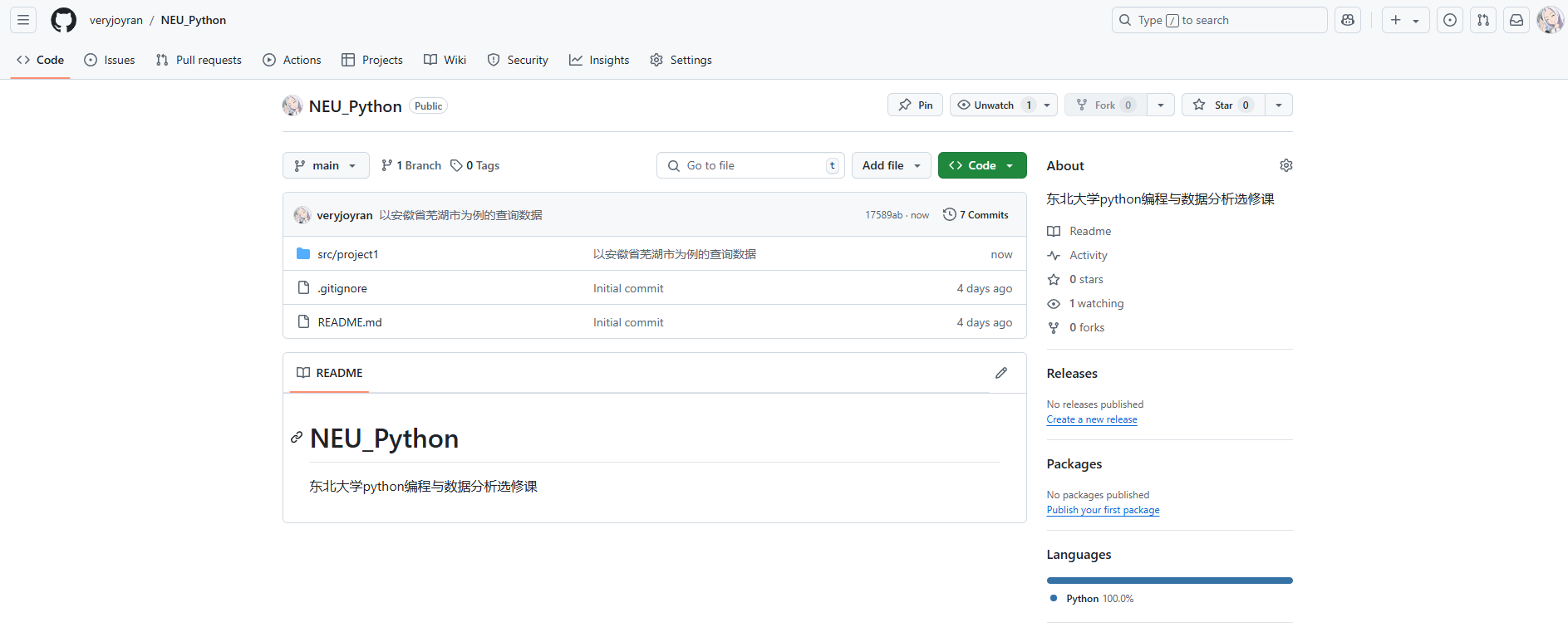
Project1：二手房查询与房价预测系统-控制台版本

人工智能2202班-刘庆然-20226494

所有代码以及README均已上传至github:

https://github.com/veryjoyran/NEU\_Python/tree/main/src/project1





目录

[一、 问题描述 3](#_Toc182612789)

[二、 说明文档 4](#_Toc182612790)

[三、实验结果 5](#_Toc182612791)

[1.数据读取 5](#_Toc182612792)

[2.数据查询（安徽省芜湖市为例） 6](#_Toc182612793)

[2.1 均值填充 6](#_Toc182612794)

[2.2 线性回归预测 7](#_Toc182612795)

[3.数据保存到文件 8](#_Toc182612796)

[4.沈阳房价查询结果示例(按照收藏数量排序) 9](#_Toc182612797)

[四、源码解读 10](#_Toc182612798)

[main.py: 10](#_Toc182612799)

[源代码： 12](#_Toc182612800)

