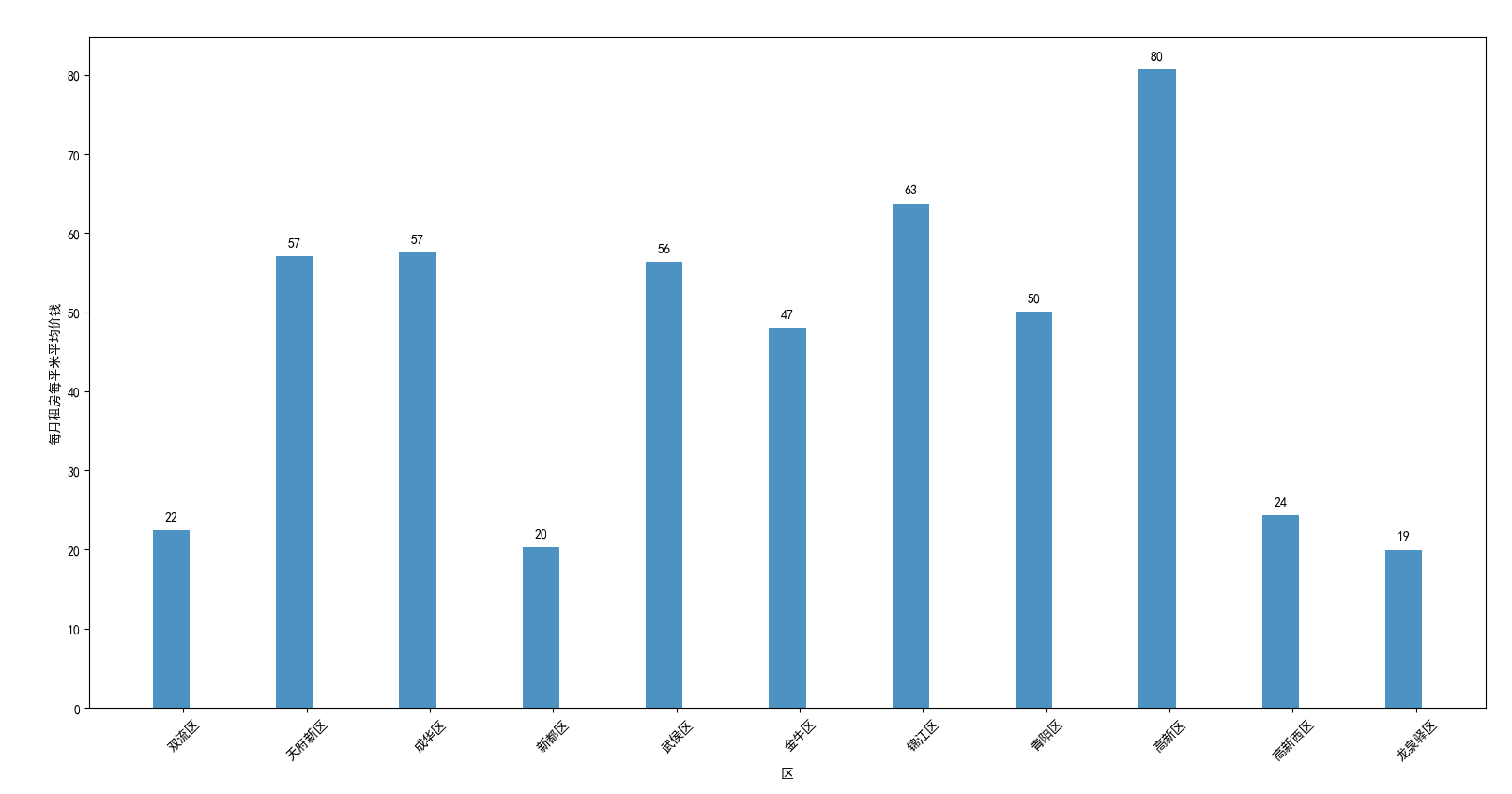


图10

究竟是什么原因造成房价这样悬殊呢？我使用了线性回归，随机森林，朴素贝叶斯三种算法对房租价钱、面积、房子所属区进行分析预测。图11，图12，图13依次是线性回归，随机森林，朴素贝叶斯的效果。

