****

**2023～2024学年第2学期**

**《大学计算机》课程报告**

**小组组长 2308010419陈雅文**

**组 员 1 2308010330邹莉**

**组 员 2 2308010418陈瑞**

**组 员 3**

|  |  |
| --- | --- |
| **得 分** |  |
| **评阅教师** | 王雪瑞 |

**2024 年 4 月 28 日**

**说 明**

1、《大学计算机》课程是一门要求学生能够运用数据采集工具或者python语言进行数据获取及处理能力的综合训练课程，目的是提高学生分析问题、建立模型、解决问题能力。

2、课程报告成绩占课程总成绩的40%。

3、关于课程报告选题问题：选题应选取与学生身份、日常生活、国计民生等领域相关的题目，应积极向上、有创意。

4、课程报告要求每个学生把自己实现的实验结果展示出来，重点描述感兴趣的某个领域数据进行获取、分析，并以可视化的方式将分析结果展示。因此，课程报告主要包括任务描述、数据获取、数据处理与分析及可视化展示等数据处理过程的内容。

**5、课程结束后，小组组长需要提交（1）源数据；（2）程序源代码（.py文件）（无代码者无需提供）；（3）报告电子版；（4）演示视频（不超过3分钟）等相关资料到惟真课堂。逾期未提交相关资料者不能参加成绩评定。**

6、课程报告要求参考本报告模板撰写，可根据具体情况适当调整，但文档结构要完整、格式排版美观、条理清晰、论述准确。

**7、文档撰写不符合要求，或者抄袭他人作品，课程报告记零分。**

|  |  |
| --- | --- |
| **课程报告评分参考** | |
| 1. 文档结构完整、格式排版美观、文字描述准确（10~15分） 2. 数据处理过程   （1）数据获取：过程及方法描述清楚准确（20分）  （2）数据处理与数据分析：能够针对获取的数据做合理的数据处理和分析（30分）  （3）数据可视化：运用多种图表展示，类型选择恰当，要求美观、完整、清晰（30分）  （4）数据管理：运用数据库管理，类型选择恰当，要求具备数据增删改查功能（5~10分） | |
| **教师评语** | |
|  | |
| **评阅教师** | **王雪瑞** |

# **题目：大学性价比数据处理分析**

1. **任务描述**

构建一个大学性价比分析和管理系统，管理大学的基本信息，如来源地区、在山东的分数线、不同标准下的评分，并通过多种可视化手段进行展示。作品的核心功能是通过统计分析，计算出各个大学的性价比指数，为山东考生填报志愿提供一定参考。

**二、数据获取**

1、数据描述

（1）数据来源：

校友会大学排名页面，如图1所示。

网站来源http://www.gaokao.com/e/20210328/606032dc1b634.shtml

软科大学排名页面，如图2所示。

网站来源https://www.shanghairanking.cn/rankings/bcur/202411

2023各大学在山东录取分数线页面，如图3所示。

网站来源https://www.gaokao.cn/lineschool



图1 2021年校友会中国双一流大学排名140强\_高考网



图2 2024中国大学排名\_软科



图3 2023各大学在山东录取分数线\_掌上高考

1. 数据获取：校友会大学排名页面（排名、学校名称、评分）、软科大学排名页面信息（排名、学校名称、省份、评分）、2023各大学在山东录取分数线页面（排名、学校名称、地区、在山东的分数线）并将这些信息写入Excel表格。

2、使用工具

Python的Idle编辑器，Pycharm编辑器

3、数据获取步骤

（1）获取校友会排名的步骤

1. 设置URL，初始化data
2. 编写getHTMLText(url)函数。使用requests.get()方法获取网页的HTML内容[1]。函数内部设置了超时时间（timeout=30），并捕获了可能出现的异常。如果请求成功，则返回HTML文本；否则返回空字符串。
3. 编写fillUnivList(ulist, html)函数使用BeautifulSoup库解析HTML内容[1]。首先找到HTML中的<tbody>标签，并遍历它的子元素[3]。对于每一个<tr>标签（代表表格的一行），它检查是否是一个有效的标签（bs4.element.Tag的实例）。如果是有效的<tr>标签，则进一步获取该标签下的所有<td>标签。
4. 将它们的内容添加到ulist列表中（去掉空格）。每个大学的信息以列表的形式存储。
5. 编写printUnivList(ulist, 140)函数负责打印，并将这些信息添加到全局变量data中。打印时使用了格式化字符串，确保输出整齐。
6. 最后创建了一个Pandas DataFrame对象df[1]，获取data中的内容，并保存到Excel文件：
7. 获取软科排名的步骤
8. 复制页面内容，存入文本文件
9. 使用open函数以读取模式（'r'）打开分数线2023.txt文件[1]，并指定编码为utf-8。
10. 读取文件内容，遍历文件的每一行。
11. 对于每所大学的信息，根据cnt的值将数据分割到不同的部分（排名、大学名称、省份和评分）。当识别到评分信息时，将前四位添加到a列表中，并将a添加到a2列表中，然后重置a和cnt。
12. 使用pd.DataFrame函数将a2列表转换为Pandas DataFrame，并指定列名。
13. 输出到Excel文件。
14. 获取在山东的分数线的步骤
15. 基本同上，只不过原网站没有排名，又增加了一个n变量来记录排名

4、程序代码

（1）获取数据及输出校友会前226位学校进行验证的代码及截图如图4所示。

文件夹名称：大学计算机大作业（陈雅文，陈瑞，邹莉）

文件名称: 爬取大学排名校友会+分数线+软科

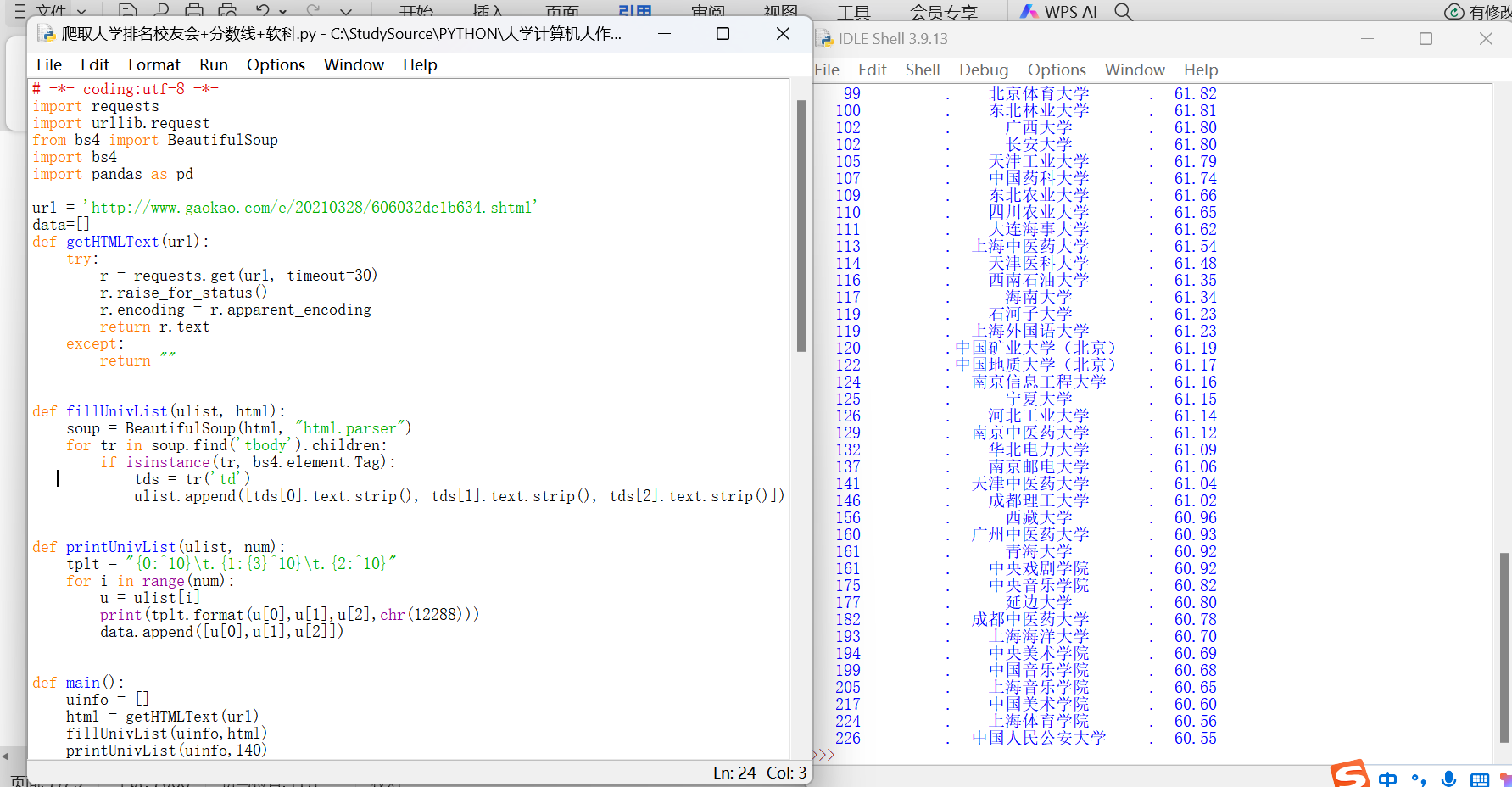


图4 爬虫程序及运行结果

1. 将获得全部大学信息存储到Excel表格中，完整代码如下：

# -\*- coding:utf-8 -\*-

import requests

import urllib.request

from bs4 import BeautifulSoup

import bs4

import pandas as pd

url = 'http://www.gaokao.com/e/20210328/606032dc1b634.shtml'

data=[]

def getHTMLText(url):

try:

r = requests.get(url, timeout=30)

r.raise\_for\_status()

r.encoding = r.apparent\_encoding

return r.text

except:

return ""

def fillUnivList(ulist, html):

soup = BeautifulSoup(html, "html.parser")

for tr in soup.find('tbody').children:

if isinstance(tr, bs4.element.Tag):

tds = tr('td')

ulist.append([tds[0].text.strip(), tds[1].text.strip(), tds[2].text.strip()])

def printUnivList(ulist, num):

tplt = "{0:^10}\t.{1:{3}^10}\t.{2:^10}"

for i in range(num):

u = ulist[i]

print(tplt.format(u[0],u[1],u[2],chr(12288)))

data.append([u[0],u[1],u[2]])

def main():

uinfo = []

html = getHTMLText(url)

fillUnivList(uinfo,html)

printUnivList(uinfo,140)

if \_\_name\_\_ == "\_\_main\_\_":

main()

filename = '大学排行榜校友会2021.xlsx'

df = pd.DataFrame(data[1:], columns=data[0])

df.to\_excel(filename, index=False)

with open('分数线2023.txt', 'r', encoding='utf-8') as file:

cnt=1

n=1

a=[]

a2=[]

for line in file:

if line.strip():

if cnt==1:

a.append(n)

a.append(line.strip())

cnt+=1

elif cnt==2:

a.append(line.strip())

cnt+=1

elif cnt==3:

b=line.split('：')

#print(b)

if len(b)==2:

a.extend([b[1][0:3]])

a2.append(a)

a=[]

cnt=1

n+=1

df = pd.DataFrame(a2,columns=['序号', '大学名称', '地区', '分数线'])

df.to\_excel('分数线2023.xlsx', index=False)

with open('软科2024.txt', 'r', encoding='utf-8') as file:

cnt=1

a=[]

a2=[]

for line in file:

if line.strip():

if cnt<5:

if cnt==1:

a.append(line.strip())

elif cnt==2:

a.append(line.strip())

cnt+=1

elif cnt==5:

b=line.split()

if len(b)==4 or len(b)==3:

a.extend([b[0],b[2]])

a2.append(a)

a=[]

cnt=1

else:

cnt+=1

elif cnt==6:

b=line.split()

a.extend([b[0],b[2]])

a2.append(a)

a=[]

cnt=1

df = pd.DataFrame(a2,columns=['排名', '大学名称', '省份', '评分'])

df.to\_excel('软科2024.xlsx', index=False)