# 问题描述

二手房查询与房价预测系统-控制台版本

功能要求： 1. 建立系统主体框架，能够运行起来；

2. 输入房屋查询条件（如城市名称、位置、户型等），列出二手房数据（可以用测试数据）；

3. 系统能够将运行信息写到日志文件。

实现系统的主体框架正序能够正常运行即可。

# 说明文档



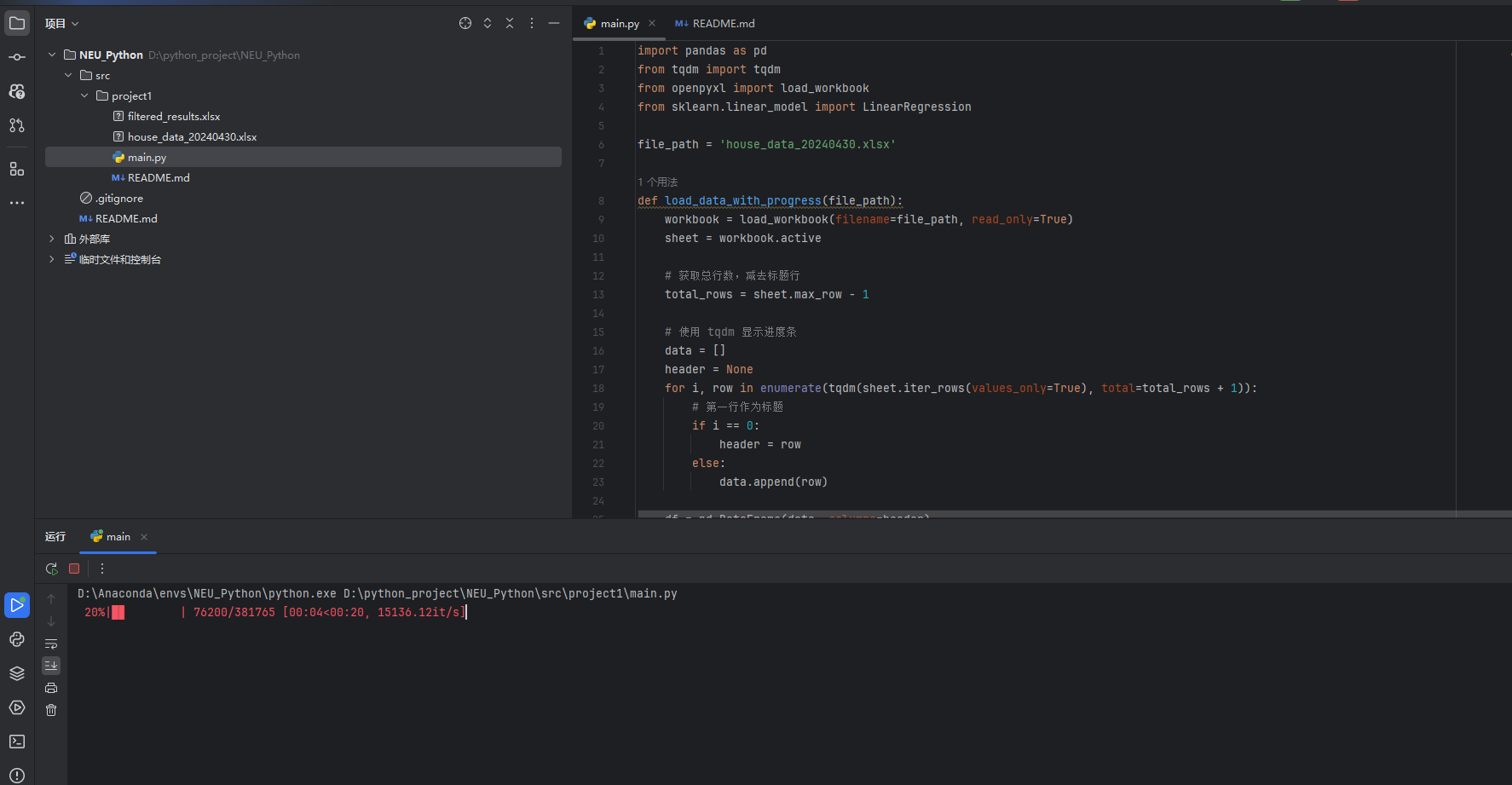


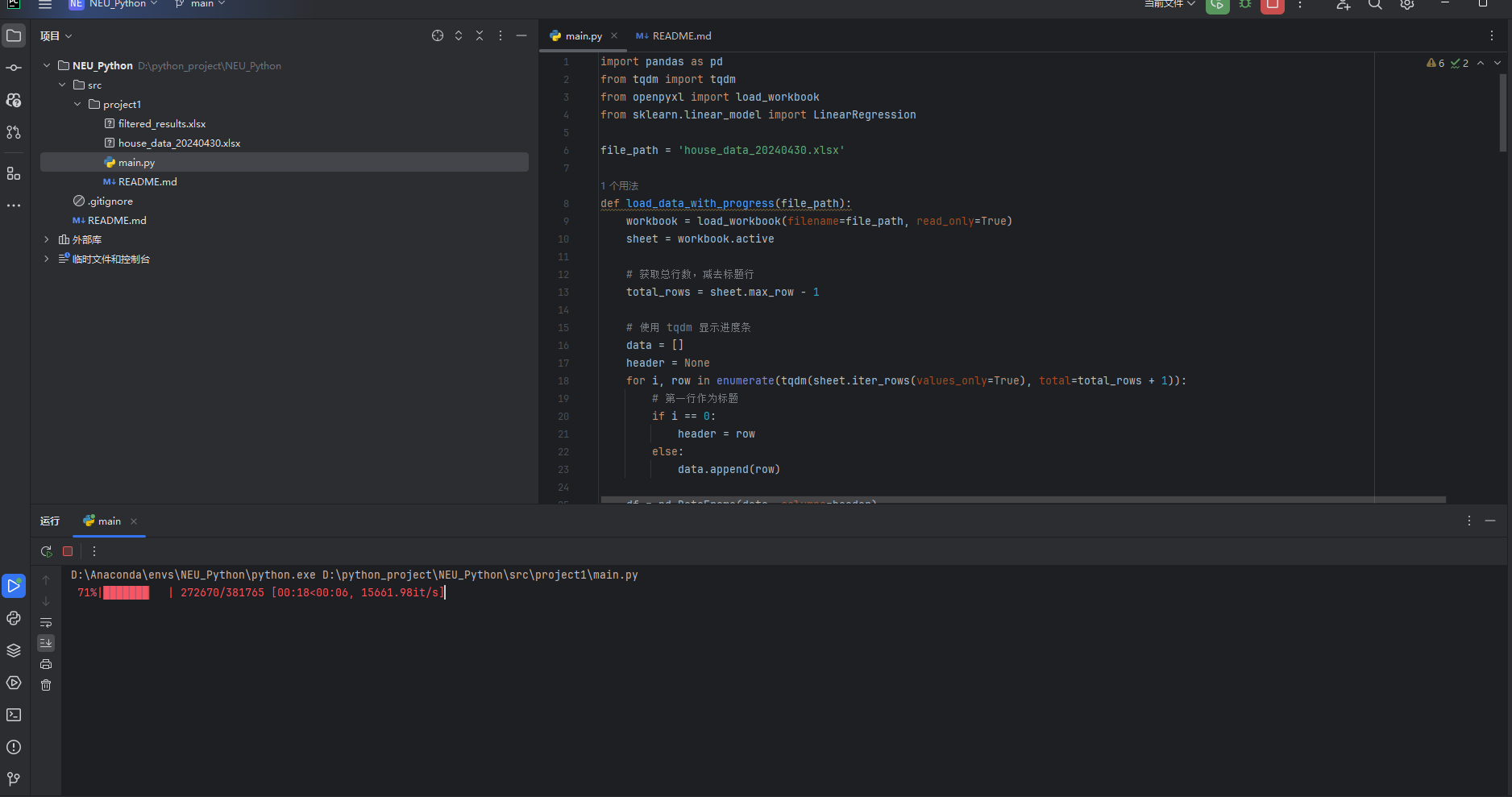


# 三、实验结果

## 1.数据读取

在运行main.py之后，等待10s左右，控制台出现进度读条，由于数据量较大（381765行），读取所需时间较长。