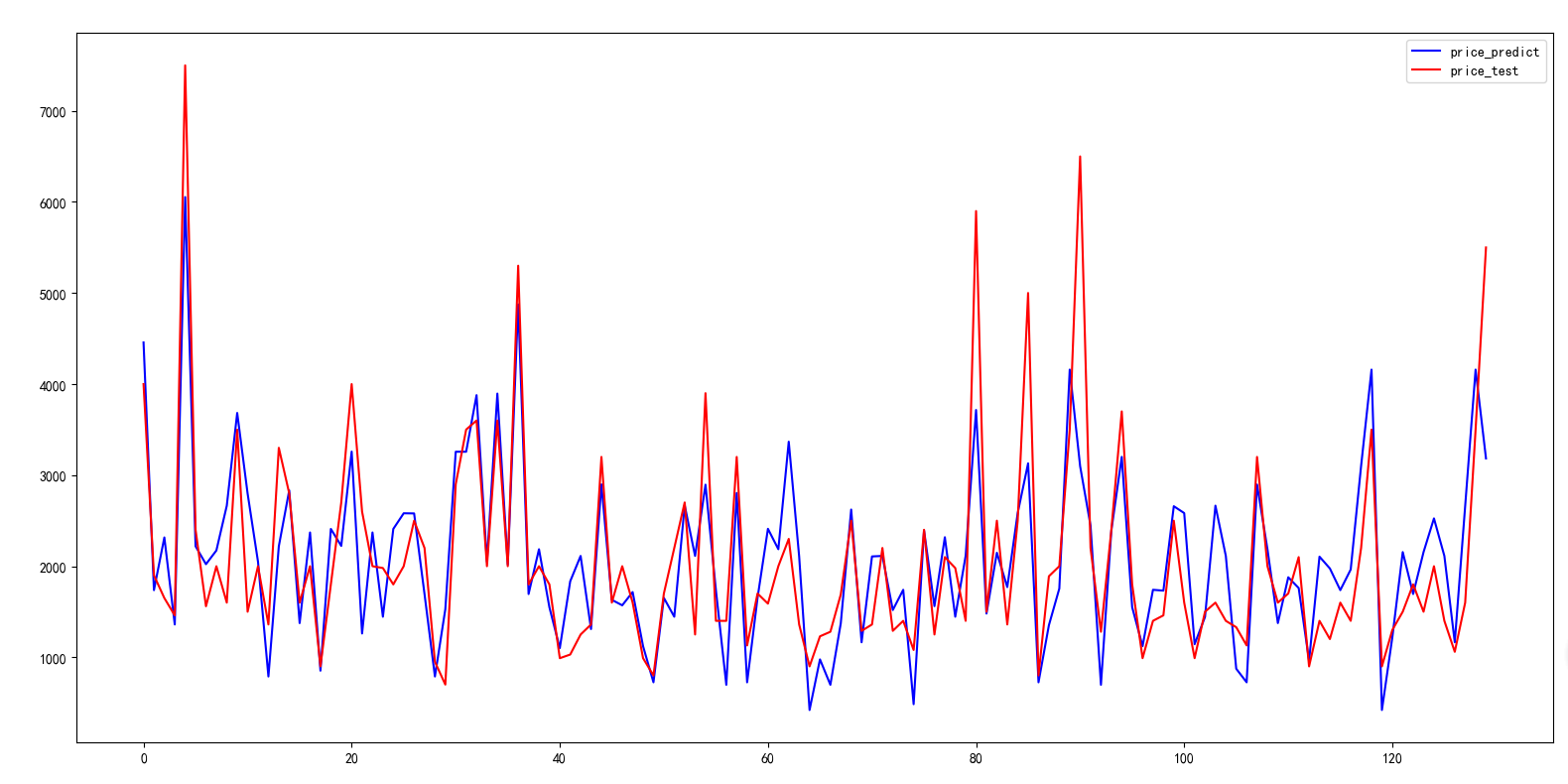


图11

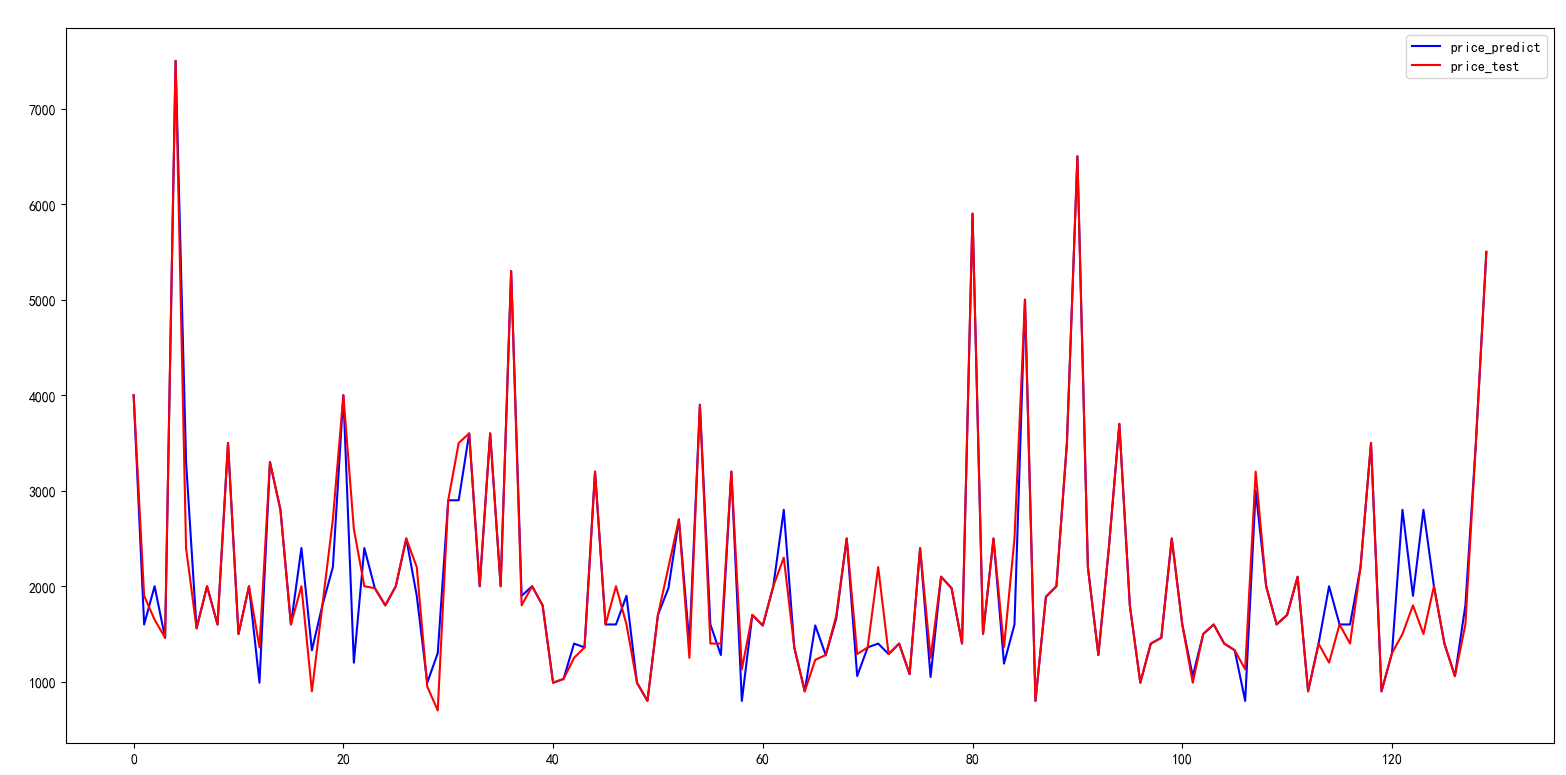


图12

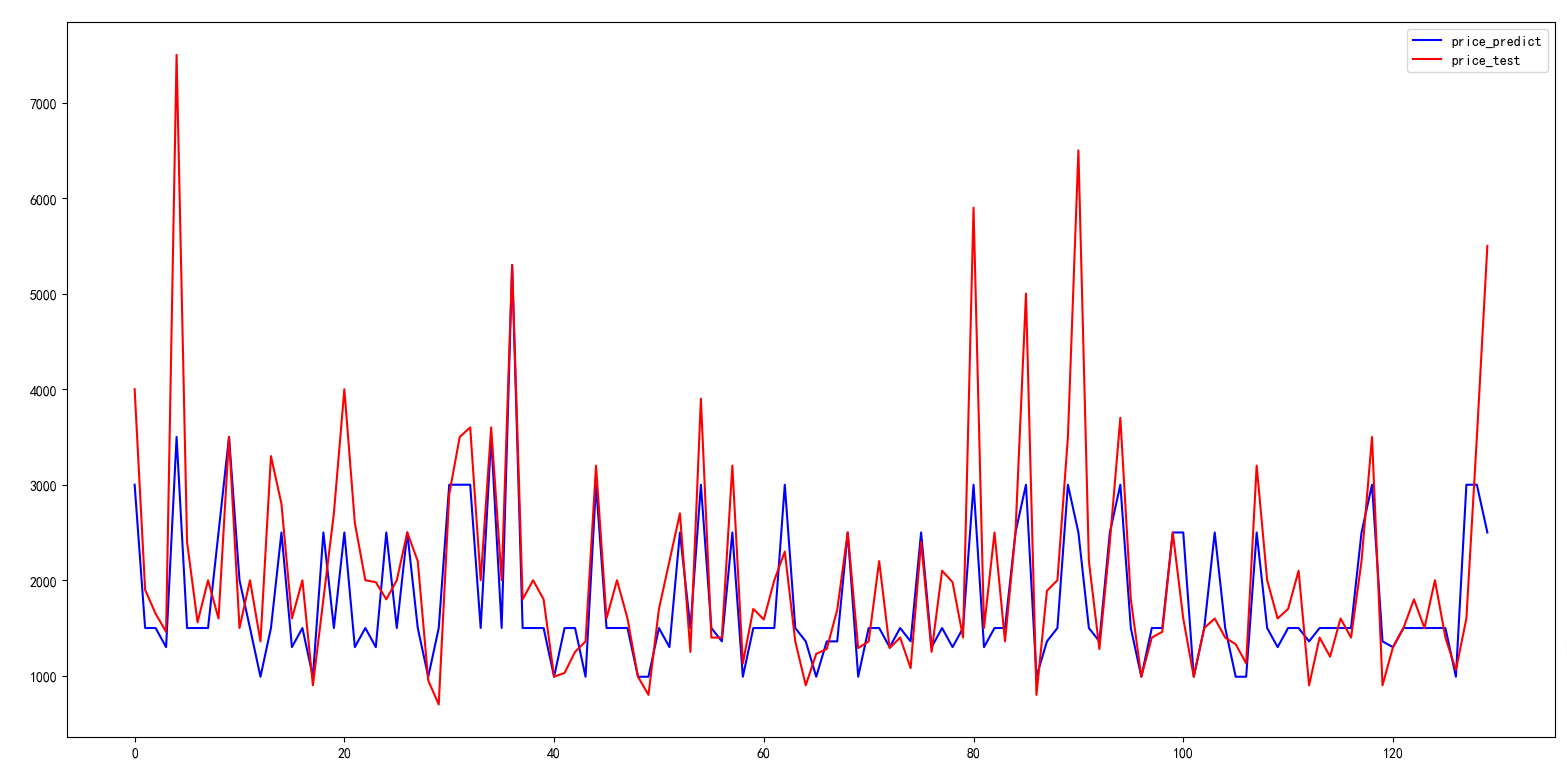


图13

可以看到效果最好的是随机森林，最差的是朴素贝叶斯。

高德地图加载房源效果如图14，利用高德地图的开发API加载爬虫爬取下来的数据，让房源展示在地图上，让租房者更易于查看。同时画出了到房源地点的交通路径。



图14

## 7项目总结

本次期末项目收获颇丰，不仅强化了我的python基础知识，同时是我更加熟悉爬虫、requests、BeautifulSoup、csv、matplotlib，pandas，numpy等库的使用。对之前不是很熟悉的机器学习算法比如线性回归，随机森林，朴素贝叶斯等也有了进一步的学习。同时更是学会了使用高德地图的开发API。