**Course Project Report**

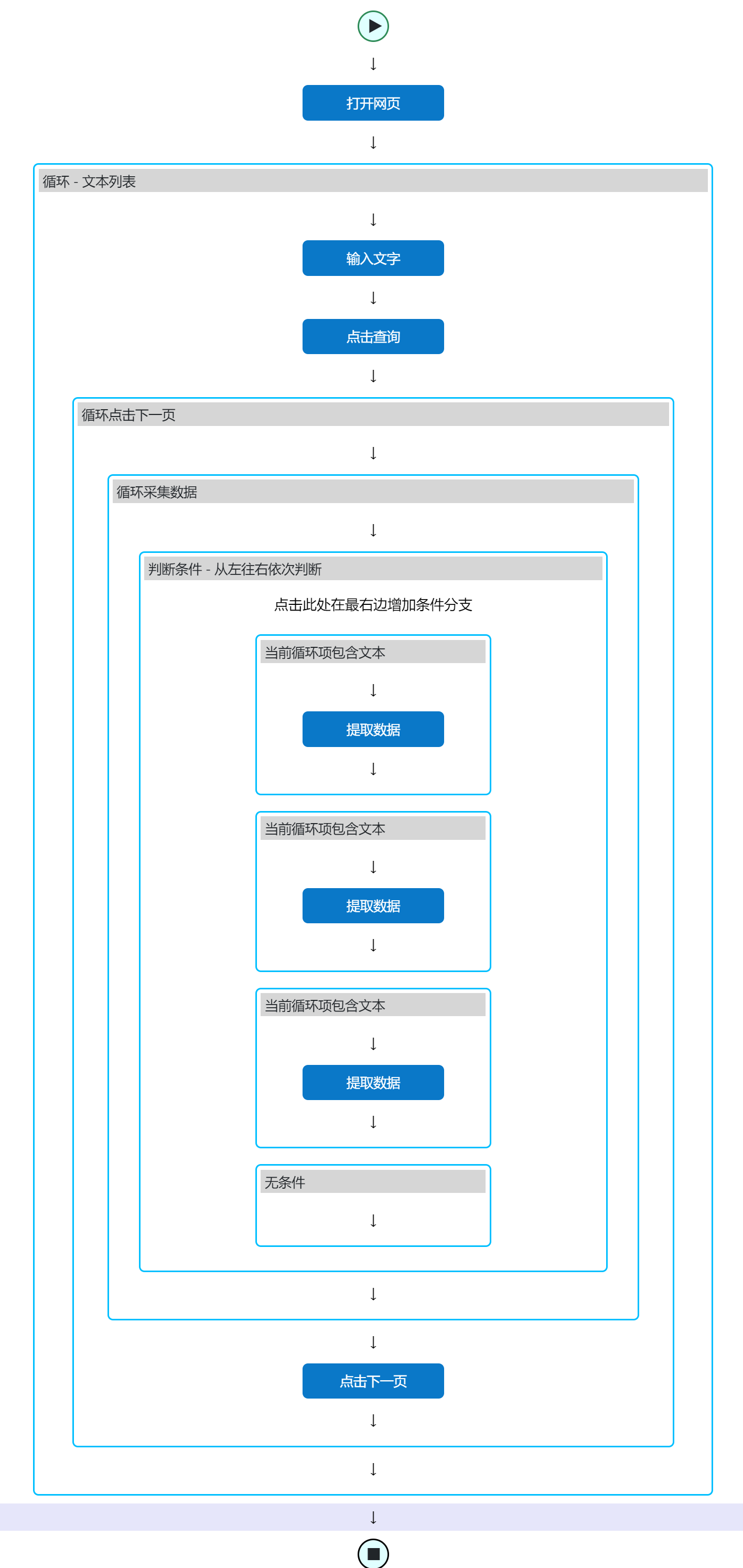
**Task 1:**

**爬取中国地震台网历史数据：**

1. **纬度分别大于10度、20度、30度、40度、50度区域的数据；**
2. **爬取查询结果页面前50页内容；**
3. **过滤结果，保存结果中含有西藏、四川、台湾的数据；**
4. **对保存的数据进行数据分析，可视化三地地震信息（可视化方式越丰富、信息表现越全面分数越高）。**