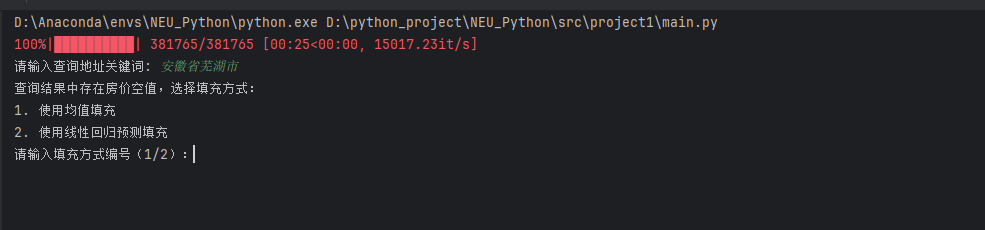




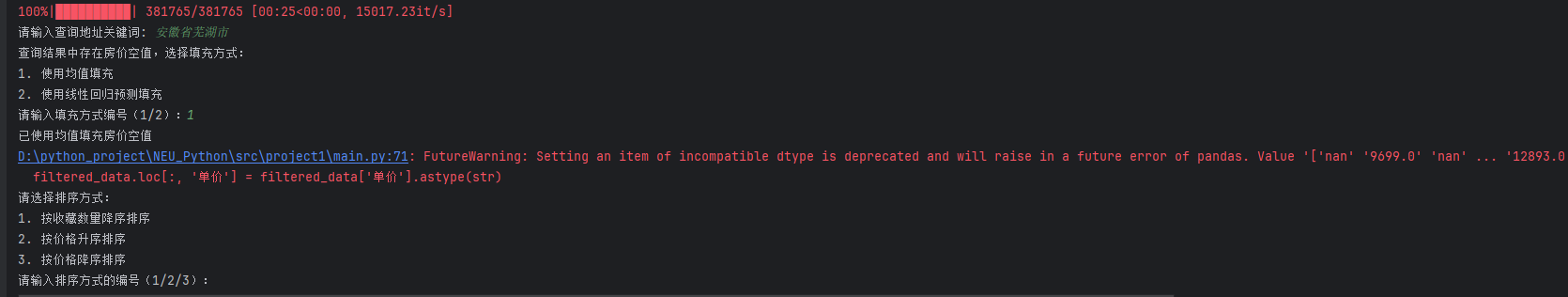
## 2.数据查询（安徽省芜湖市为例）

### 2.1 均值填充

然后输入查询地址关键词，以我的家乡（安徽省芜湖市为例）：

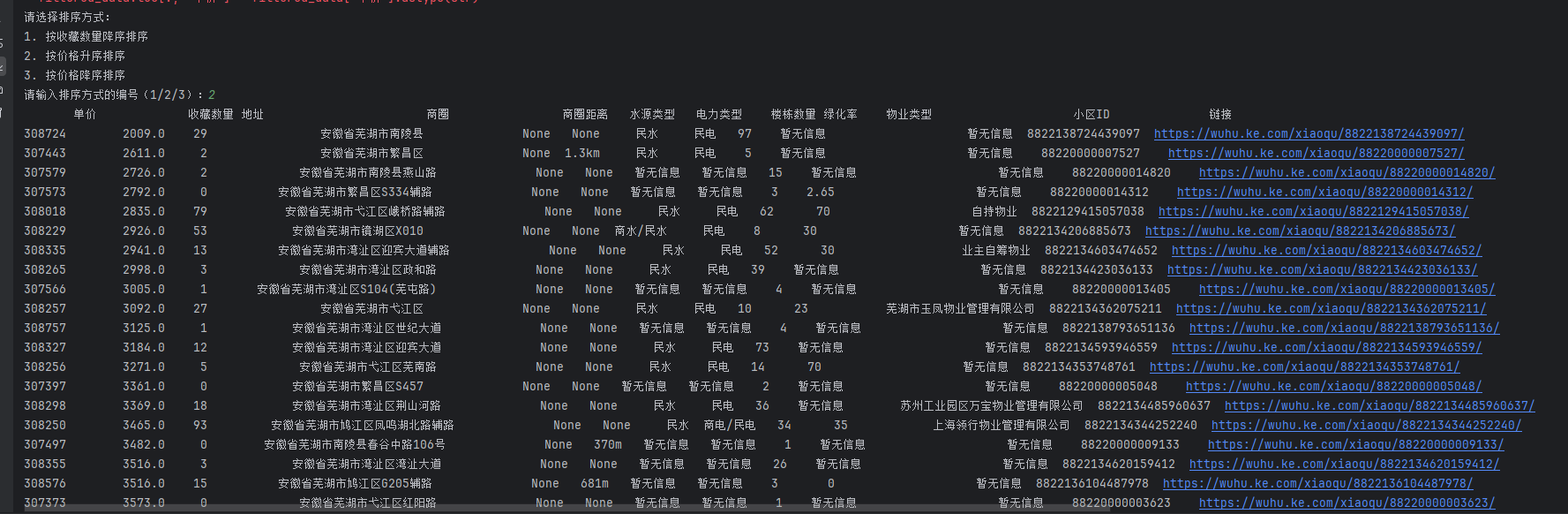


由于该数据截至到2024年4月30日，所以有部分房价数据空缺，给出两种填充方式，一种是以地区均值代替，一种使用简单的线性回归进行预测，特征设置为楼栋数量以及收藏数量(此处选择均值填充)：