5、数据保存到Excel文件，Excel文件截图如图5图6图7所示。

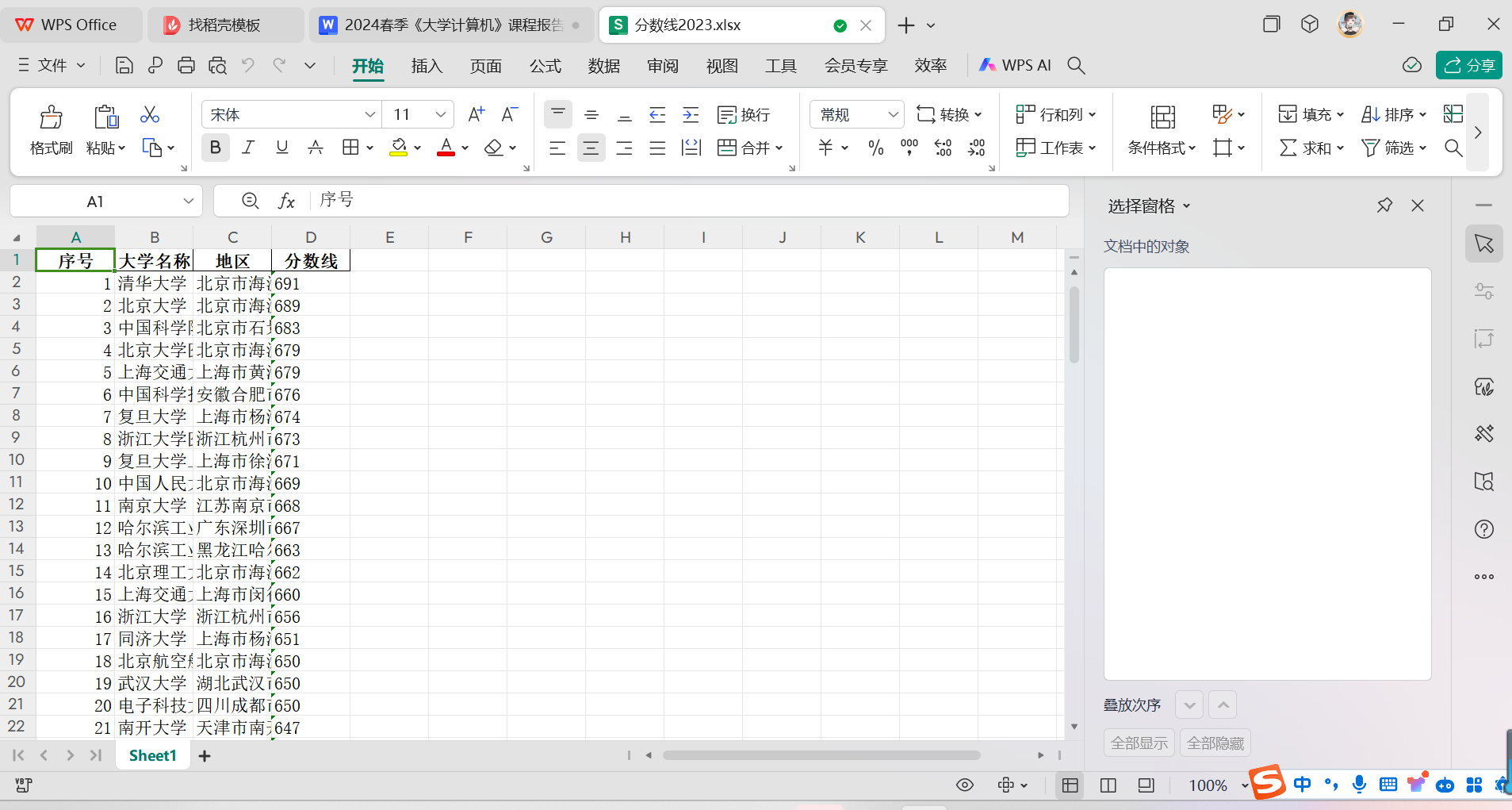


图5 关于2023分数线的Excel文件截图

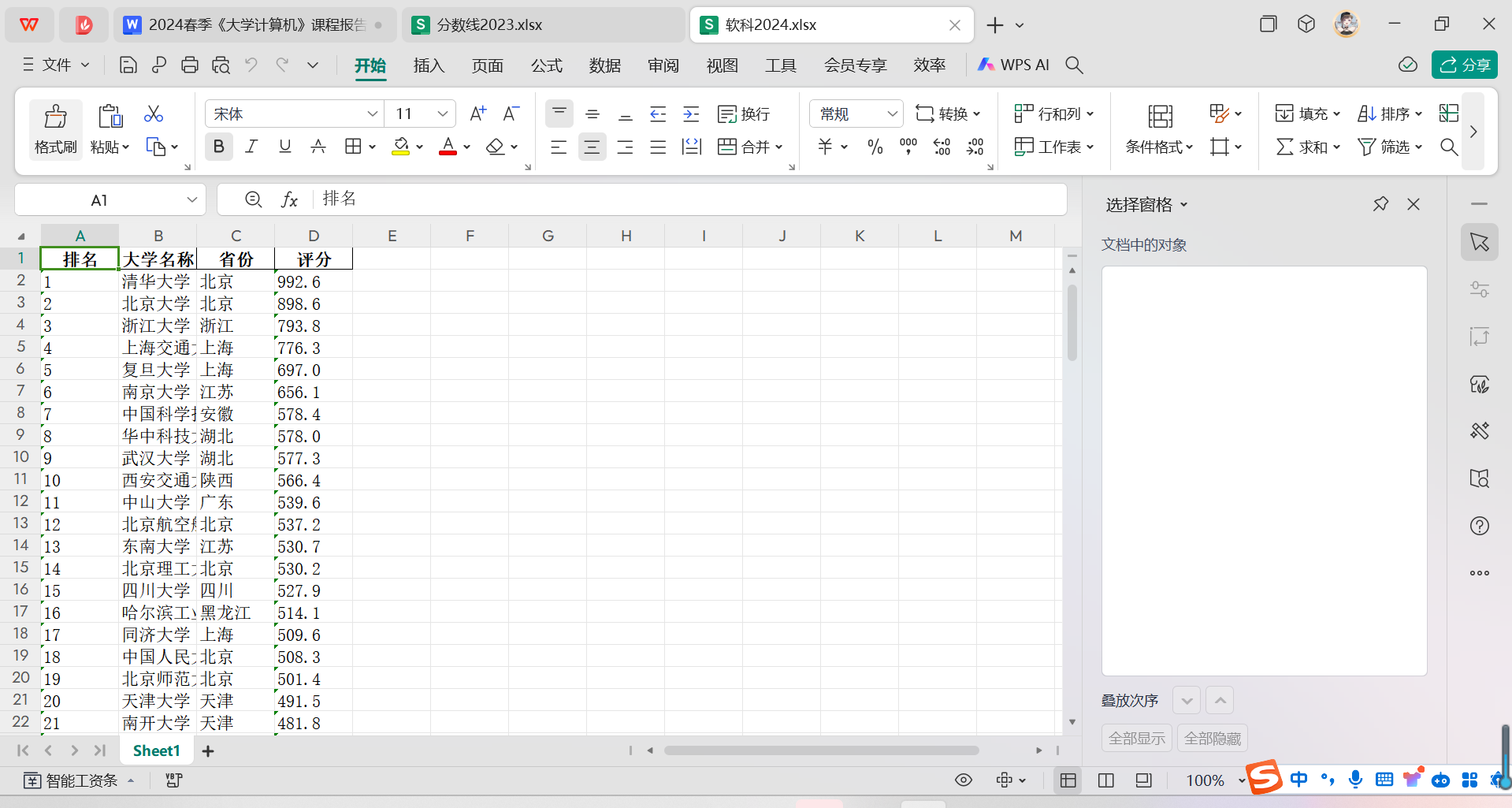


图6 关于2024软科排名数据的Excel文件截图

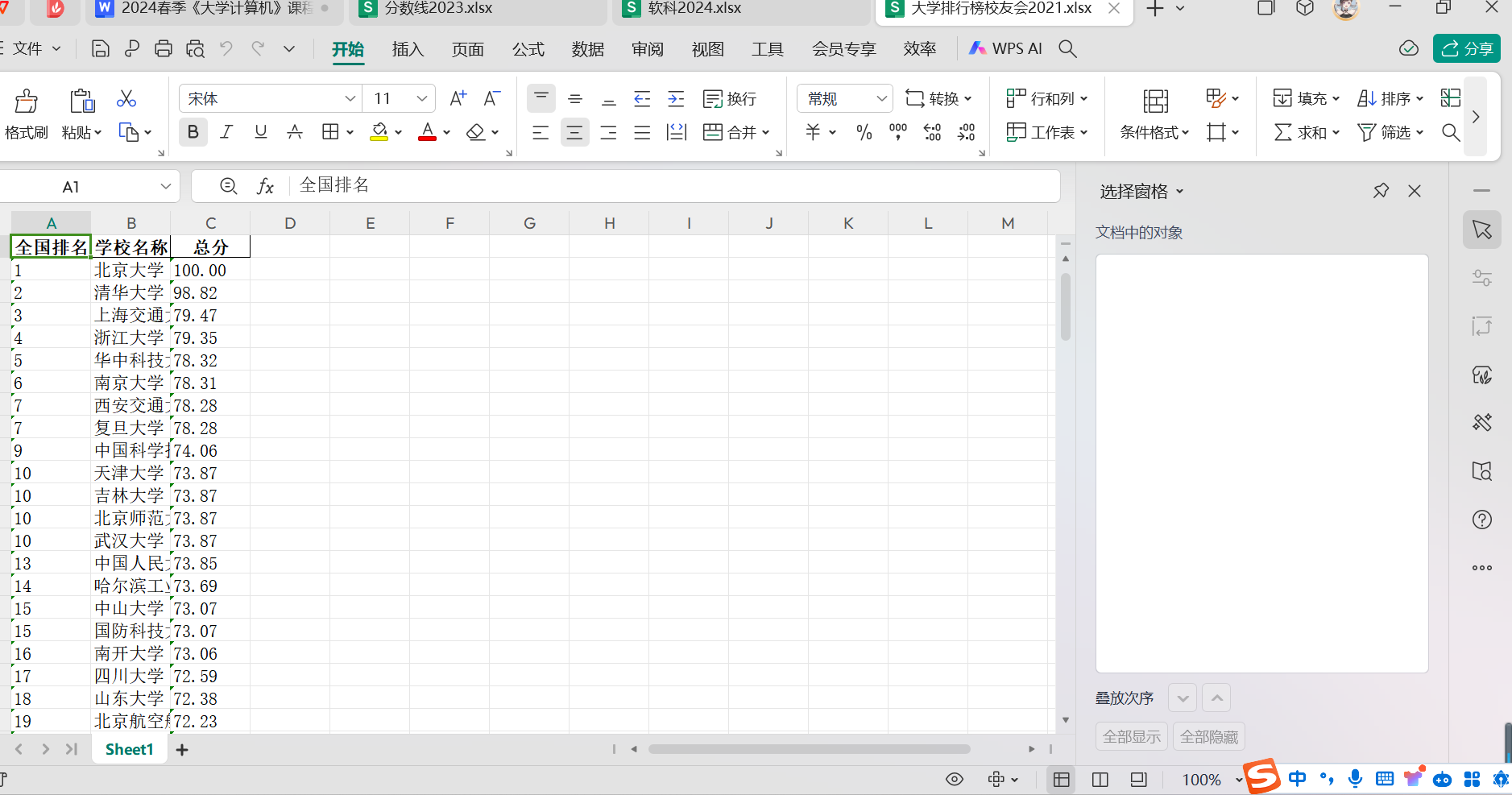


图7 关于2021校友会排名数据的Excel文件截图

**三、数据分析与可视化展示**

在数据分析与可视化展示的过程中，我使用了Python作为工具来执行以下任务：

1. 数据准备：

首先，我们获取了软科评分、校友会评分和分数线等相关信息并存为excel，上述文章中已经提到这一过程。

1. 饼图绘制：

使用Python的matplotlib库，我们绘制了饼图来展示软科评分前120名和分数线前120名高校的来源地区分布情况，并将来源山东省的部分做了突出处理，如图8图9所示。

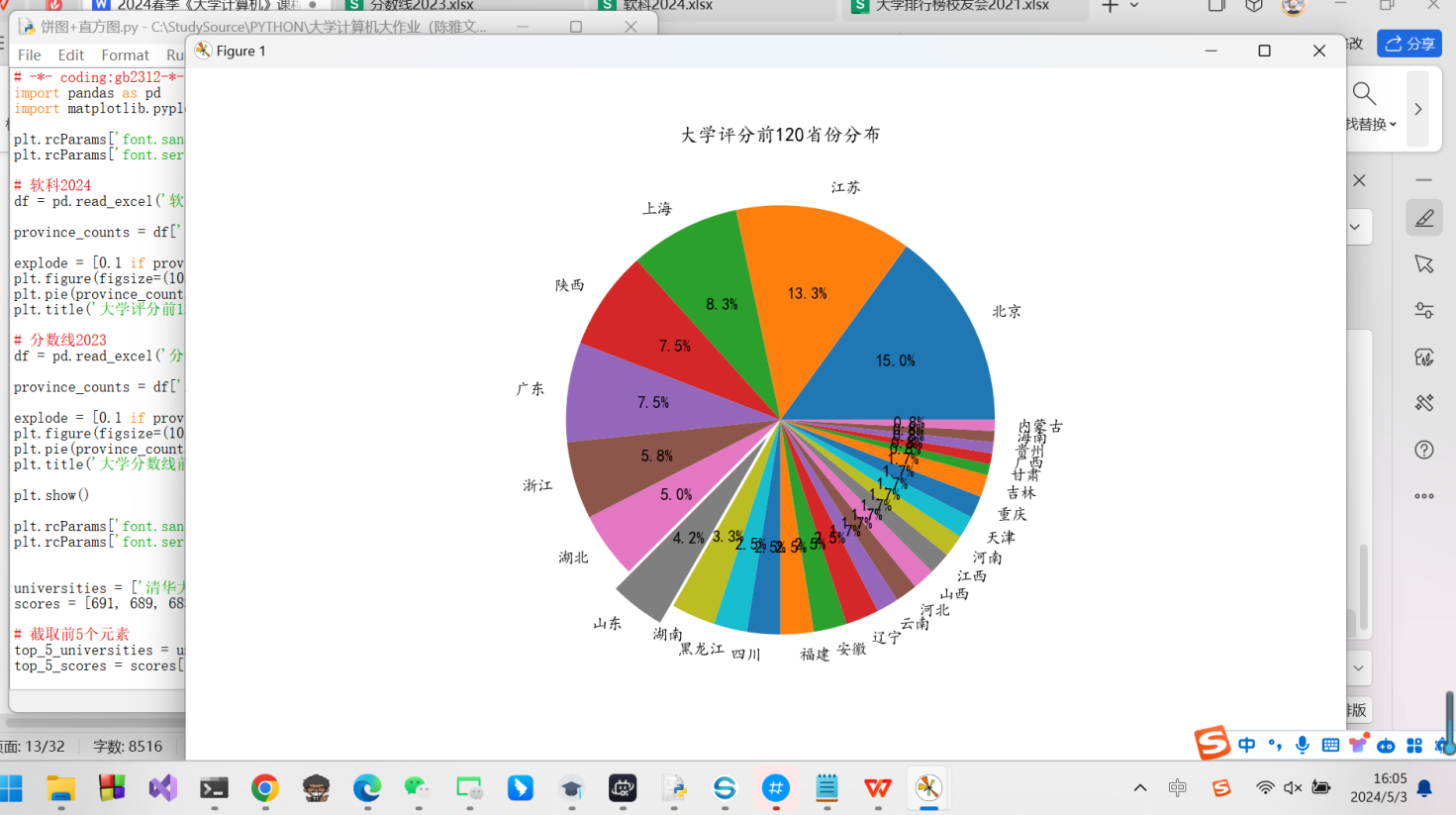


图8 评分前120地区分布截图

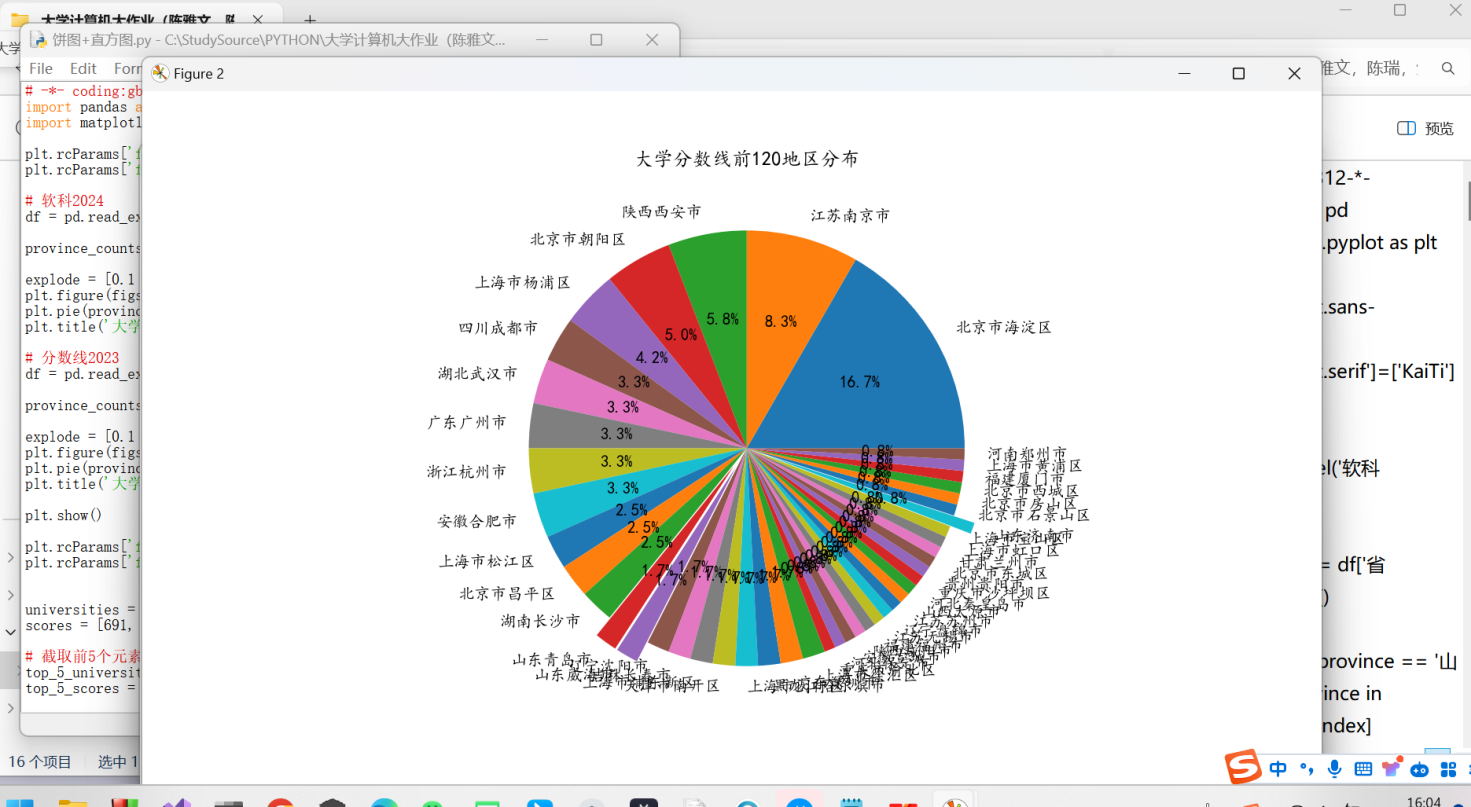


图9 分数线前120地区分布截图

通过饼图，我们可以直观地看出北京市在高等教育方面表现非常突出，无论是软科评分还是分数线占比都是第一，江苏省其次。山东的大学在软科评分前120中有5所，占到了4.2%，在分数线前120同样是5所，占到了4.2%。