**报告中需要详细介绍设计思路和步骤（截图），并且展示可视化信息并进行分析。将爬取保存的数据文件命名为：小组名-Task1.csv（如有多个文件则分别命名后再压缩成一个压缩包，命名为小组名-Task1.zip） ，与报告一起上传至Moodle。**

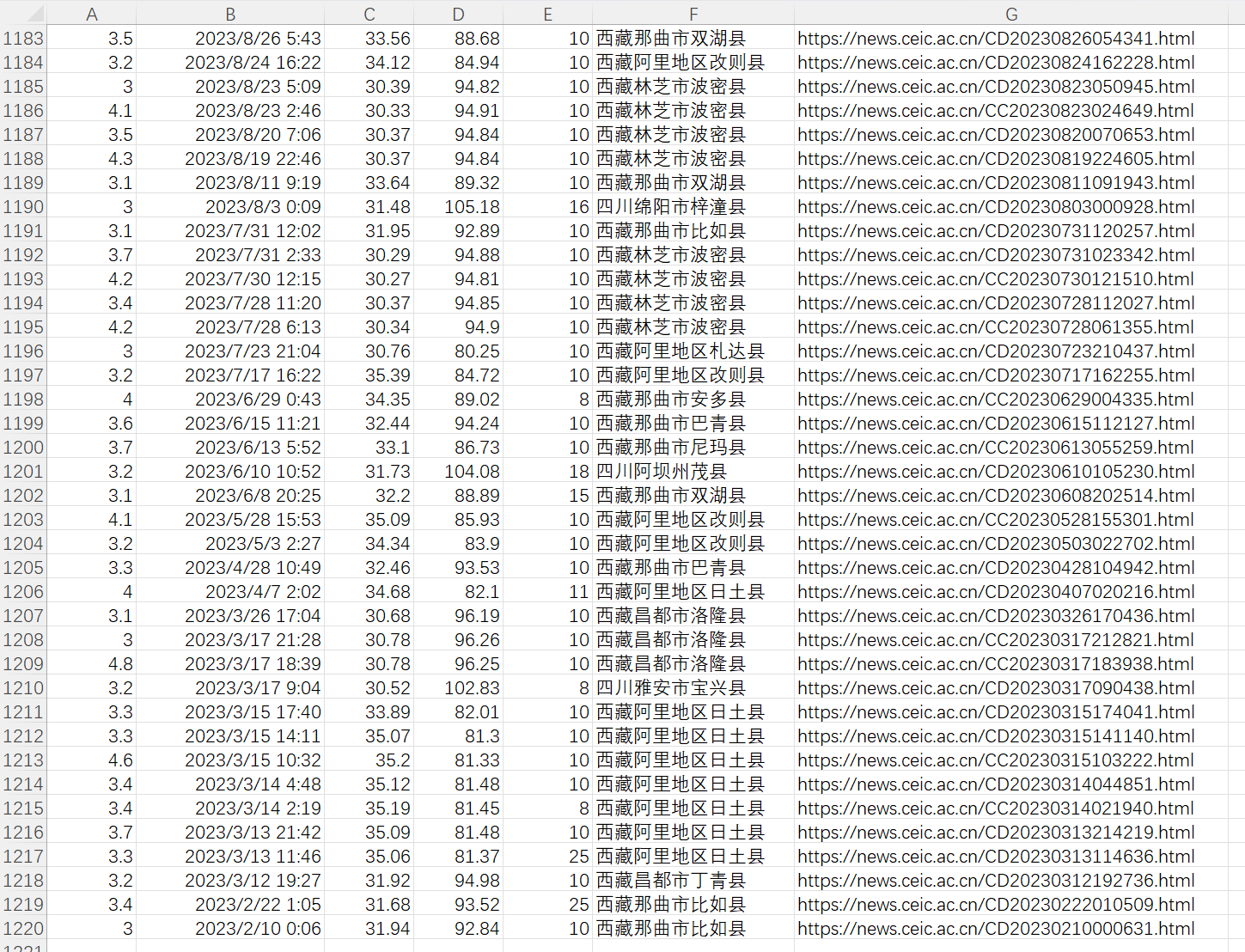
首先设计easyspider数据爬取脚本:



该脚本的逻辑为:打开网页后，循环输入维度并点击查询，点击查询后循环点击下一页，对于每次点击下一页，循环采集数据，如果字段有西藏或四川或台湾就采集，采集完后点击下一页，以此循环49次，来爬取50页的内容

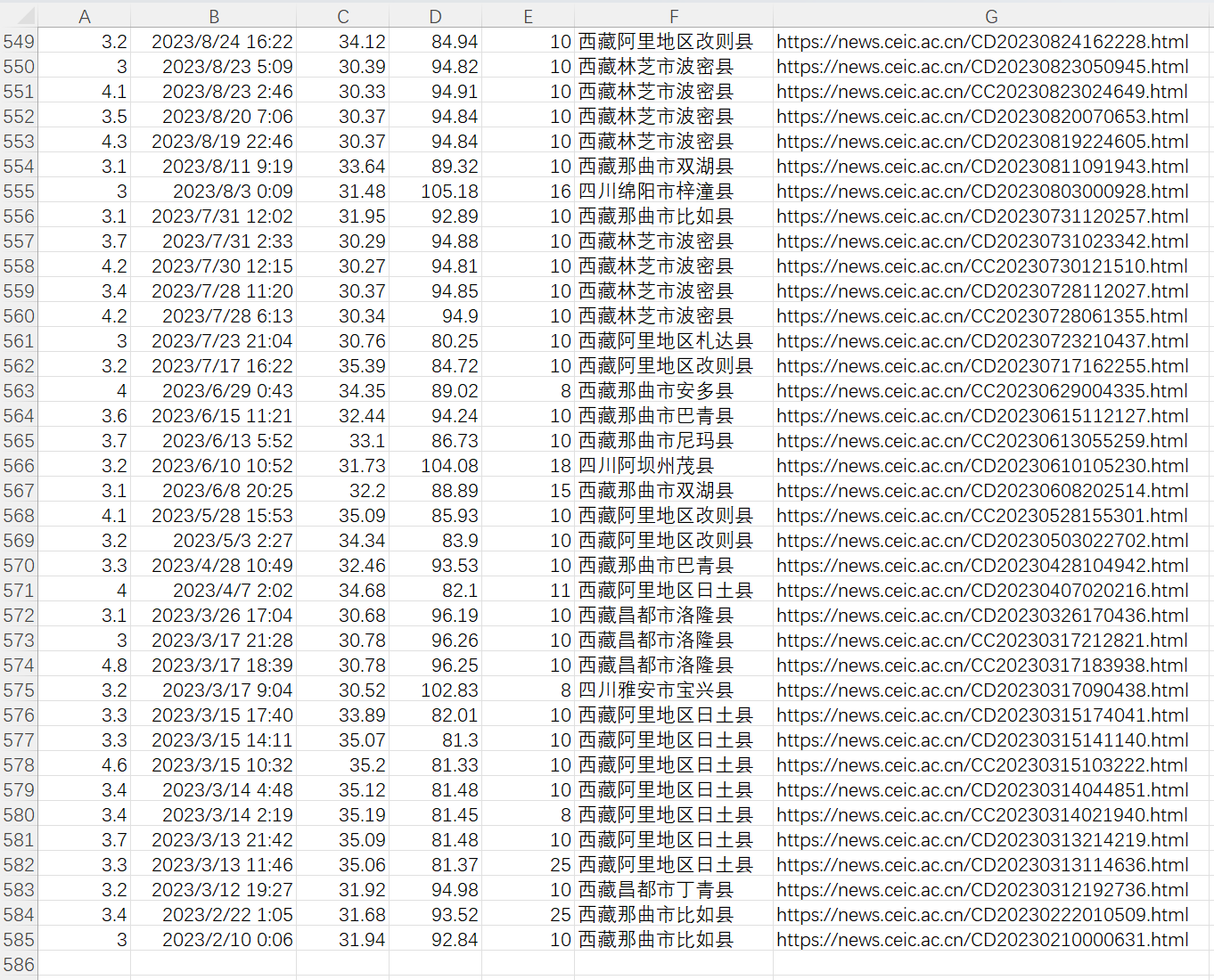
然后是简单的对数据进行分析，可以看到，任务的要求是，爬取纬度大于10的，纬度大于20的，可以很显然地看到，大于20的一定大于10，所以大于20的一定是大于10的一个子集，数据会有很多的重复，所以我爬取了两个数据集，一个是没去过重的，一个是去过重的。