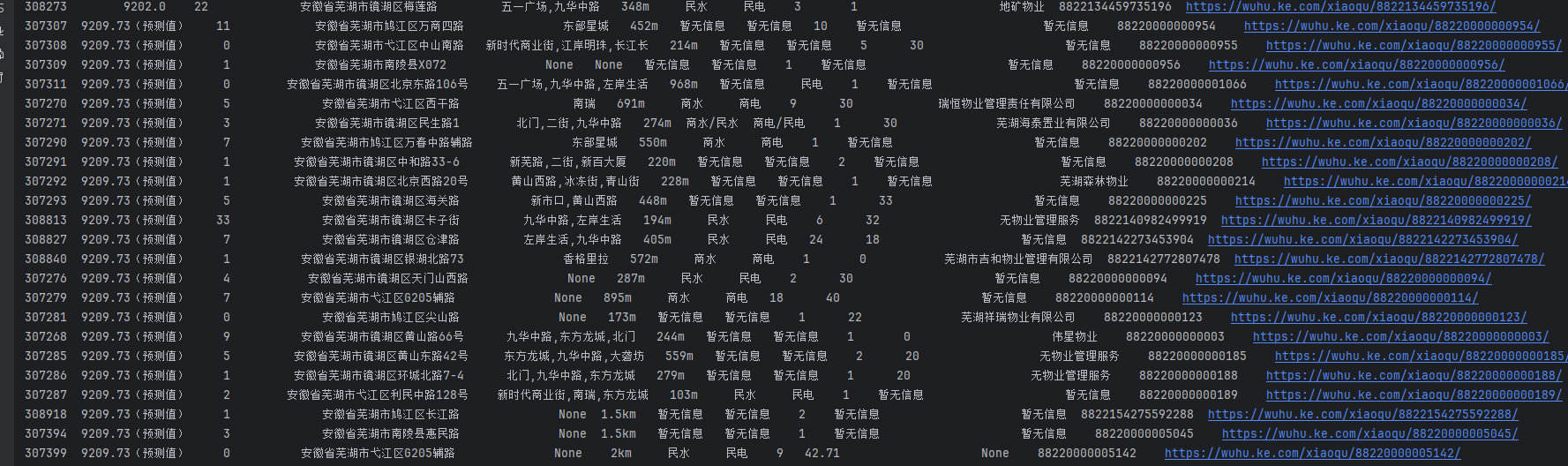


然后选择排序方式（此处选择按价格升序排序），依次列出

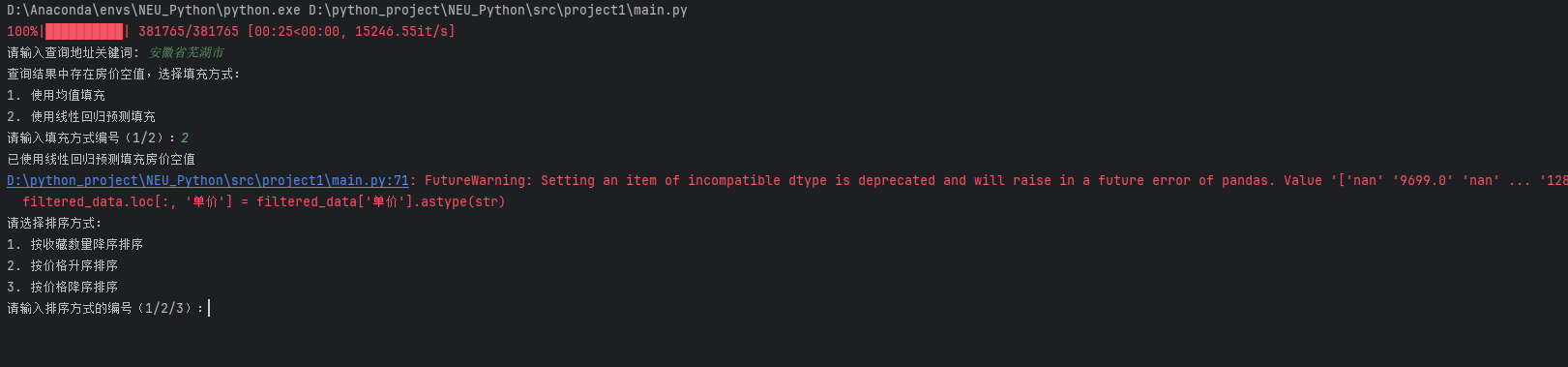
单价 收藏数量 地址 商圈 商圈距离 水源类型 电力类型 楼栋数量绿化率 物业类型 小区ID 链接：