1. 直方图绘制：

同样使用matplotlib库，我们绘制了直方图来展示软科评分和在山东分数线最高5名和最低5名高校的评分和分数线分布情况。如图10,11所示。

直方图的条形高度代表评分或分数线的数值，通过对比最高和最低排名的高校，我可以了解评分和分数线的分布范围和极端情况。

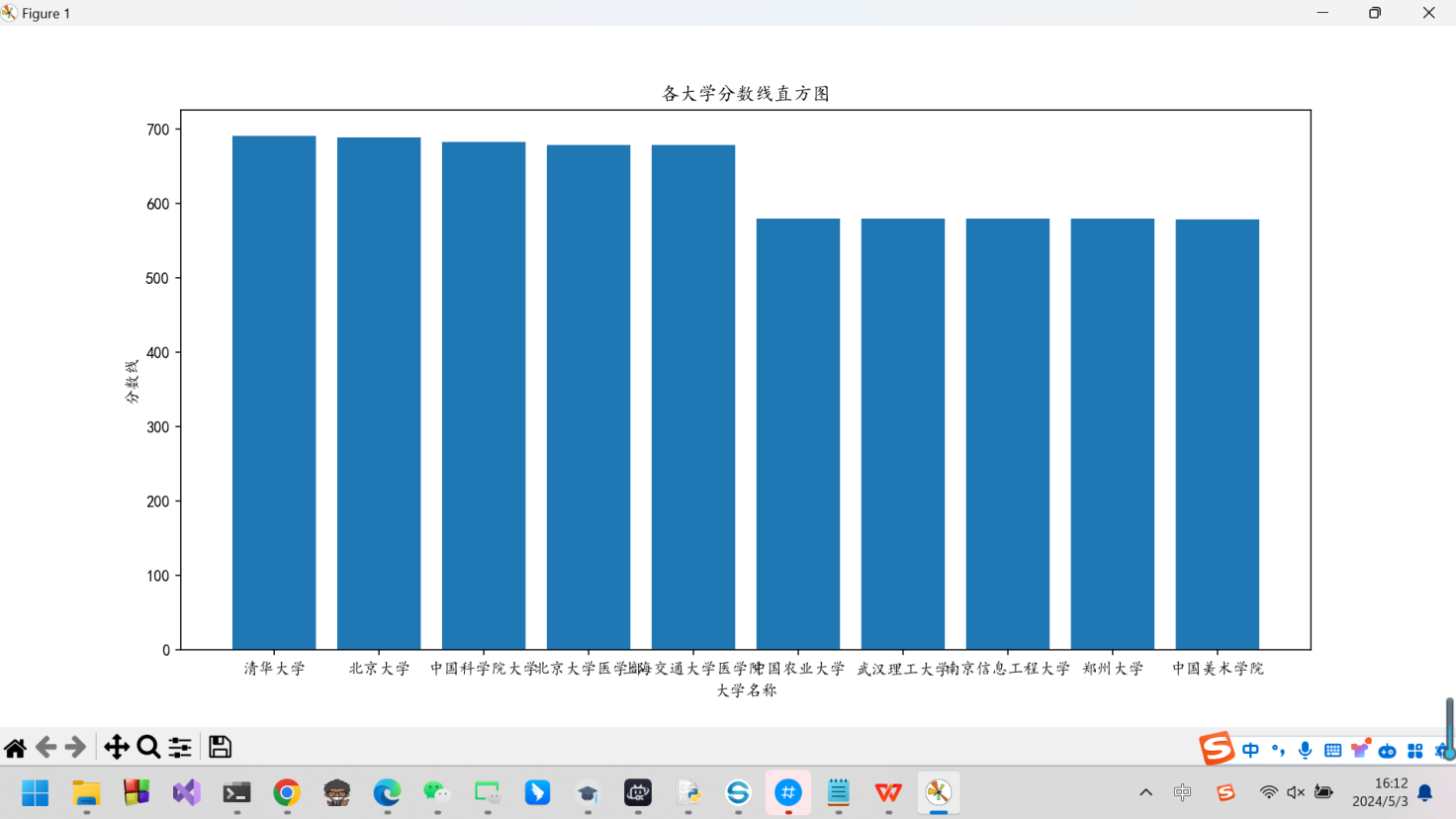


图10 分数线直方图截图

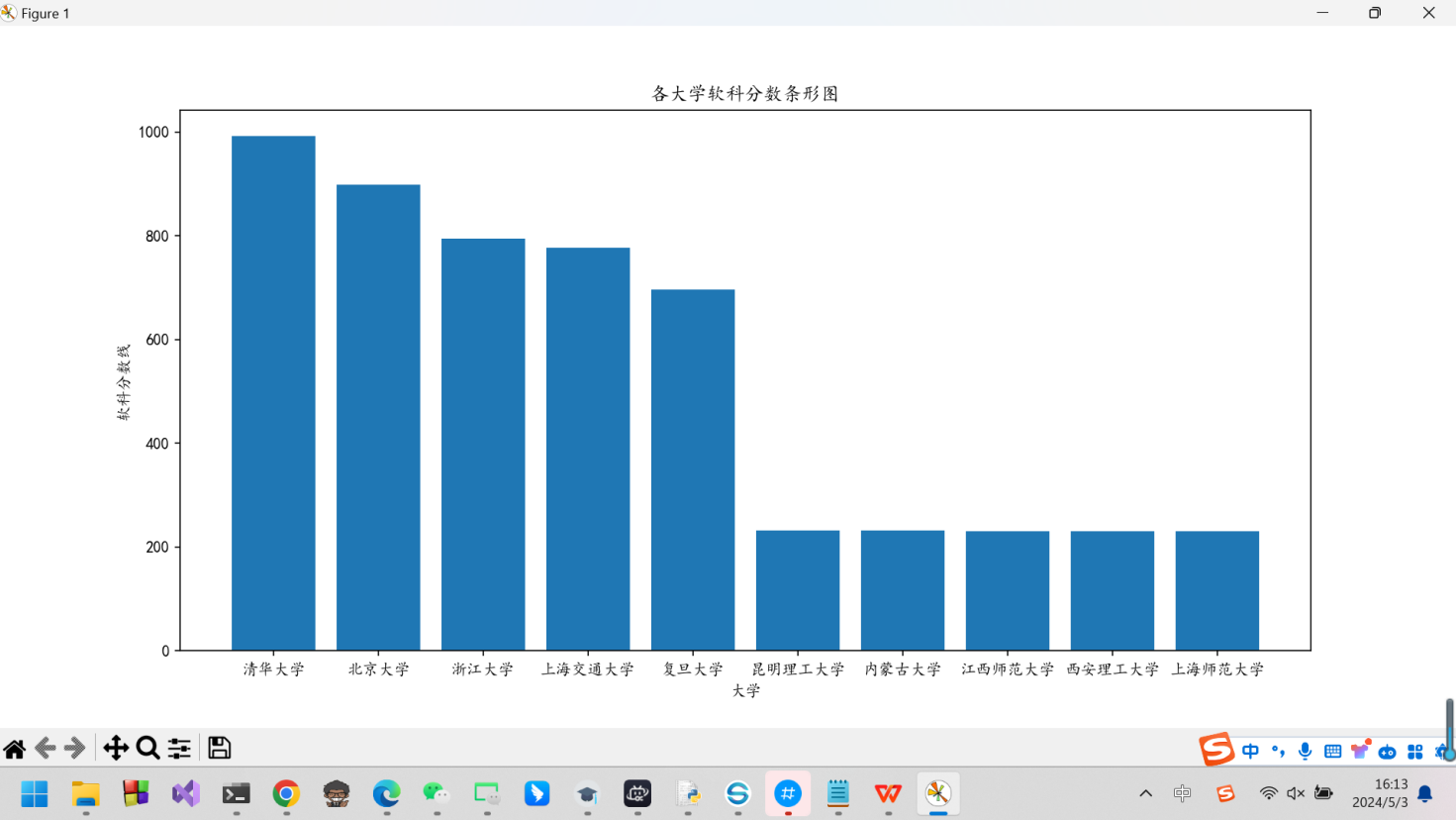


图11 软科评分直方图截图

在数据分析的过程中，我们注意到分数线的分布范围大致在580分到690分之间，极差约为110分。进一步观察发现，前五名高校的分数线都接近700分这一区域，而后五名高校的分数线则集中在接近600分的区域。五所高校之间的差距都非常小，即便是前五名与后五名相比，分数线的差异也仅为100分左右。体现了现在高考的竞争激烈，一分就可能决定命运。