没去过重的数据如下:



一共1219条数据

去重后的数据如下:



一共584条数据

接下来对数据进行可视化:

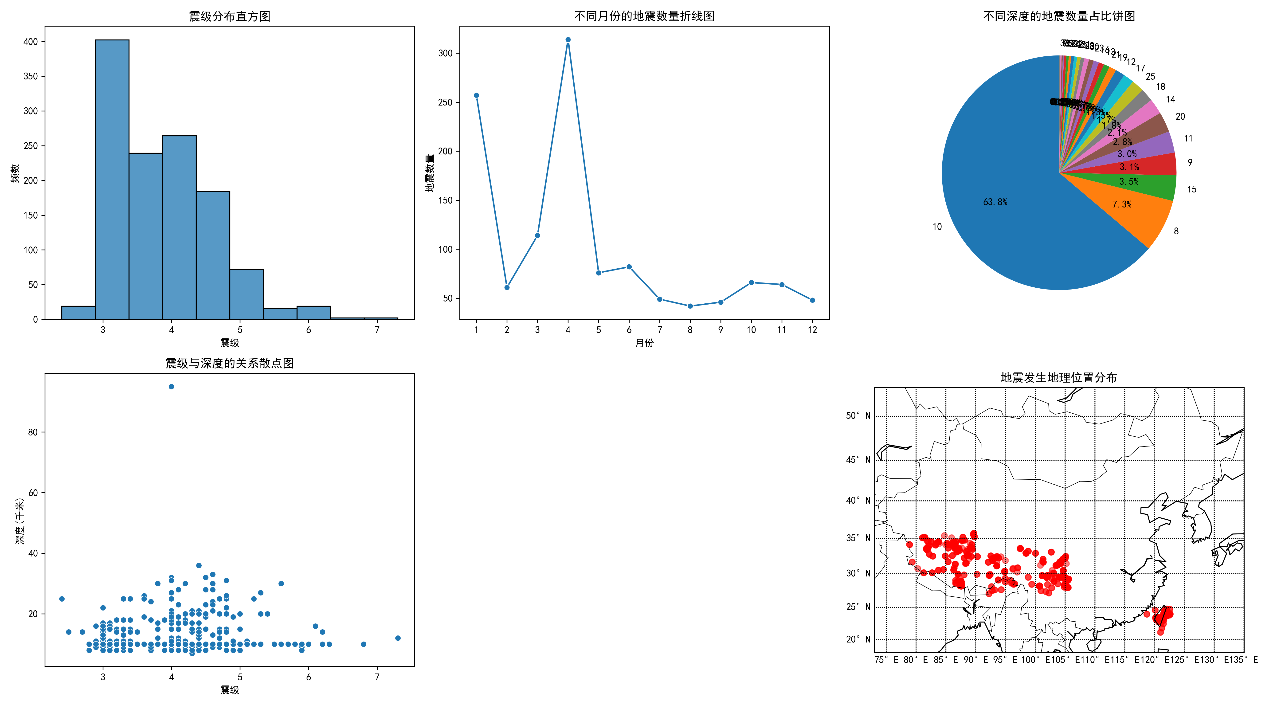
因为要保证可视化方法丰富，且信息表现全面，所以我设计了五幅图，并分别从不同角度对数据进行分析。

首先是对未去重的数据进行可视化:

可视化的代码如下:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
*import* seaborn *as* sns  
*from* mpl\_toolkits.basemap *import* Basemap  
  
  
  
# 设置图片清晰度  
plt.rcParams['figure.dpi'] = 300  
  
# 设置中文字体  
plt.rcParams['font.sans-serif'] = ['SimHei']  
plt.rcParams['axes.unicode\_minus'] = *False*# 加载数据  
df = pd.read\_csv('AA-Task1.csv')  
  