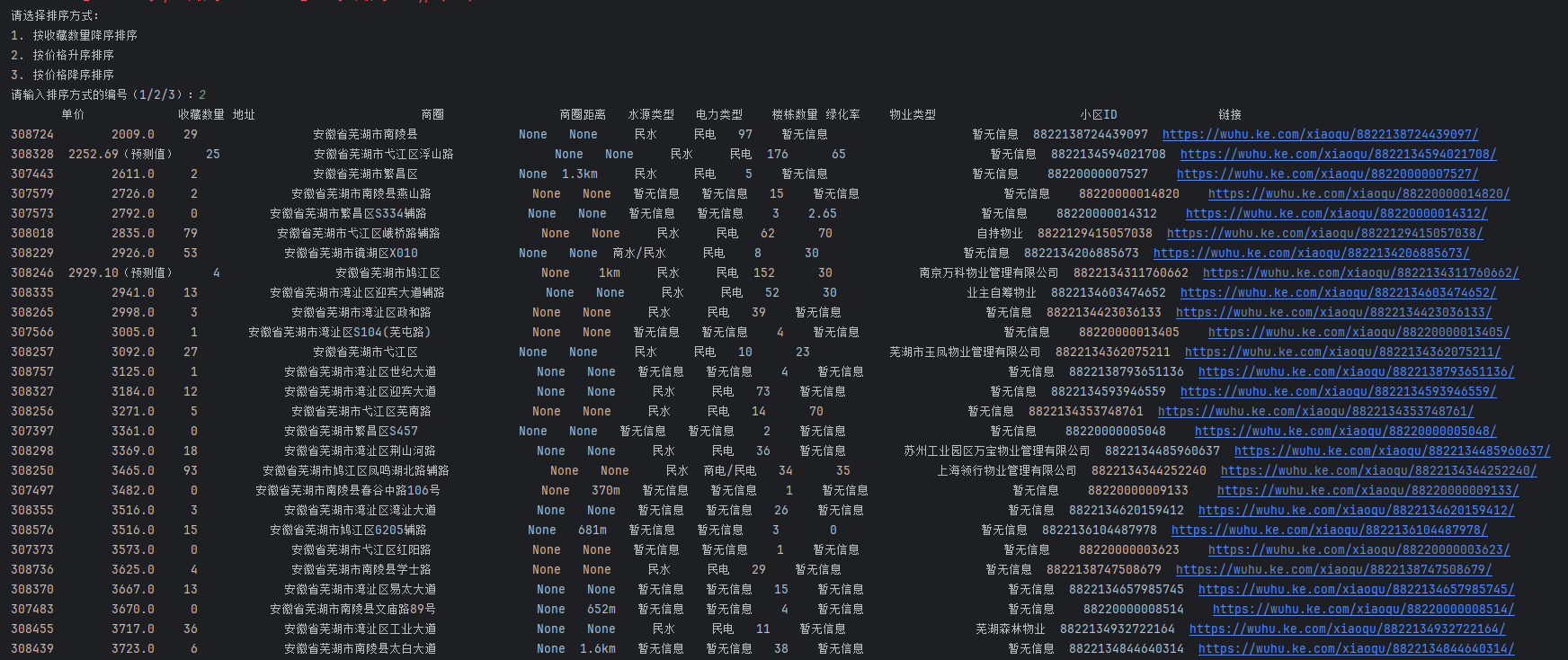
并且预测的房价也会标为（预测值）：

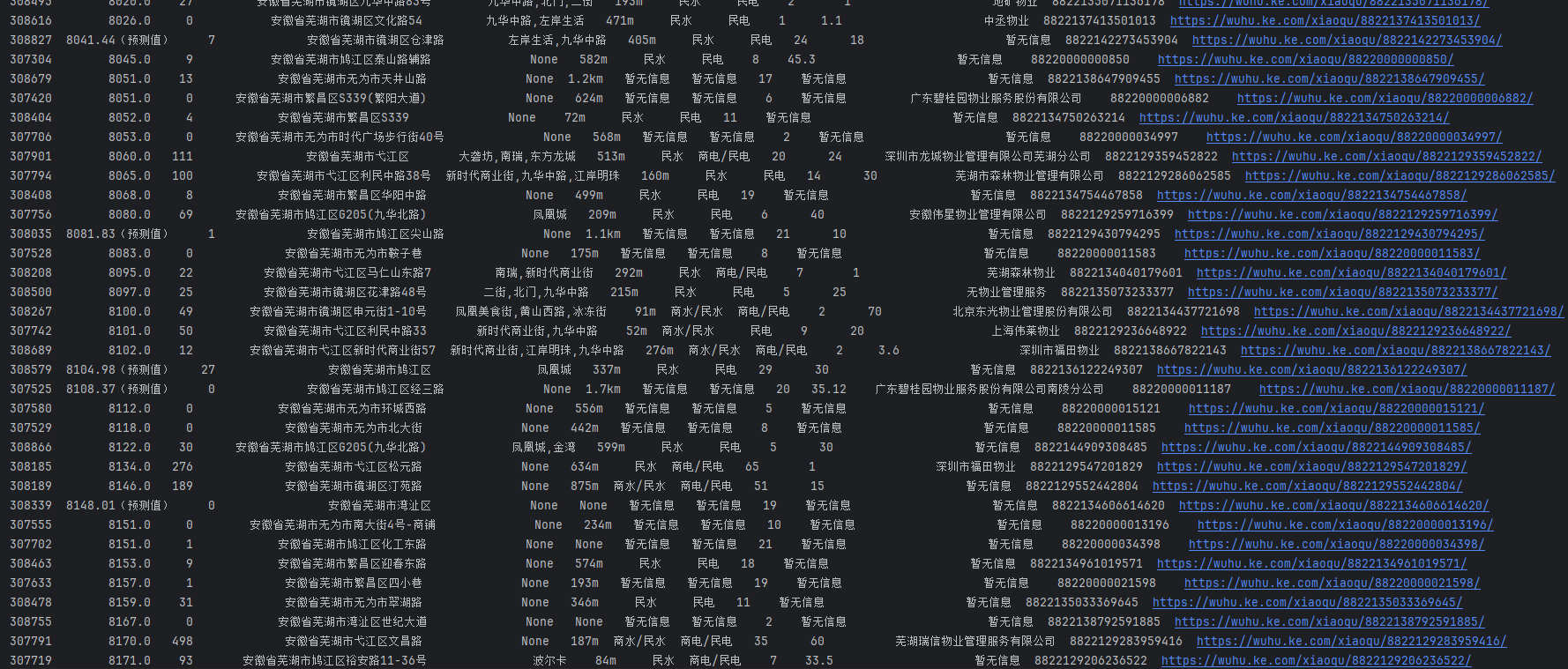


### 2.2 线性回归预测

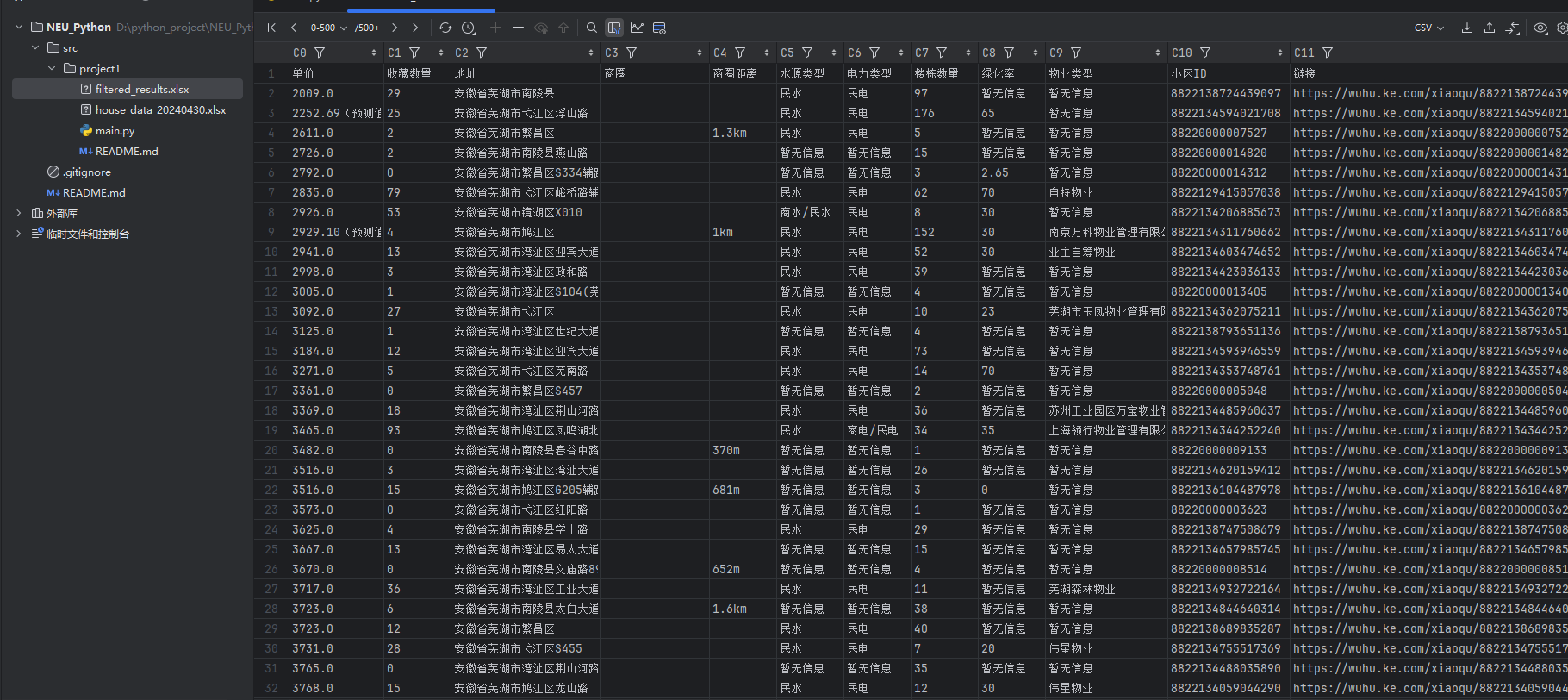


在此处可以较为明显的看到预测值：

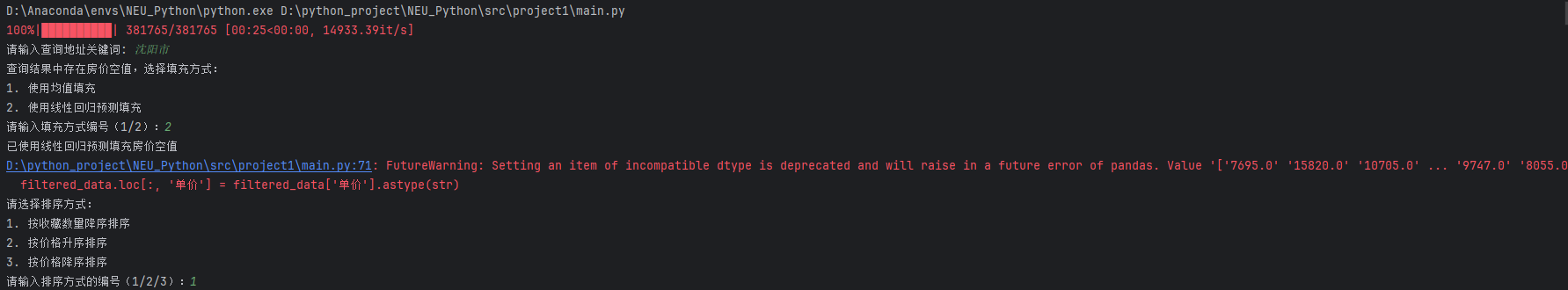