而评分的差异性就相对较大，分布范围约在200分到1000分之间，极差约为800分。前五名之间的差异较大，后五名之间的差异较小。

1. 饼图和直方图完整代码截图如图12，图13所示

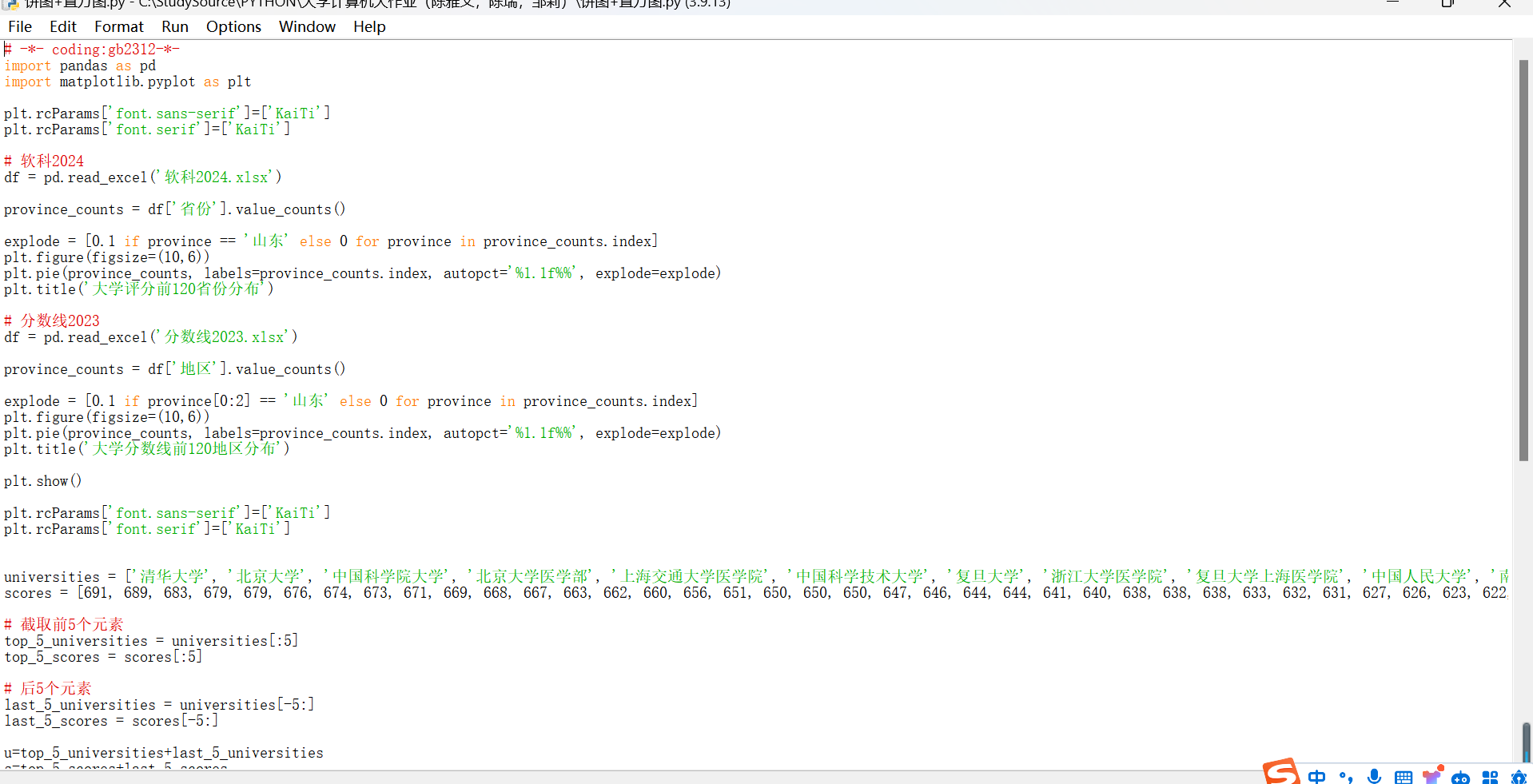


图11 饼图截图

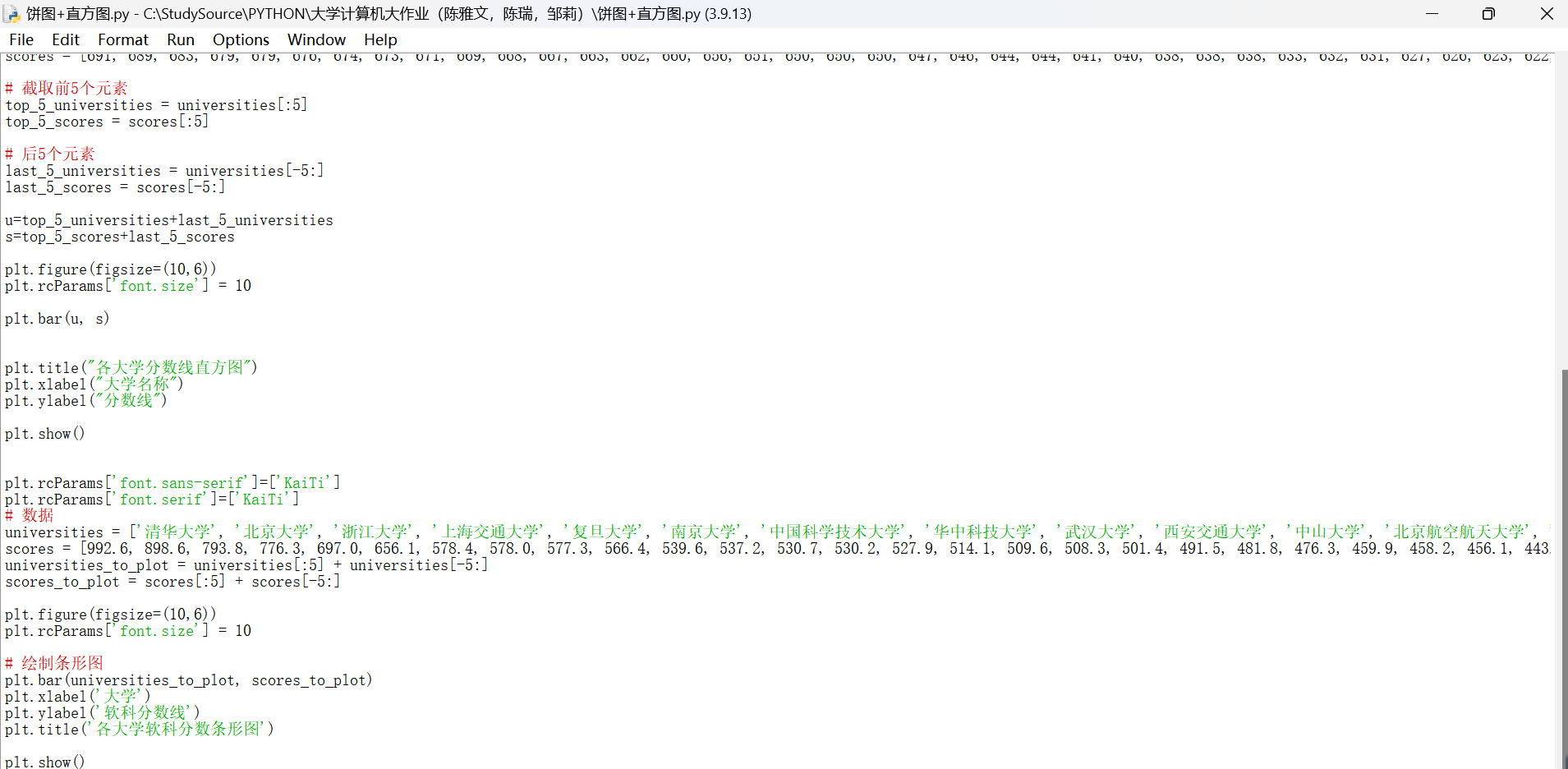


图12 直方图代码截图

1. 为线性拟合准备数据：运用python内置的数据类型set集合[2]，整合分数线、软科、校友会三个文件。文件如图13所示，完整代码如图14所示。

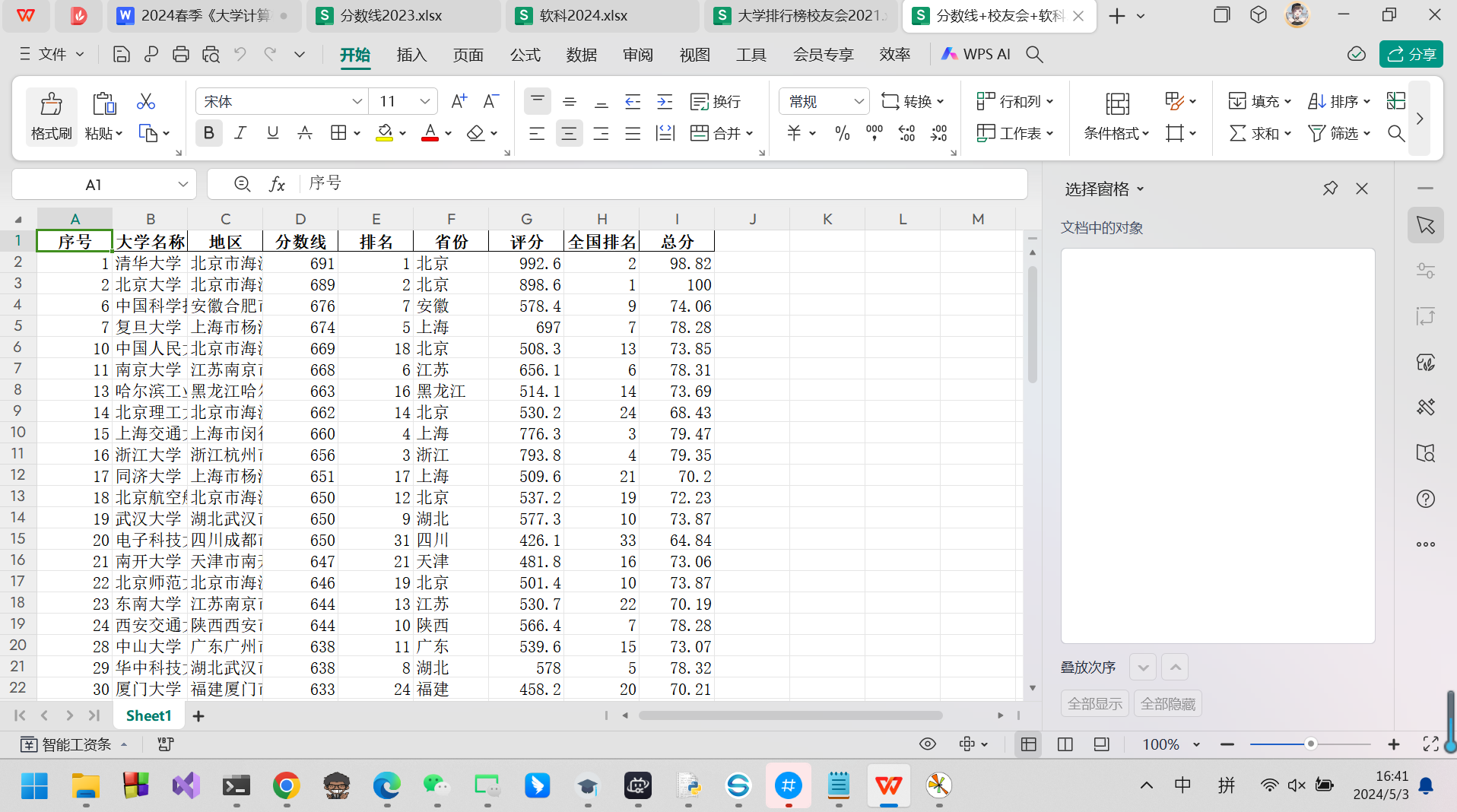


图13 合并文件截图

三个文件中只有75所学校的数据是重合的，因此合并后数据量受到了一定影响

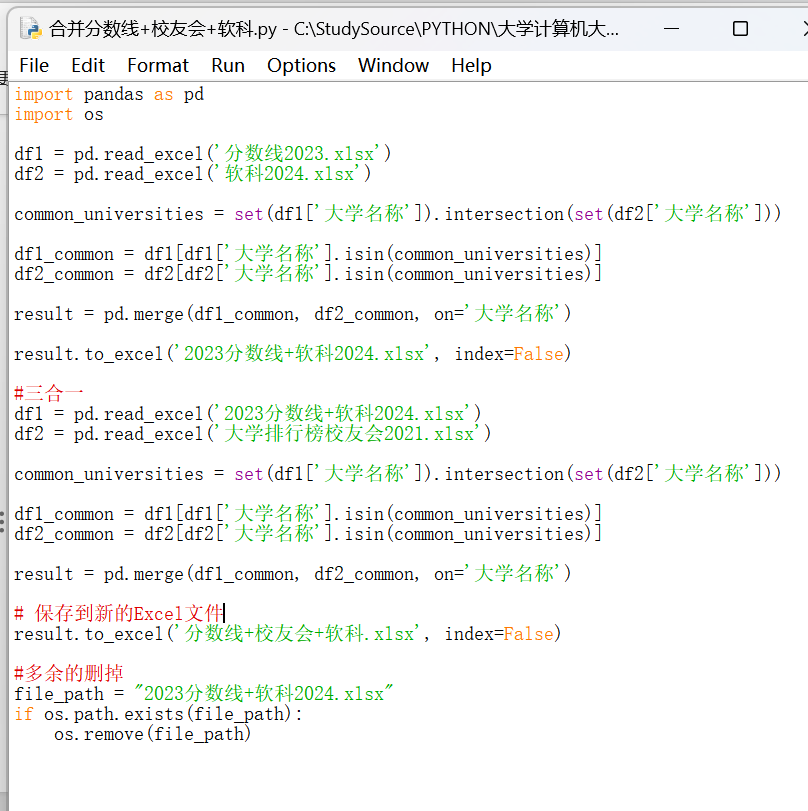


图14 合并文件代码截图

1. 线性拟合曲线绘制，学校性价比分析：

利用Python的数据分析库scipy，我进行了线性回归分析来拟合校友会总分和分数线、软科评分和分数线之间的关系。

使用matplotlib和seaborn绘制了线性拟合曲线，并通过曲线直观地展示了变量之间的线性关系。如图15所示

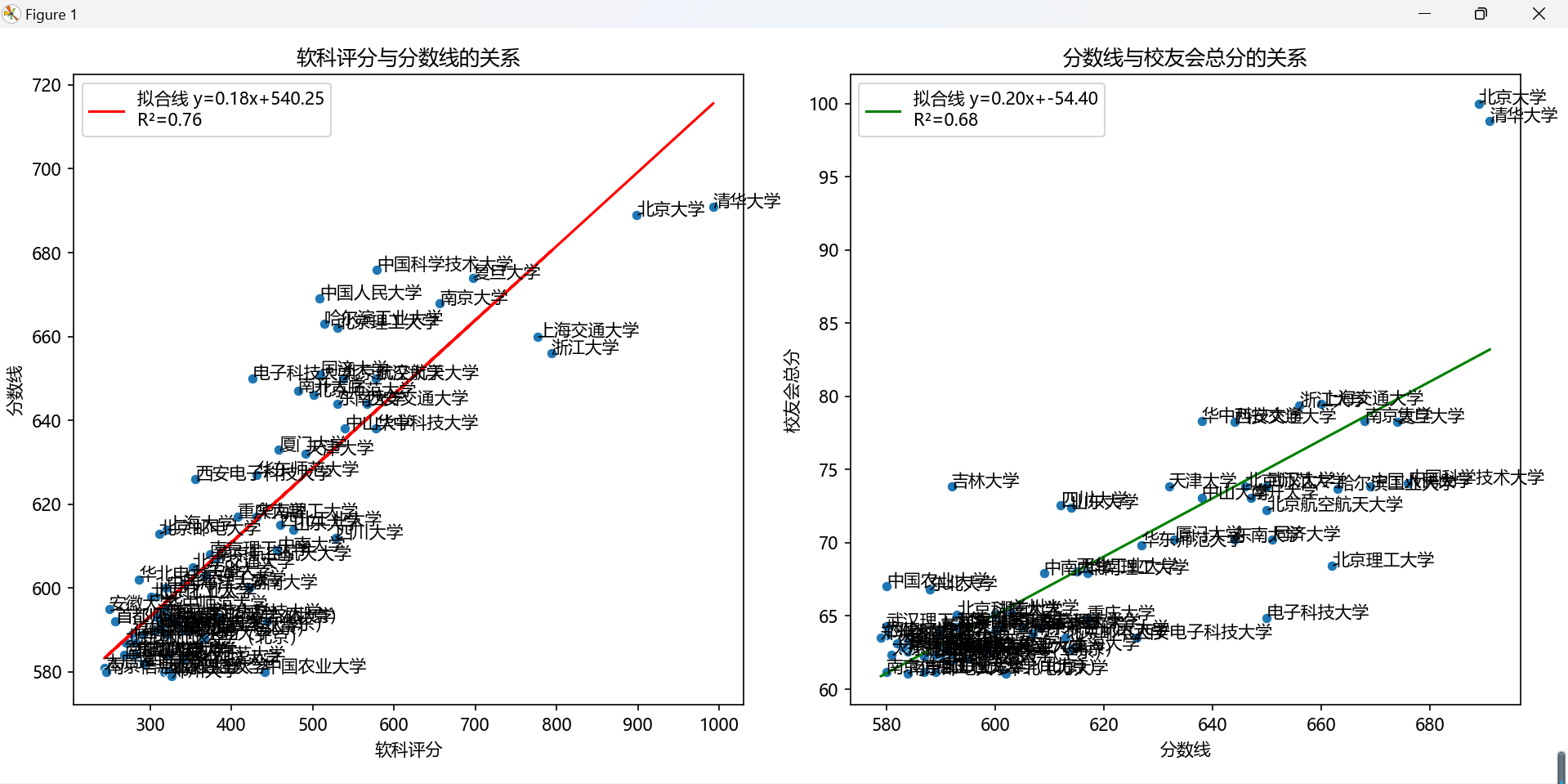
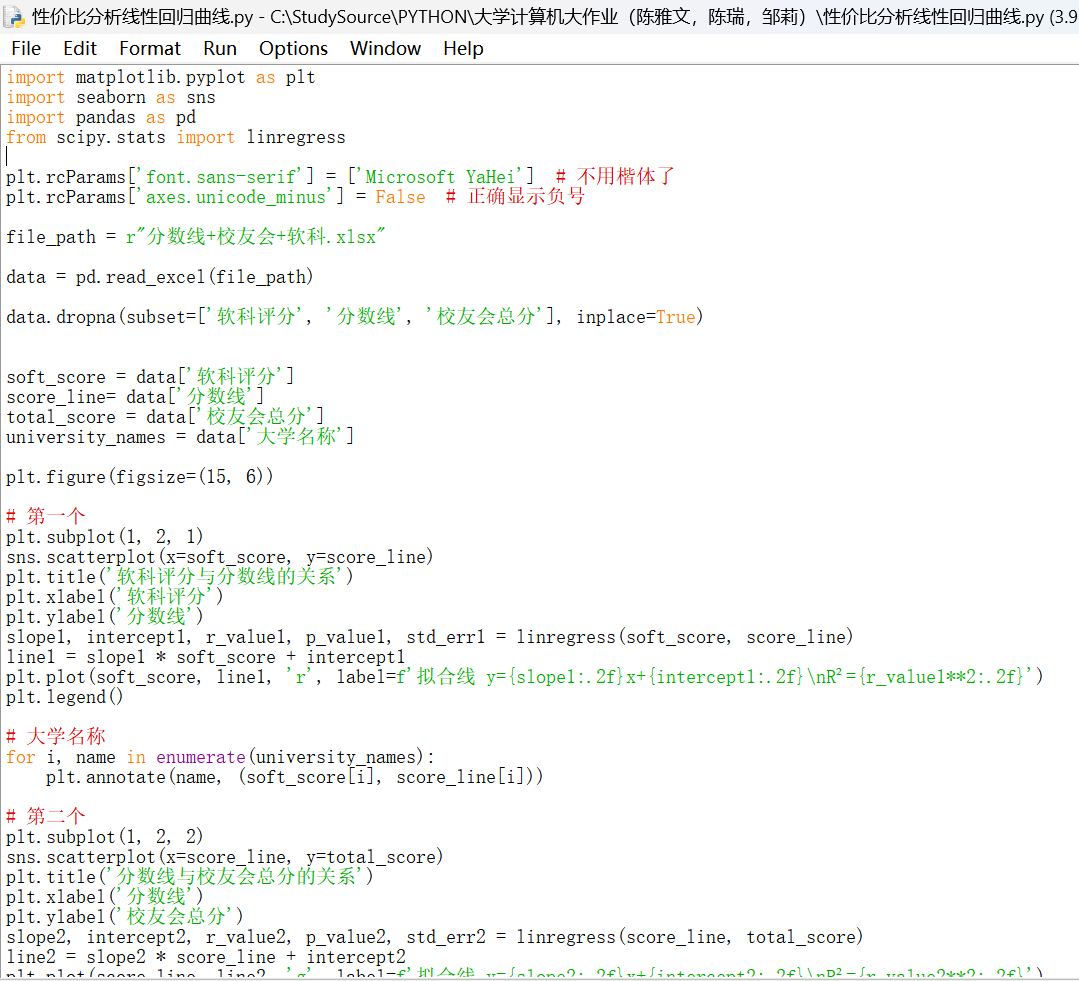


图15 拟合曲线截图

对于一所学校而言，评分意味着它的综合实力，分数线则意味着它的录取难度。根据图15 可以看出，分数线和评分基本呈正相关，但相关度并没有很高，这就说明，每所学校的综合实力和它的录取难度并不完全匹配。一所学校的综合实力越强，录取难度越低，就意味着该校的性价比越高，反之，相反。

对于分数线——软科曲线，由于它将软科评分作为x轴，因此落在曲线右下方的学校性价比较高，离曲线越远性价比越高，左上方的学校性价比较低，离曲线越远性价比越低。校友会——分数线曲线则相反，曲线左上方代表着高性价比。