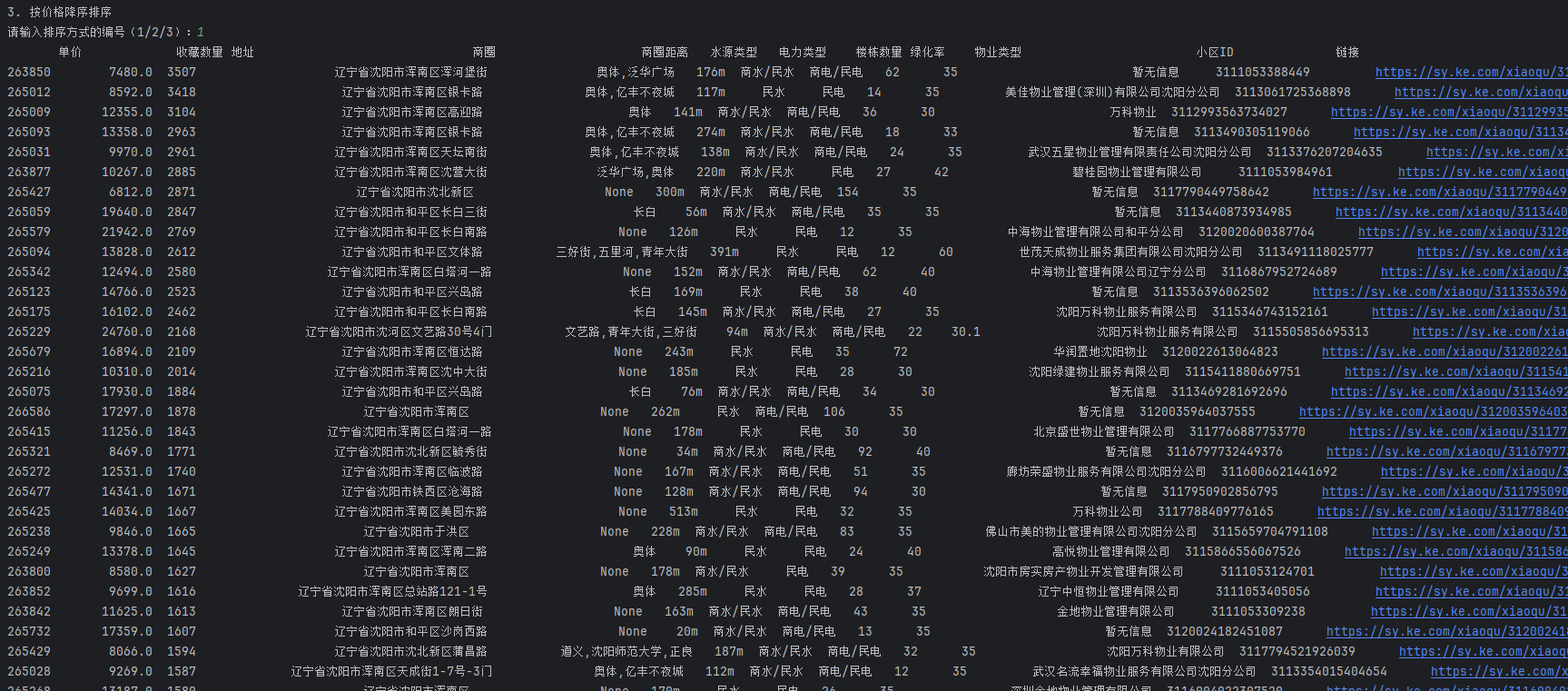


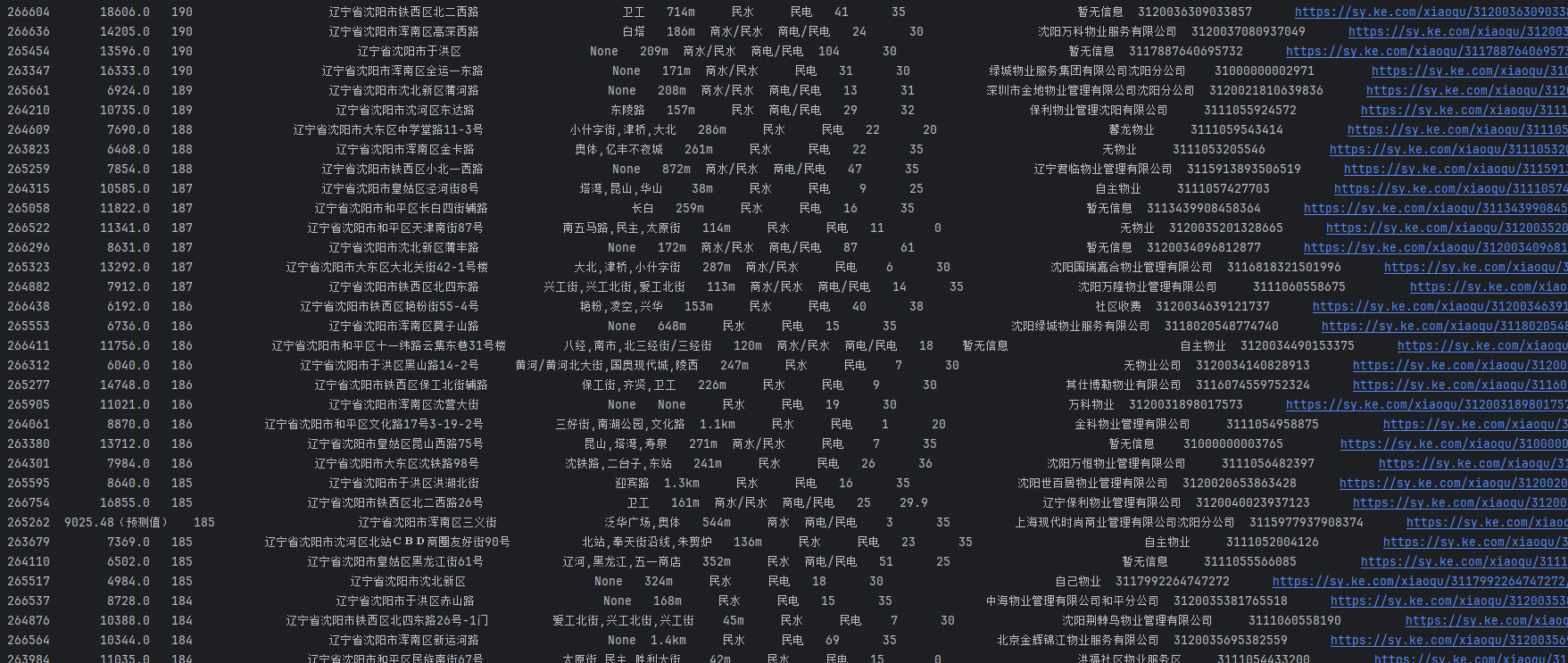
## 3.数据保存到文件

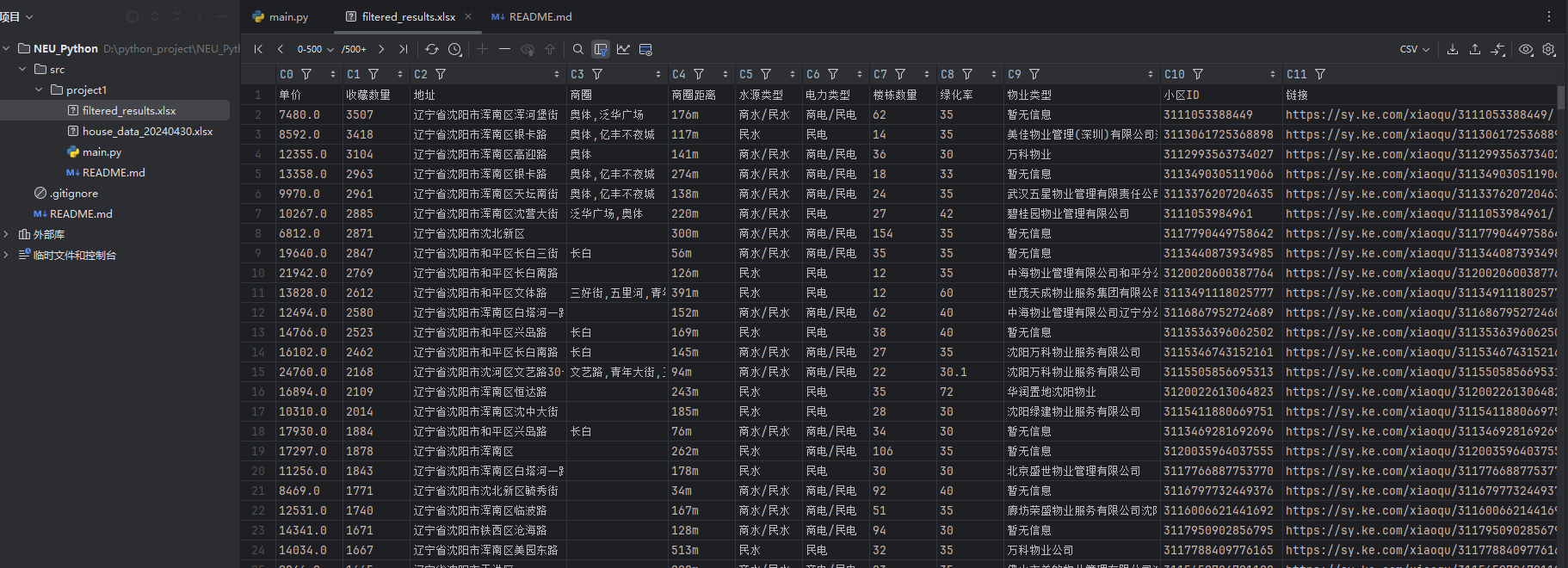


## 4.沈阳房价查询结果示例(按照收藏数量排序)









# 四、源码解读

 **数据加载与处理**

* 使用 load\_data\_with\_progress() 函数从 Excel 文件加载二手房数据，显示加载进度条并生成 Pandas 数据框。

 **列名翻译**

* 使用字典将英文列名翻译为中文名称，便于理解和后续操作。

 **条件查询**

* 根据用户输入的地址关键词筛选数据，返回符合条件的房源记录。

 **缺失值填充**

* 检测筛选结果中的房价缺失值。
* 提供两种填充方式：
  + **均值填充**：计算房价的平均值填充空缺。
  + **线性回归预测填充**：使用特征（如收藏数量、楼栋数量）训练模型，对缺失值进行预测。