1. 性价比拟合完整代码截图如图16所示



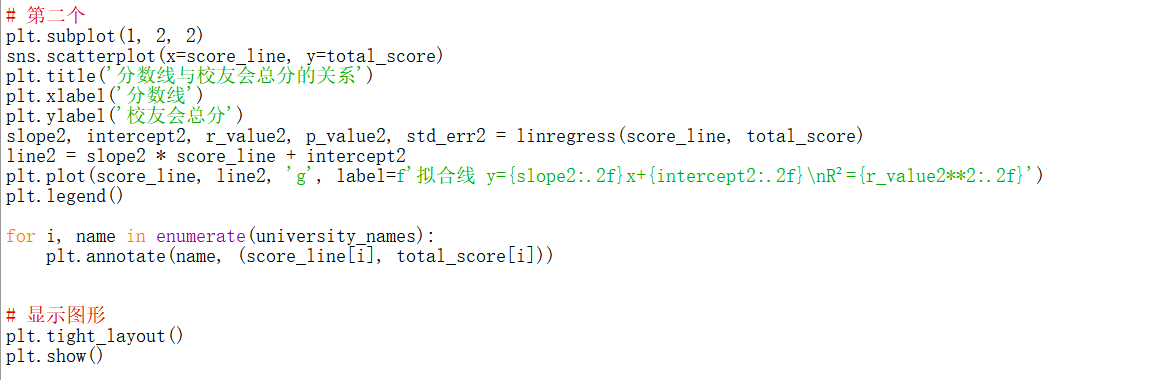


图16 拟合曲线代码截图

1. 性价比指数计算：

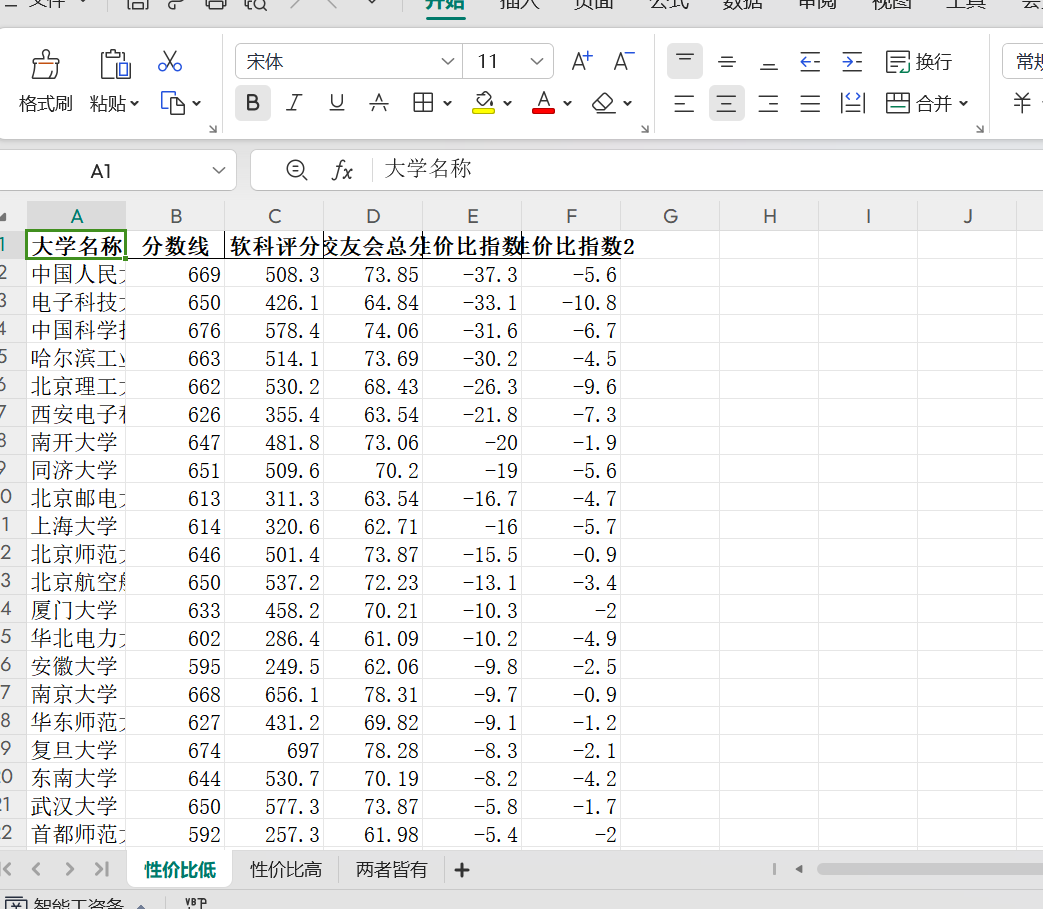
线性拟合曲线给出了校友会总分和分数线，软科评分和分数线等的函数表达式，我们将它作为性价比指数的计算公式，为每个高校计算了性价比指数。

在软科评分下的性价比公式为：性价比指数1=0.18 \* 软科分数 + 540.25 - 分数线

在校友会总分下的性价比公式为：性价比指数2=校友会评分 - 0.2 \* 分数线 + 54.4

得到性价比指数后，我将两项性价比指数都为正数的学校存入excel的“性价比高”这一页，都为负数的存为“性价比低”这一页，一正一负的存入“两者皆有”这一页。两种评分中，软科评分更为权威，因此我以性价比指数1为主要分析点

性价比表格中包含了每所高校的名称、两种标准下的评分、分数线以及计算得到的两项性价比指数，如图17，图18，图19所示



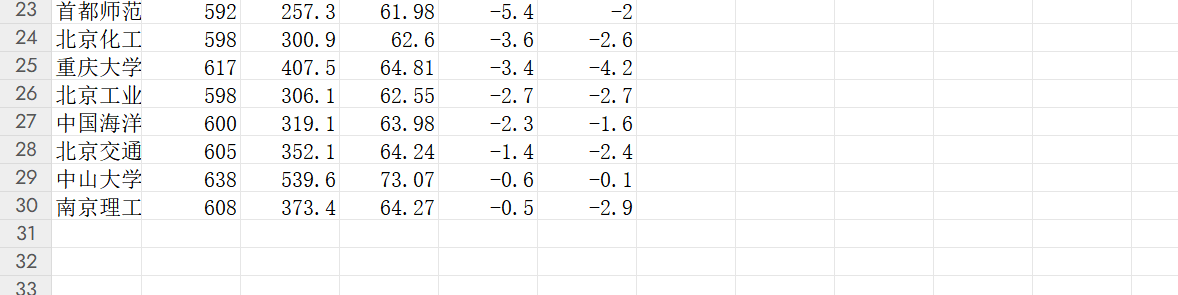


图17 低性价比学校截图

总共有29所学校被归到了性价比低的部分，这一部分我以性价比指数1为标准升序排序，可以看到，性价比指数1最低的是中国人民大学，但这所学校的性价比指数2不算很低。原因可能如下。

软科排名主要关注学校的学术水平，包括科研成果，科研人才等方面，人大偏文科，可能在这些方面得分较低，因此软科评分低。

而人大的校友资源丰富、历史悠久，因此校友会评分不低。

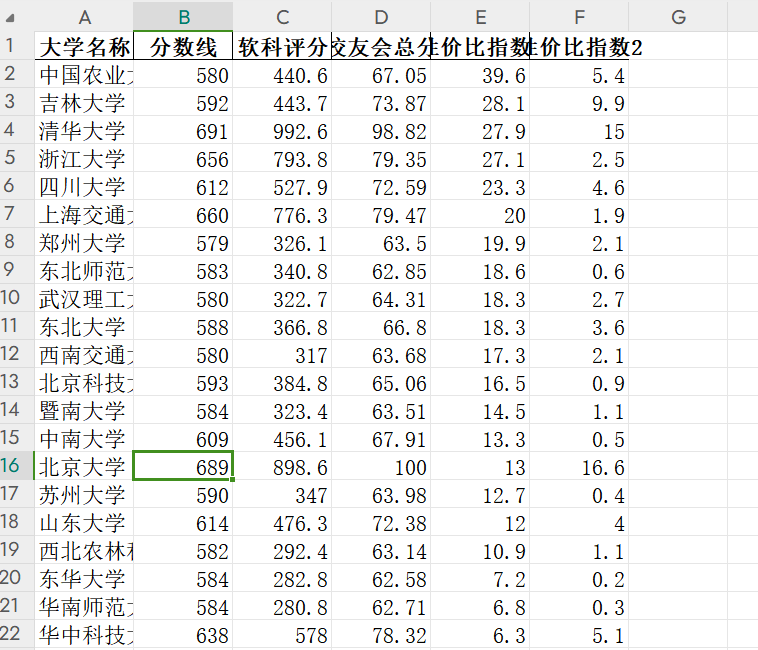




图18 高性价比学校截图

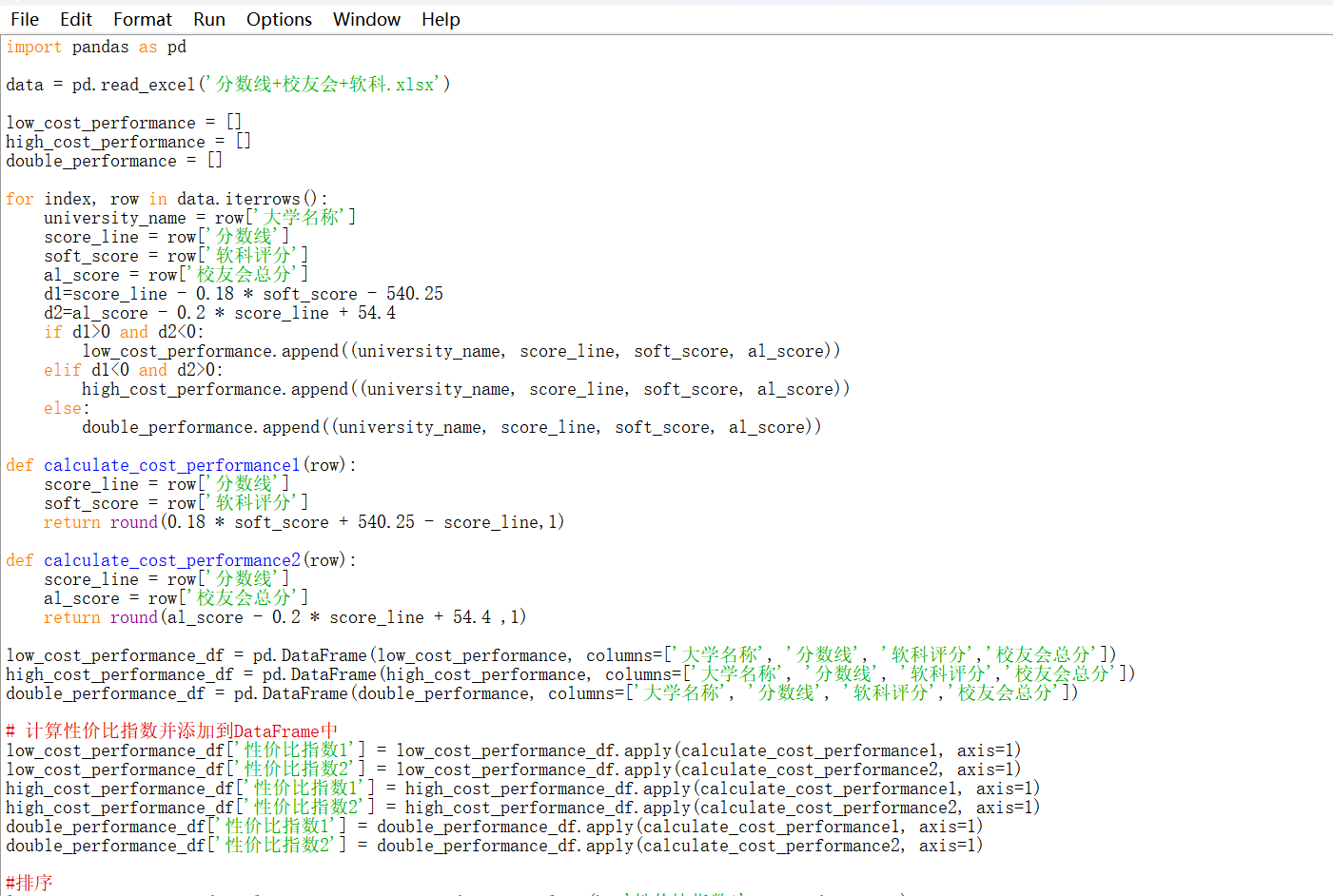
总共有25所学校被归到了性价比高的部分，这一部分我以性价比指数1为标准降序排序。可以看到，性价比指数1最高的是中国农业大学，这可能是因为它在农业、林业及相关学科领域的强大实力让它软科评价中得到优势，从而获得较高的软科评分，还有一个原因是考生对农业类专业的就业前景不够看好，导致不愿意报考农业类专业，从而导致分数线较低，综合原因导致了中国农大的性价比指数1为所有学校中最高。



图19 两者皆有学校截图

总共有22所学校被归到了两者皆有的部分，这一部分我以性价比指数1为标准降序排序。可以看到，在这一部分性价比指数1最高的是湖南大学，但它的性价比指数2却为负数，这可能和湖南大学偏理工的原因有关。

1. 性价比分析完整代码如图20所示



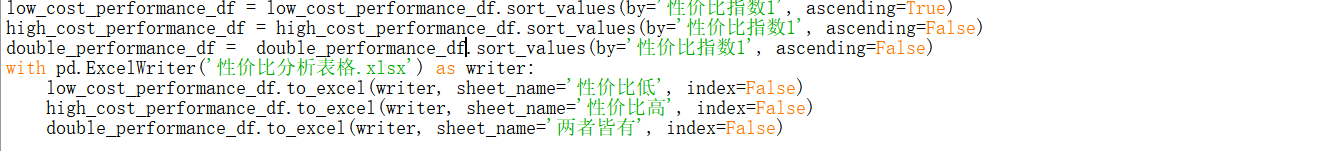


图20 性价比分析代码截图

1. **数据管理展示**

在数据管理展示的过程中，我使用了Python作为工具，导入sqlite3模块，使用sqlite3.connect()方法连接到数据库，读取Excel文件并存入数据库。