 **数据排序**

* 提供用户选择的排序方式：
  + 按收藏数量降序。
  + 按房价升序。
  + 按房价降序。
* 默认排序为收藏数量降序。

 **结果输出**

* 显示筛选和排序后的数据，支持按指定的关键列输出。
* 保存最终结果到 Excel 文件 filtered\_results.xlsx。

 **用户交互**

* 控制台提示用户输入查询关键词、选择填充方式及排序方式。
* 显示操作结果及错误处理提示。

## 源代码：

import pandas as pd  
from tqdm import tqdm  
from openpyxl import load\_workbook  
from sklearn.linear\_model import LinearRegression  
  
file\_path = 'house\_data\_20240430.xlsx'  
  
def load\_data\_with\_progress(file\_path):  
 workbook = load\_workbook(filename=file\_path, read\_only=True)  
 sheet = workbook.active  
  
 # 获取总行数，减去标题行  
 total\_rows = sheet.max\_row - 1  
  
 # 使用 tqdm 显示进度条  
 data = []  
 header = None  
 for i, row in enumerate(tqdm(sheet.iter\_rows(values\_only=True), total=total\_rows + 1)):  
 # 第一行作为标题  
 if i == 0:  
 header = row  
 else:  
 data.append(row)  
  
 df = pd.DataFrame(data, columns=header)  
 return df  
  
house\_data = load\_data\_with\_progress(file\_path)  
  
column\_translation = {  
 'link': '链接',  
 'buildingCount': '楼栋数量',  
 'buildingType': '建筑类型',  
 'electricType': '电力类型',  
 'favCount': '收藏数量',  
 'greenRate': '绿化率',  
 'heatingType': '供暖类型',  
 'name': '名称',  
 'property': '物业类型',  
 'resblockId': '小区ID',  
 'shopDistance': '商圈距离',  
 'unitPrice': '单价',  
 'waterType': '水源类型',  
 'f地址': '地址',  
 '商圈': '商圈',  
 '省份': '省份',  
 '城市': '城市',  
 '区县': '区县',  
 '街镇': '街镇'  
}  
  
house\_data.rename(columns=column\_translation, inplace=True)  
  
pd.set\_option('display.max\_columns', None)  
pd.set\_option('display.max\_rows', None)  
pd.set\_option('display.width', 1000)  
pd.set\_option('display.colheader\_justify', 'left')  
  
query\_address = input("请输入查询地址关键词: ")  
  
filtered\_data = house\_data[house\_data['地址'].str.contains(query\_address, na=False)]  
  
# 填充房价空值  
if filtered\_data['单价'].isna().any():  
 print("查询结果中存在房价空值，选择填充方式：")  
 print("1. 使用均值填充")  
 print("2. 使用线性回归预测填充")  
 fill\_option = input("请输入填充方式编号（1/2）：")  
  
  
 filtered\_data.loc[:, '单价'] = filtered\_data['单价'].astype(str)  
  
 if fill\_option == '1':  
 mean\_price = filtered\_data['单价'].astype(float).mean()  
 filtered\_data.loc[filtered\_data['单价'] == 'nan', '单价'] = f"{mean\_price:.2f}（预测值）"  
 print("已使用均值填充房价空值")  
  
 elif fill\_option == '2':  
 # 使用线性回归模型预测房价空值  
 # 提取特征列（如收藏数量、楼栋数量等）和目标列（单价）  
 features = filtered\_data[['收藏数量', '楼栋数量']].copy() # 示例特征  
 features['收藏数量'] = features['收藏数量'].fillna(features['收藏数量'].mean())  
 features['楼栋数量'] = features['楼栋数量'].fillna(features['楼栋数量'].mean())  
  
 # 拆分有房价和无房价的行  
 known\_price = filtered\_data[filtered\_data['单价'] != 'nan']  
 unknown\_price = filtered\_data[filtered\_data['单价'] == 'nan']  
  
 # 训练模型  
 model = LinearRegression()  
 model.fit(known\_price[['收藏数量', '楼栋数量']], known\_price['单价'].astype(float))  
  
 # 预测并填充缺失值，添加“（预测值）”标记  
 predicted\_prices = model.predict(unknown\_price[['收藏数量', '楼栋数量']])  
 filtered\_data.loc[filtered\_data['单价'] == 'nan', '单价'] = [f"{price:.2f}（预测值）" for price in predicted\_prices]  
 print("已使用线性回归预测填充房价空值")  
  
 else:  
 print("无效选择，未进行填充")  
else:  
 print("查询结果中无房价空值")  
  
# 获取用户选择的排序方式  
print("请选择排序方式：")  
print("1. 按收藏数量降序排序")  
print("2. 按价格升序排序")  
print("3. 按价格降序排序")  
sort\_option = input("请输入排序方式的编号（1/2/3）：")  
  
# 根据用户选择的排序方式排序数据  
if sort\_option == '1':  
 sorted\_data = filtered\_data.sort\_values(by='收藏数量', ascending=False)  
elif sort\_option == '2':  
 sorted\_data = filtered\_data.sort\_values(by='单价', ascending=True, key=lambda x: pd.to\_numeric(x.str.replace('（预测值）', ''), errors='coerce'))  
elif sort\_option == '3':  
 sorted\_data = filtered\_data.sort\_values(by='单价', ascending=False, key=lambda x: pd.to\_numeric(x.str.replace('（预测值）', ''), errors='coerce'))  
else:  
 print("无效的输入，默认按收藏数量降序排序")  
 sorted\_data = filtered\_data.sort\_values(by='收藏数量', ascending=False)  
  
# 指定要输出的列，包括收藏数量  
output\_columns = [  
 '单价', '收藏数量', '地址', '商圈', '商圈距离', '水源类型', '电力类型',  
 '楼栋数量', '绿化率', '物业类型', '小区ID', '链接'  
]  
  
  
result = sorted\_data[output\_columns]  
print(result)  
result.to\_excel("filtered\_results.xlsx", index=False)