总共建立了两个数据库，数据库1关于校友会排名、软科排名、分数线这三个文件，数据库2关于上述文件合并后的文件

现在说明一下我的数据库的增删改查功能。

1. 增加功能，效果如图21所示。

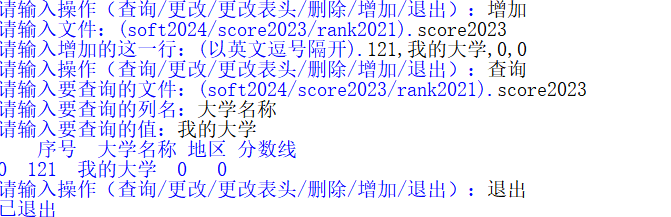


图21 增加部分截图

如图所示，我们在表中增加了“我的大学”这一行内容，并查询得到验证

1. 更改功能，分为普通的“更改”和更改表头。

对于数据库1，我们要更改校友会排名的表头，以便后续合并文件，如图22所示

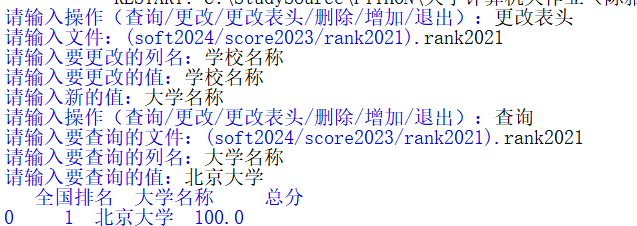


图22 更改部分截图1

当我们合并表格存入数据库2时，我们会发现，如果表头只有“评分”和“总分”，我们难以判断它来自哪个文件，因此我们要在这之前加上“软科”和“校友会”的前缀



图23 更改部分截图2

如果我们觉得某些大学的名称太复杂，不利于后续可视化展示，也可以将其简写，如图24所示

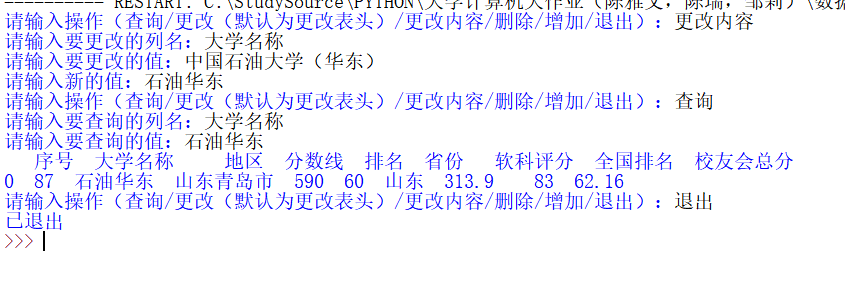


图24 更改部分截图3

1. 删除功能，效果如图25所示

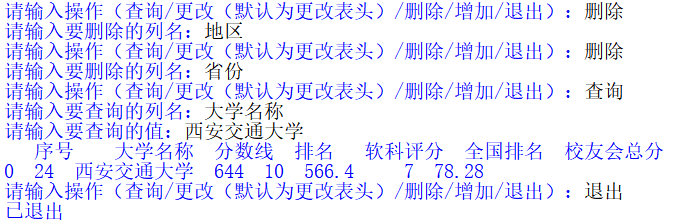


图25 删除部分截图

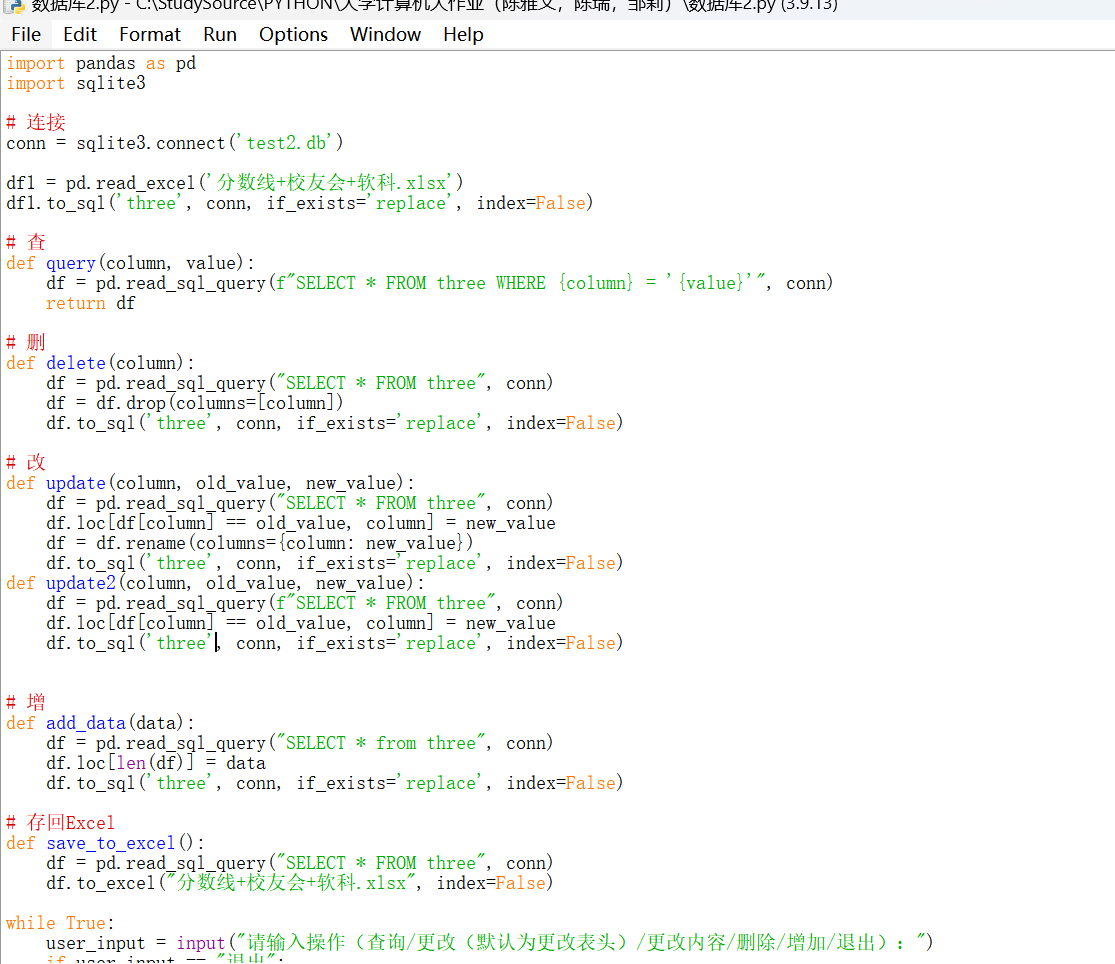
如图，合并后我们只需要大学名称，分数线和评分，地区和省份显得比较多余，因此我们就将这两列删除。

1. 查找功能。查找功能在以上三种功能的演示中已经体现，就不单独演示了
2. 数据库1的代码如图26所示，数据库2的代码如图27所示。





图26 数据库1代码截图



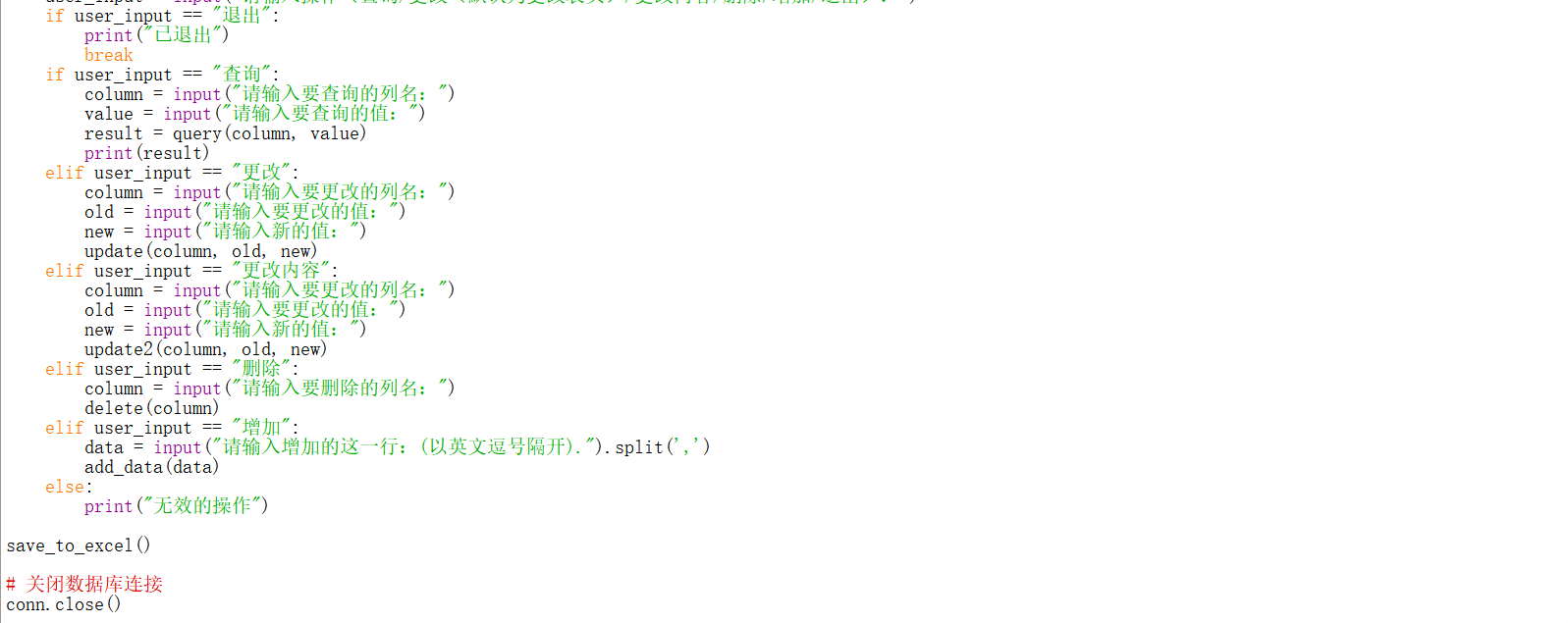


图27 数据库2代码截图

**五、遇到的问题与解决方法**

1. 遇到的问题

陈雅文：

1. Pycharm安装后无法使用。
2. 爬虫后打印的格式不够整齐。
3. 本来是只打算爬取分数线和软科排名的，但发现它们是动态网站，翻页后网站名不变，只能爬到一页的信息。
4. 文件提取时，由于复制过来的内容并不完全按照规律分布，有很多乱码，常常出错
5. 绘制饼图时，发现无法显示中文。
6. 数据库处理，查的部分发现输出的是元组，略略不美观。
7. 数据库处理，数据被改动后不方便存回原本文件，如果将更改后的文件另存为，文件就太多了显得乱。
8. 数据库处理，发现普通的更改无法更改表头。
9. 用来绘制回归曲线的scipy库安装不了
10. 软科——分数线曲线绘制时，如果以分数线作为x轴，数据就会太拥挤，查看不方便

邹莉：

1. 安装第三方库时一直红码，下载不下来。
2. 文件路径运行时，用文件名称读取不了。

陈瑞：

1. 由于网络或配置问题，始终无法顺利下载并安装matplotlib。
2. 在编写代码的过程中，遇到了一些小错误。
3. 在使用PyCharm这个集成开发环境（IDE）方面，意识到自己还不够熟练
4. 解决办法

陈雅文：

1. 选择下载第三方库后用idle，后来知道是自己的Pycharm误下载了专业版。
2. 反复尝试调整格式化输出，直到有一种整齐的输出方式
3. 先去找其他网站看能不能代替，排名评分方面找到了高考网的校友会排名，但校友会排名没有大学的来源省份，而且分数线的网站无论如何都没有找到替代品，替代法不可行。我也曾犹豫要不要只提取一页的数据，最终还是决定不要这么敷衍。最后我使用了一个笨办法，复制内容到文本再编写代码提取有用信息到excel。同时，我把校友会也列入了我们数据分析中，增添了一个新的分析维度。
4. 先去word里通过查找替换删掉一些出现频率高的乱码，再分情况处理提取
5. 发现是开头没有设置中文字体，加上了plt.rcParams语句。
6. 利用pd.read\_sql\_query()函数，将结果存储在DataFrame对象中，输出时，DataFrame的内容会被格式化为一个易于阅读的形式。
7. 完善增删改部分的代码，使这些改动可逆，这样就能放心存回原本文件。
8. 在程序中新增了“更改表头”这一选项并对此编写代码。
9. 自己从https://pypi.org/下载.whl文件，然后安装。[4]
10. 以分数线作为y轴，软科评分作为x轴。虽然这么做和我想表达的因果关系不太符合

邹莉：

1. 发现是网络原因，更换流畅网络下载成功。
2. 创建文件夹保存在d盘读取文件位置运行。

陈瑞：

1. 选择了使用Jupyter Notebook作为替代方案。
2. 仔细地检查了每一行代码，寻找潜在的逻辑和语法错误。
3. 投入时间学习了PyCharm的各种功能，如代码自动完成、调试工具以及项目管理等。

**六、小组工作分工**

查找网站和资料、爬虫获取数据存入excel、饼图制作：陈雅文

直方图制作：陈雅文、陈瑞

合并excel、数据库建立、制作性价比指数表格：陈雅文

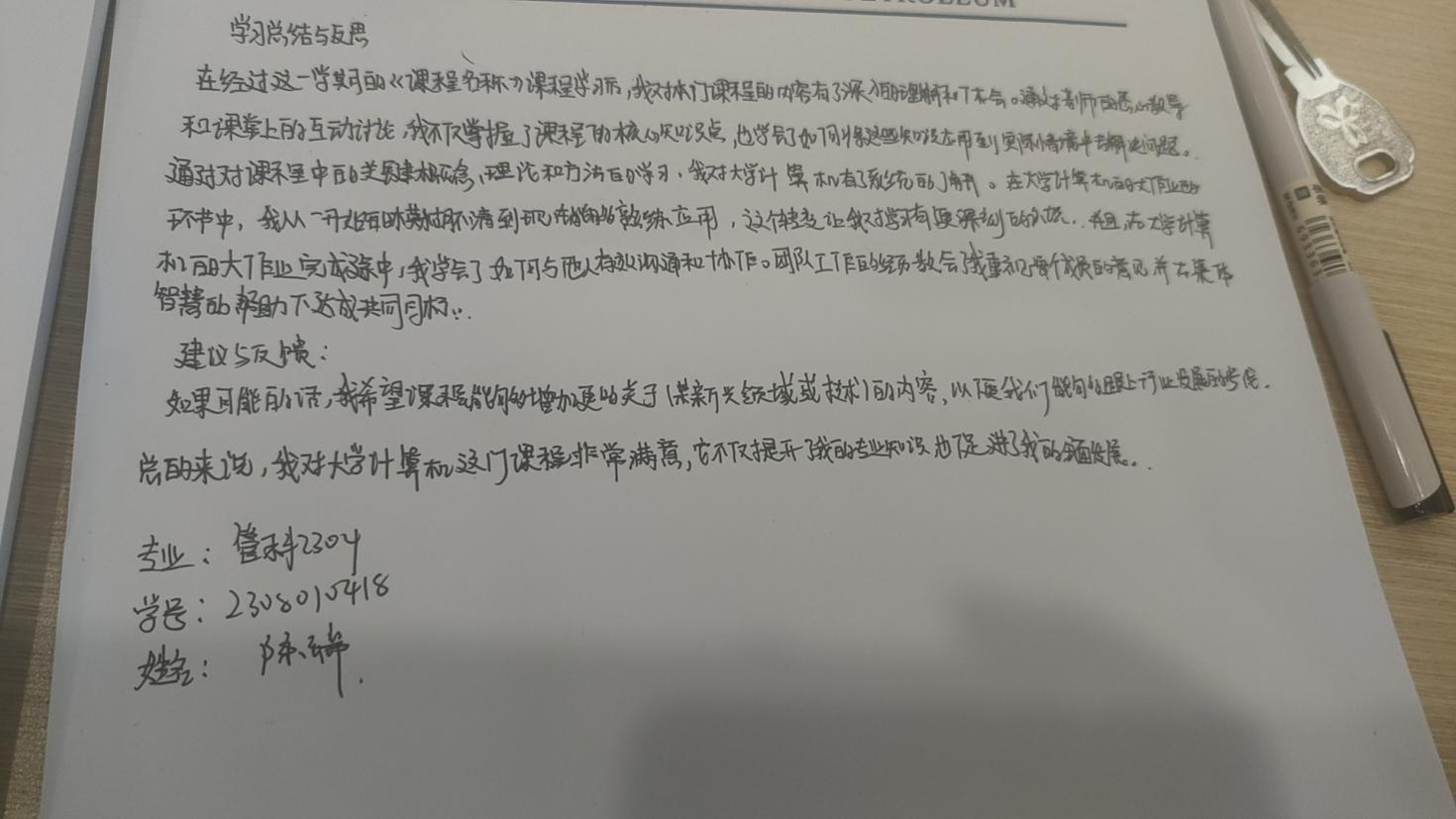
制作性价比分析回归曲线：邹莉、陈雅文

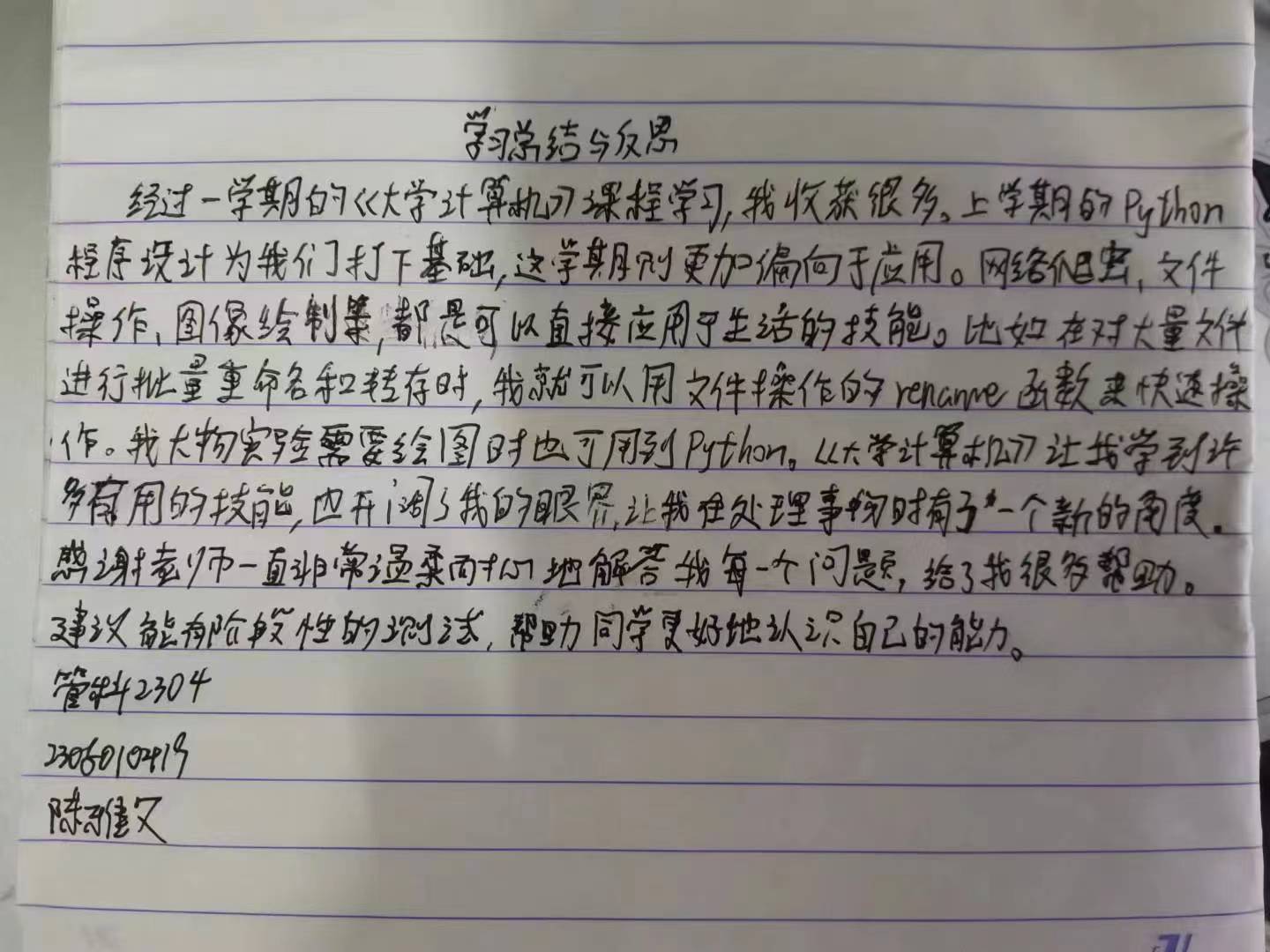
报告撰写：陈雅文

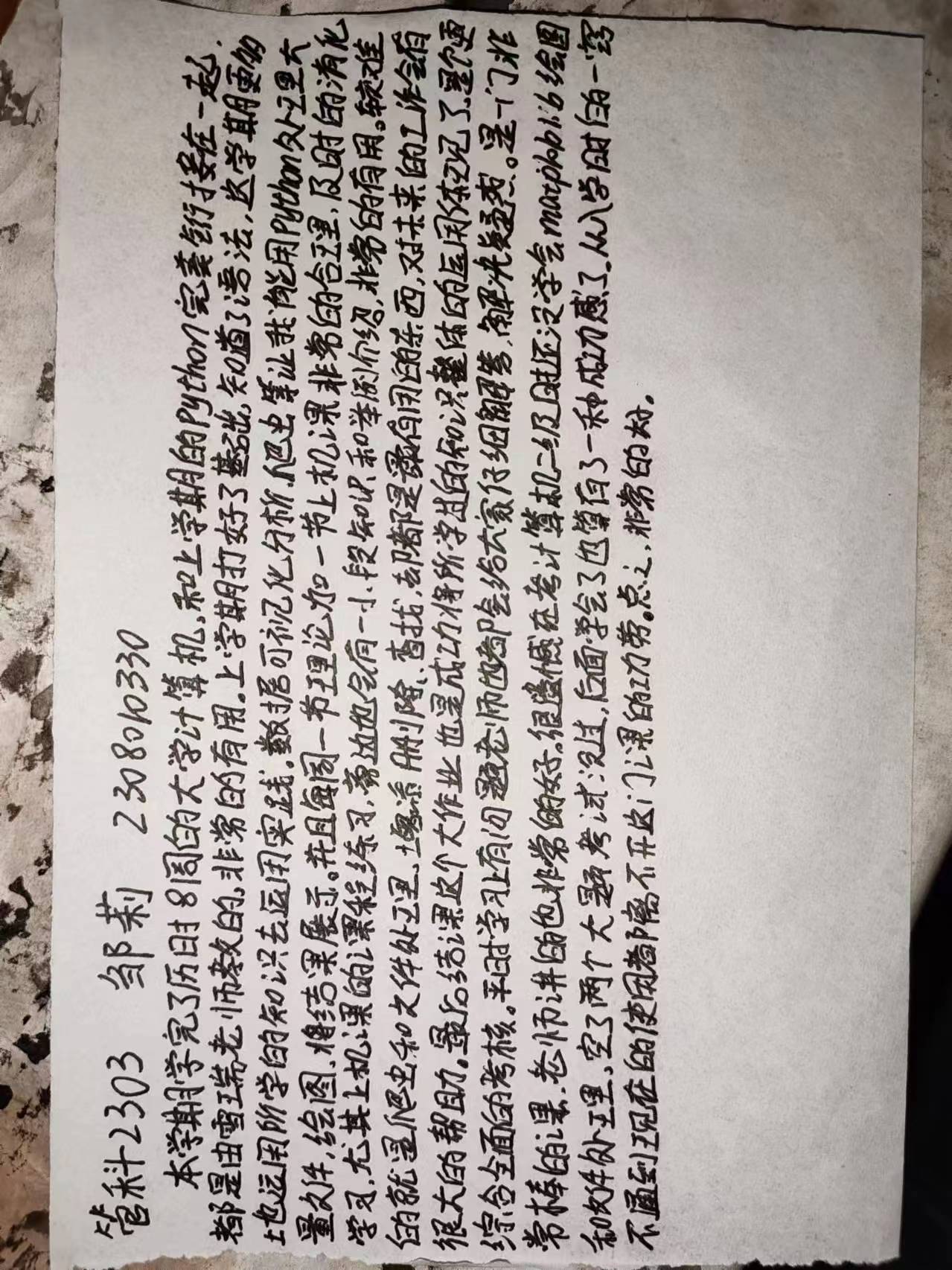
提供意见：邹莉、陈瑞

**七、学习总结与反思**

正文部分：本部分主要写你学习本门课程的心得和总结，也欢迎写下你对课程及老师的宝贵建议。可多可少，避免空谈，感情要真挚。**（每位同学都要在纸上手写，写上专业、学号、姓名，拍照截图放在下面）**

****





**八、附件**

附件1、完整代码（此处粘贴代码，不要截图）

#1

# -\*- coding:utf-8 -\*-

# -\*- coding:gb2312-\*-

import requests

import urllib.request

from bs4 import BeautifulSoup

import bs4

import pandas as pd

text = "一，爬取大学排名校友会+分数线+软科"

print(f"{text:^{40}}")

url = 'http://www.gaokao.com/e/20210328/606032dc1b634.shtml'

data=[]

def getHTMLText(url):

try:

r = requests.get(url, timeout=30)

r.raise\_for\_status()

r.encoding = r.apparent\_encoding

return r.text

except:

return ""

def fillUnivList(ulist, html):

soup = BeautifulSoup(html, "html.parser")

for tr in soup.find('tbody').children:

if isinstance(tr, bs4.element.Tag):

tds = tr('td')

ulist.append([tds[0].text.strip(), tds[1].text.strip(), tds[2].text.strip()])

def printUnivList(ulist, num):

tplt = "{0:^10}\t.{1:{3}^10}\t.{2:^10}"

for i in range(num):

u = ulist[i]

print(tplt.format(u[0],u[1],u[2],chr(12288)))

data.append([u[0],u[1],u[2]])

def main():

uinfo = []

html = getHTMLText(url)

fillUnivList(uinfo,html)

printUnivList(uinfo,140)

if \_\_name\_\_ == "\_\_main\_\_":

main()

filename = '大学排行榜校友会2021.xlsx'

df = pd.DataFrame(data[1:], columns=data[0])

df.to\_excel(filename, index=False)

with open('分数线2023.txt', 'r', encoding='utf-8') as file:

cnt=1

n=1

a=[]

a2=[]

for line in file:

if line.strip():

if cnt==1:

a.append(n)

a.append(line.strip())

cnt+=1

elif cnt==2:

a.append(line.strip())

cnt+=1

elif cnt==3:

b=line.split('：')

#print(b)

if len(b)==2:

a.extend([b[1][0:3]])

a2.append(a)

a=[]

cnt=1

n+=1

df = pd.DataFrame(a2,columns=['序号', '大学名称', '地区', '分数线'])

df.to\_excel('分数线2023.xlsx', index=False)

with open('软科2024.txt', 'r', encoding='utf-8') as file:

cnt=1

a=[]

a2=[]

for line in file:

if line.strip():

if cnt<5:

if cnt==1:

a.append(line.strip())

elif cnt==2:

a.append(line.strip())

cnt+=1

elif cnt==5:

b=line.split()

if len(b)==4 or len(b)==3:

a.extend([b[0],b[2]])

a2.append(a)

a=[]

cnt=1

else:

cnt+=1

elif cnt==6:

b=line.split()

a.extend([b[0],b[2]])

a2.append(a)

a=[]

cnt=1

df = pd.DataFrame(a2,columns=['排名', '大学名称', '省份', '评分'])

df.to\_excel('软科2024.xlsx', index=False)

#2

text = "二，绘制饼图+直方图"

print(f"{text:^{40}}")

import pandas as pd

import matplotlib.pyplot as plt

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

plt.rcParams['font.serif']=['KaiTi']

# 软科2024

df = pd.read\_excel('软科2024.xlsx')

province\_counts = df['省份'].value\_counts()

explode = [0.1 if province == '山东' else 0 for province in province\_counts.index]

plt.figure(figsize=(10,6))

plt.pie(province\_counts, labels=province\_counts.index, autopct='%1.1f%%', explode=explode)

plt.title('大学评分前120省份分布')

# 分数线2023

df = pd.read\_excel('分数线2023.xlsx')

province\_counts = df['地区'].value\_counts()

explode = [0.1 if province[0:2] == '山东' else 0 for province in province\_counts.index]

plt.figure(figsize=(10,6))

plt.pie(province\_counts, labels=province\_counts.index, autopct='%1.1f%%', explode=explode)

plt.title('大学分数线前120地区分布')

plt.show()

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

plt.rcParams['font.serif']=['KaiTi']

universities = ['清华大学', '北京大学', '中国科学院大学', '北京大学医学部', '上海交通大学医学院', '中国科学技术大学', '复旦大学', '浙江大学医学院', '复旦大学上海医学院', '中国人民大学', '南京大学', '哈尔滨工业大学（深圳）', '哈尔滨工业大学', '北京理工大学', '上海交通大学', '浙江大学', '同济大学', '北京航空航天大学', '武汉大学', '电子科技大学', '南开大学', '北京师范大学', '东南大学', '西安交通大学', '中国人民解放军国防科技大学', '上海中医药大学', '对外经济贸易大学', '中山大学', '华中科技大学', '厦门大学', '天津大学', '中国人民公安大学', '华东师范大学', '西安电子科技大学', '上海财经大学', '中国政法大学', '中央财经大学', '中国社会科学院大学', '华南理工大学', '重庆大学', '北京电影学院', '西北工业大学', '山东大学', '上海大学', '哈尔滨工业大学（威海）', '北京邮电大学', '上海外国语大学', '四川大学', '华东政法大学', '中南大学', '南京理工大学', '西南财经大学', '南京航空航天大学', '山东大学（威海）', '北京交通大学', '中国传媒大学', '上海对外经贸大学', '中央民族大学', '中国药科大学', '兰州大学', '海军军医大学', '华北电力大学', '华东理工大学', '深圳大学', '中国海洋大学', '中国医科大学', '湖南大学', '北京协和医学院', '北京工业大学', '贵州医科大学', '北京化工大学', '中央美术学院', '安徽大学', '西南政法大学', '华中师范大学', '东北大学秦皇岛分校', '北京科技大学', '南京师范大学', '河海大学', '首都师范大学', '吉林大学', '中国石油大学（北京）', '陕西师范大学', '西北大学', '苏州大学', '中国石油大学（华东）', '哈尔滨工程大学', '大连理工大学盘锦校区', '西北政法大学', '江南大学', '福州大学', '中国地质大学（北京）', '东北大学', '北京外国语大学', '北京林业大学', '中国矿业大学（北京）', '安徽医科大学', '南京审计大学', '中国人民解放军空军军医大学', '暨南大学', '杭州电子科技大学', '东华大学', '华南师范大学', '合肥工业大学', '南京邮电大学', '上海海关学院', '东北师范大学', '西北农林科技大学', '合肥工业大学宣城校区', '华北电力大学保定校区', '太原理工大学', '北京中医药大学', '西南交通大学', '上海电力大学', '中国农业大学', '武汉理工大学', '南京信息工程大学', '郑州大学', '中国美术学院']

scores = [691, 689, 683, 679, 679, 676, 674, 673, 671, 669, 668, 667, 663, 662, 660, 656, 651, 650, 650, 650, 647, 646, 644, 644, 641, 640, 638, 638, 638, 633, 632, 631, 627, 626, 623, 622, 622, 618, 617, 617, 616, 615, 614, 614, 613, 613, 612, 612, 612, 610, 609, 608, 607, 607, 605, 605, 605, 605, 604, 604, 603, 603, 602, 602, 601, 601, 600, 600, 600, 599, 598, 598, 598, 598, 595, 595, 595, 593, 593, 593, 593, 592, 592, 592, 592, 592, 591, 590, 590, 590, 590, 590, 589, 589, 589, 588, 588, 587, 587, 587, 585, 585, 585, 584, 584, 584, 584, 583, 583, 582, 582, 581, 581, 581, 580, 580, 580, 580, 579]

# 截取前5个元素

top\_5\_universities = universities[:5]

top\_5\_scores = scores[:5]

# 后5个元素

last\_5\_universities = universities[-5:]

last\_5\_scores = scores[-5:]

u=top\_5\_universities+last\_5\_universities

s=top\_5\_scores+last\_5\_scores

plt.figure(figsize=(10,6))

plt.rcParams['font.size'] = 10

plt.bar(u, s)

plt.title("各大学分数线直方图")

plt.xlabel("大学名称")

plt.ylabel("分数线")

plt.show()

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

plt.rcParams['font.serif']=['KaiTi']

# 数据

universities = ['清华大学', '北京大学', '浙江大学', '上海交通大学', '复旦大学', '南京大学', '中国科学技术大学', '华中科技大学', '武汉大学', '西安交通大学', '中山大学', '北京航空航天大学', '东南大学', '北京理工大学', '四川大学', '哈尔滨工业大学', '同济大学', '中国人民大学', '北京师范大学', '天津大学', '南开大学', '山东大学', '西北工业大学', '厦门大学', '中南大学', '吉林大学', '中国农业大学', '大连理工大学', '华东师范大学', '华南理工大学', '电子科技大学', '湖南大学', '重庆大学', '南方科技大学', '北京科技大学', '南京航空航天大学', '南京理工大学', '兰州大学', '东北大学', '西安电子科技大学', '北京交通大学', '上海科技大学', '苏州大学', '华东理工大学', '哈尔滨工程大学', '东北师范大学', '华中农业大学', '中国石油大学（北京）', '南京农业大学', '郑州大学', '暨南大学', '南京师范大学', '武汉理工大学', '中国矿业大学', '江南大学', '上海大学', '中国海洋大学', '华中师范大学', '西南交通大学', '中国石油大学（华东）', '中国地质大学（武汉）', '陕西师范大学', '北京邮电大学', '浙江工业大学', '河海大学', '北京工业大学', '西北大学', '北京化工大学', '西南大学', '云南大学', '南昌大学', '深圳大学', '西北农林科技大学', '中国地质大学（北京）', '华北电力大学', '东华大学', '中国矿业大学（北京）', '宁波大学', '华南师范大学', '江苏大学', '福州大学', '扬州大学', '合肥工业大学', '北京林业大学', '南京邮电大学', '湖南师范大学', '福建师范大学', '浙江师范大学', '南京工业大学', '广西大学', '首都师范大学', '长安大学', '南京林业大学', '华南农业大学', '安徽大学', '贵州大学', '杭州电子科技大学', '广州大学', '河南大学', '广东工业大学', '山西大学', '湘潭大学', '南京信息工程大学', '海南大学', '山东师范大学', '浙江理工大学', '上海理工大学', '太原理工大学', '河北工业大学', '青岛大学', '大连海事大学', '燕山大学', '东北林业大学', '西安建筑科技大学', '昆明理工大学', '内蒙古大学', '江西师范大学', '西安理工大学', '上海师范大学']

scores = [992.6, 898.6, 793.8, 776.3, 697.0, 656.1, 578.4, 578.0, 577.3, 566.4, 539.6, 537.2, 530.7, 530.2, 527.9, 514.1, 509.6, 508.3, 501.4, 491.5, 481.8, 476.3, 459.9, 458.2, 456.1, 443.7, 440.6, 435.7, 431.2, 430.6, 426.1, 422.2, 407.5, 396.4, 384.8, 380.1, 373.4, 367.6, 366.8, 355.4, 352.1, 348.5, 347.0, 341.5, 340.9, 340.8, 338.1, 331.4, 330.8, 326.1, 323.4, 322.7, 322.7, 321.7, 320.6, 319.1, 318.5, 317.0, 313.9, 313.2, 312.6, 311.3, 309.7, 307.7, 306.1, 305.2, 300.9, 300.6, 297.8, 297.0, 296.4, 292.4, 291.4, 286.4, 282.8, 282.3, 281.3, 277.5, 271.8, 270.2, 269.6, 263.7, 263.1, 262.4, 258.9, 257.6, 257.3, 253.0, 251.9, 249.5, 249.5, 248.2, 247.7, 247.6, 247.2, 244.8, 243.5, 238.5, 237.5, 237.3, 236.9, 232.5, 231.7, 230.8, 230.7, 230.0]

universities\_to\_plot = universities[:5] + universities[-5:]

scores\_to\_plot = scores[:5] + scores[-5:]

plt.figure(figsize=(10,6))

plt.rcParams['font.size'] = 10

# 绘制

plt.bar(universities\_to\_plot, scores\_to\_plot)

plt.xlabel('大学')

plt.ylabel('软科分数线')

plt.title('各大学软科分数条形图')

plt.show()

#3

text = "三，将以上数据存入数据库"

print(f"{text:^{40}}")

import pandas as pd

import sqlite3

# 创建数据库连接

conn = sqlite3.connect('test.db')

# 读取Excel文件并存入数据库

df1 = pd.read\_excel('软科2024.xlsx')

df1.to\_sql('soft2024', conn, if\_exists='replace', index=False)

df2 = pd.read\_excel('分数线2023.xlsx')

df2.to\_sql('score2023', conn, if\_exists='replace', index=False)

df3 = pd.read\_excel('大学排行榜校友会2021.xlsx')

df3.to\_sql('rank2021', conn, if\_exists='replace', index=False)

# 查

def query(file, column, value):

df = pd.read\_sql\_query(f"SELECT \* FROM {file} WHERE {column} = '{value}'", conn)

return df

# 删

def delete(file, column):

df = pd.read\_sql\_query(f"SELECT \* FROM {file}", conn)

df = df.drop(columns=[column])

df.to\_sql(file, conn, if\_exists='replace', index=False)

# 改

def update(file, column, old\_value, new\_value):

df = pd.read\_sql\_query(f"SELECT \* FROM {file}", conn)

df.loc[df[column] == old\_value, column] = new\_value

df.to\_sql(file, conn, if\_exists='replace', index=False)

def updatehead(file, column, old\_value, new\_value):

df = pd.read\_sql\_query(f"SELECT \* FROM {file}", conn)

df.loc[df[column] == old\_value, column] = new\_value

df = df.rename(columns={column: new\_value})

df.to\_sql(file, conn, if\_exists='replace', index=False)

# 增

def add\_data(file\_name, data):

table\_name = file\_name.split('.')[0]

df = pd.read\_sql\_query("SELECT \* from {}".format(table\_name), conn)

df.loc[len(df)] = data

df.to\_sql(table\_name, conn, if\_exists='replace', index=False)

# 将数据库内容存回Excel

def save\_to\_excel(file):

df = pd.read\_sql\_query(f"SELECT \* FROM {file}", conn)

df.to\_excel(f"{file}.xlsx", index=False)

print("小提醒：必须更改rank2021的表头‘学校名称’为‘大学名称’，不然影响后续合并")

while True:

user\_input = input("请输入操作（查询/更改/更改表头/删除/增加/退出）：")

if user\_input == "退出":

print("已退出")

break

if user\_input == "查询":

file = input("请输入要查询的文件：(soft2024/score2023/rank2021).")

column = input("请输入要查询的列名：")

value = input("请输入要查询的值：")

result = query(file,column, value)

print(result)

elif user\_input == "更改":

file = input("请输入文件：(soft2024/score2023/rank2021).")

column = input("请输入要更改的列名：")

row\_id = input("请输入要更改的值：")

new\_value = input("请输入新的值：")

update(file,column, row\_id, new\_value)

elif user\_input == "更改表头":

file = input("请输入文件：(soft2024/score2023/rank2021).")

column = input("请输入要更改的列名：")

row\_id = input("请输入要更改的值：")

new\_value = input("请输入新的值：")

updatehead(file,column, row\_id, new\_value)

elif user\_input == "删除":

file = input("请输入文件：(soft2024/score2023/rank2021).")

column = input("请输入要删除的列名：")

delete(file,column)

elif user\_input == "增加":

file = input("请输入文件：(soft2024/score2023/rank2021).")

data = input("请输入增加的这一行：(以英文逗号隔开).").split(',')

add\_data(file, data)

else:

print("无效的操作")

# 从数据库中读取数据并保存到原始Excel文件

file="rank2021"

df = pd.read\_sql\_query(f"SELECT \* FROM {file}", conn)

df.to\_excel("大学排行榜校友会2021.xlsx", index=False)

# 关闭数据库连接

conn.close()

#4

text = "四，合并校友会+分数线+软科"

print(f"{text:^{40}}")

import pandas as pd

import os

df1 = pd.read\_excel('分数线2023.xlsx')

df2 = pd.read\_excel('软科2024.xlsx')

common\_universities = set(df1['大学名称']).intersection(set(df2['大学名称']))

df1\_common = df1[df1['大学名称'].isin(common\_universities)]

df2\_common = df2[df2['大学名称'].isin(common\_universities)]

result = pd.merge(df1\_common, df2\_common, on='大学名称')

result.to\_excel('2023分数线+软科2024.xlsx', index=False)

#三合一

df1 = pd.read\_excel('2023分数线+软科2024.xlsx')

df2 = pd.read\_excel('大学排行榜校友会2021.xlsx')

common\_universities = set(df1['大学名称']).intersection(set(df2['大学名称']))

df1\_common = df1[df1['大学名称'].isin(common\_universities)]

df2\_common = df2[df2['大学名称'].isin(common\_universities)]

result = pd.merge(df1\_common, df2\_common, on='大学名称')

# 保存到新的Excel文件

result.to\_excel('分数线+校友会+软科.xlsx', index=False)

#多余的删掉

file\_path = "2023分数线+软科2024.xlsx"

if os.path.exists(file\_path):

os.remove(file\_path)

#5

text = "五，建立合并文件的数据库"

print(f"{text:^{40}}")

print("提示：必须将在总分和评分前加上校友会和软科的前缀")

import pandas as pd

import sqlite3

# 连接

conn = sqlite3.connect('test2.db')

df1 = pd.read\_excel('分数线+校友会+软科.xlsx')

df1.to\_sql('three', conn, if\_exists='replace', index=False)

# 查

def query(column, value):

df = pd.read\_sql\_query(f"SELECT \* FROM three WHERE {column} = '{value}'", conn)

return df

# 删

def delete(column):

df = pd.read\_sql\_query("SELECT \* FROM three", conn)

df = df.drop(columns=[column])

df.to\_sql('three', conn, if\_exists='replace', index=False)

# 改

def update(column, old\_value, new\_value):

df = pd.read\_sql\_query("SELECT \* FROM three", conn)

df.loc[df[column] == old\_value, column] = new\_value

df = df.rename(columns={column: new\_value})

df.to\_sql('three', conn, if\_exists='replace', index=False)

# 增

def add\_data(data):

df = pd.read\_sql\_query("SELECT \* from three", conn)

df.loc[len(df)] = data

df.to\_sql('three', conn, if\_exists='replace', index=False)

# 存回Excel

def save\_to\_excel():

df = pd.read\_sql\_query("SELECT \* FROM three", conn)

df.to\_excel("分数线+校友会+软科.xlsx", index=False)

while True:

user\_input = input("请输入操作（查询/更改（默认为更改表头）/删除/增加/退出）：")

if user\_input == "退出":

print("已退出")

break

if user\_input == "查询":

column = input("请输入要查询的列名：")

value = input("请输入要查询的值：")

result = query(column, value)

print(result)

elif user\_input == "更改":

column = input("请输入要更改的列名：")

old = input("请输入要更改的值：")

new = input("请输入新的值：")

update(column, old, new)

elif user\_input == "删除":

column = input("请输入要删除的列名：")

delete(column)

elif user\_input == "增加":

data = input("请输入增加的这一行：(以英文逗号隔开).").split(',')

add\_data(data)

else:

print("无效的操作")

save\_to\_excel()

# 关闭数据库连接

conn.close()

#6

text = "六，拟合曲线的绘制"

print(f"{text:^{40}}")

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

from scipy.stats import linregress

plt.rcParams['font.sans-serif'] = ['Microsoft YaHei'] # 来点和楷体不一样的

plt.rcParams['axes.unicode\_minus'] = False # 正确显示负号

file\_path = r"分数线+校友会+软科.xlsx"

data = pd.read\_excel(file\_path)

data.dropna(subset=['软科评分', '分数线', '校友会总分'], inplace=True)

soft\_score = data['软科评分']

score\_line= data['分数线']

total\_score = data['校友会总分']

university\_names = data['大学名称']

plt.figure(figsize=(15, 6))

# 第一个

plt.subplot(1, 2, 1)

sns.scatterplot(x=soft\_score, y=score\_line)

plt.title('软科评分与分数线的关系')

plt.xlabel('软科评分')

plt.ylabel('分数线')

slope1, intercept1, r\_value1, p\_value1, std\_err1 = linregress(soft\_score, score\_line)

line1 = slope1 \* soft\_score + intercept1

plt.plot(soft\_score, line1, 'r', label=f'拟合线 y={slope1:.2f}x+{intercept1:.2f}\nR²={r\_value1\*\*2:.2f}')

plt.legend()

# 添加大学名称注释

for i, name in enumerate(university\_names):

plt.annotate(name, (soft\_score[i], score\_line[i]))

# 第二个

plt.subplot(1, 2, 2)

sns.scatterplot(x=score\_line, y=total\_score)

plt.title('分数线与校友会总分的关系')

plt.xlabel('分数线')

plt.ylabel('校友会总分')

slope2, intercept2, r\_value2, p\_value2, std\_err2 = linregress(score\_line, total\_score)

line2 = slope2 \* score\_line + intercept2

plt.plot(score\_line, line2, 'g', label=f'拟合线 y={slope2:.2f}x+{intercept2:.2f}\nR²={r\_value2\*\*2:.2f}')

plt.legend()

for i, name in enumerate(university\_names):

plt.annotate(name, (score\_line[i], total\_score[i]))

plt.tight\_layout()

plt.show()

#7

text = "七，性价比指数计算"

print(f"{text:^{40}}")

print("分数线越低，评分越高，则性价比越高")

import pandas as pd

data = pd.read\_excel('分数线+校友会+软科.xlsx')

low\_cost\_performance = []

high\_cost\_performance = []

double\_performance = []

for index, row in data.iterrows():

university\_name = row['大学名称']

score\_line = row['分数线']

soft\_score = row['软科评分']

al\_score = row['校友会总分']

d1=score\_line - 0.18 \* soft\_score - 540.25

d2=al\_score - 0.2 \* score\_line + 54.4

if d1>0 and d2<0:

low\_cost\_performance.append((university\_name, score\_line, soft\_score, al\_score))

elif d1<0 and d2>0:

high\_cost\_performance.append((university\_name, score\_line, soft\_score, al\_score))

else:

double\_performance.append((university\_name, score\_line, soft\_score, al\_score))

def calculate\_cost\_performance1(row):

score\_line = row['分数线']

soft\_score = row['软科评分']

return round(0.18 \* soft\_score + 540.25 - score\_line,1)

def calculate\_cost\_performance2(row):

score\_line = row['分数线']

al\_score = row['校友会总分']

return round(al\_score - 0.2 \* score\_line + 54.4 ,1)

low\_cost\_performance\_df = pd.DataFrame(low\_cost\_performance, columns=['大学名称', '分数线', '软科评分','校友会总分'])

high\_cost\_performance\_df = pd.DataFrame(high\_cost\_performance, columns=['大学名称', '分数线', '软科评分','校友会总分'])

double\_performance\_df = pd.DataFrame(double\_performance, columns=['大学名称', '分数线', '软科评分','校友会总分'])

# 计算性价比指数并添加到DataFrame中

low\_cost\_performance\_df['性价比指数1'] = low\_cost\_performance\_df.apply(calculate\_cost\_performance1, axis=1)

low\_cost\_performance\_df['性价比指数2'] = low\_cost\_performance\_df.apply(calculate\_cost\_performance2, axis=1)

high\_cost\_performance\_df['性价比指数1'] = high\_cost\_performance\_df.apply(calculate\_cost\_performance1, axis=1)

high\_cost\_performance\_df['性价比指数2'] = high\_cost\_performance\_df.apply(calculate\_cost\_performance2, axis=1)

double\_performance\_df['性价比指数1'] = double\_performance\_df.apply(calculate\_cost\_performance1, axis=1)

double\_performance\_df['性价比指数2'] = double\_performance\_df.apply(calculate\_cost\_performance2, axis=1)

#排序

low\_cost\_performance\_df = low\_cost\_performance\_df.sort\_values(by='性价比指数1', ascending=True)

high\_cost\_performance\_df = high\_cost\_performance\_df.sort\_values(by='性价比指数1', ascending=False)

with pd.ExcelWriter('性价比分析表格.xlsx') as writer:

low\_cost\_performance\_df.to\_excel(writer, sheet\_name='性价比低', index=False)

high\_cost\_performance\_df.to\_excel(writer, sheet\_name='性价比高', index=False)

double\_performance\_df.to\_excel(writer, sheet\_name='两者皆有', index=False)

附件2、参考文献

1. 李宗民,刘培刚.大学计算机[M].北京:石油工业出版社,2021.12.
2. 何旭莉,刘培刚.程序设计Python版[M].北京:石油工业出版社,2021.9.
3. 即使再小的船也能远航.【爬虫】爬取大学排名信息[EB/OL].(2023-09-12)[2024-05-03].https://blog.csdn.net/qq\_57268251/article/details/129192351
4. 佩禾.Python安装第三方库[EB/OL].(2022-11-16)[2024-05-03].https://blog.csdn.net/weixin\_45867695/article/details/124943124