# 转换发震时刻(UTC+8)列为日期时间类型  
df['发震时刻(UTC+8)'] = pd.to\_datetime(df['发震时刻(UTC+8)'])  
  
# 提取月份信息  
df['发震月份'] = df['发震时刻(UTC+8)'].dt.month  
  
# 剔除链接地址列  
df = df.drop(columns='链接地址')  
  
# 创建一个2x3的子图布局  
fig, axes = plt.subplots(2, 3, figsize=(18, 10))  
  
# 可视化1：震级分布直方图  
sns.histplot(df['震级(M)'], bins=10, kde=*False*, ax=axes[0, 0])  
axes[0, 0].set\_title('震级分布直方图')  
axes[0, 0].set\_xlabel('震级')  
axes[0, 0].set\_ylabel('频数')  
  
# 可视化2：不同月份的地震数量折线图  
monthly\_counts = df['发震月份'].value\_counts().sort\_index()  
sns.lineplot(x=monthly\_counts.index, y=monthly\_counts.values, marker='o', ax=axes[0, 1])  
axes[0, 1].set\_title('不同月份的地震数量折线图')  
axes[0, 1].set\_xlabel('月份')  
axes[0, 1].set\_ylabel('地震数量')  
axes[0, 1].set\_xticks(range(1, 13))  
  
# 可视化3：不同深度的地震数量占比饼图  
depth\_counts = df['深度(千米)\t'].value\_counts()  
axes[0, 2].pie(depth\_counts.values, labels=depth\_counts.index, autopct='%1.1f%%', startangle=90)  
axes[0, 2].set\_title('不同深度的地震数量占比饼图')  
  
# 可视化4：震级与深度的关系散点图  
sns.scatterplot(x='震级(M)', y='深度(千米)\t', data=df, ax=axes[1, 0])  
axes[1, 0].set\_title('震级与深度的关系散点图')  
axes[1, 0].set\_xlabel('震级')  
axes[1, 0].set\_ylabel('深度(千米)')  
  
# 隐藏第5个子图  
axes[1, 1].axis('off')  
  
# 可视化6：地震发生地理位置分布（使用Basemap绘制中国地图）  
m = Basemap(llcrnrlon=73, llcrnrlat=18, urcrnrlon=135, urcrnrlat=53, projection='merc', ax=axes[1, 2])  
m.drawcoastlines()  
m.drawcountries()  
m.drawparallels(range(15, 55, 5), labels=[1, 0, 0, 0])  
m.drawmeridians(range(70, 140, 5), labels=[0, 0, 0, 1])  
x, y = m(df['经度(°)\t'].tolist(), df['纬度(°)'].tolist())  
m.scatter(x, y, c='r', marker='o', alpha=0.5)  
axes[1, 2].set\_title('地震发生地理位置分布')  
  
# 自动调整子图布局  
plt.tight\_layout()  
plt.show()

最终可视化的结果如下:



根据可视化数据，可以对数据进行以下简单分析:

1.震级分布：

震级范围在2.4-7.3级之间，集中分布在3.0-4.0级（占比约70%），属于中小规模地震。大于等于5.0级的高震级事件较少，主要集中在台湾地区（如2024年4月23日台湾花莲县海域6.3级地震）。

2.高频震级区间：

3.0-3.5级地震占比最高（约35%），其次是3.5-4.0级（约25%），符合中国地震以中小震级为主的特征。

3.月份分布：

地震数量在月份上无显著季节性规律，但3-4月、7-8月、10-12月相对高频，可能与地壳活动周期有关。2025年1月地震数量最多（约150次），可能为数据采集周期内的活跃期。

4.时间连续性：

数据覆盖两年多，记录密度均匀，未发现明显数据缺失或异常集中现象。

5.深度集中区间：

超90%的地震深度在10-20千米之间，符合全球浅源地震占比高的规律。少数地震深度超过30千米（如2025年4月9日台湾宜兰县33千米深度），深源地震极少。

6.震级与深度关系：

震级与深度无显著线性相关，浅源地震中既有3-4级的小震，也存在中强震（如2024年6月1日台湾花莲5.2级）。

7.高频区域：

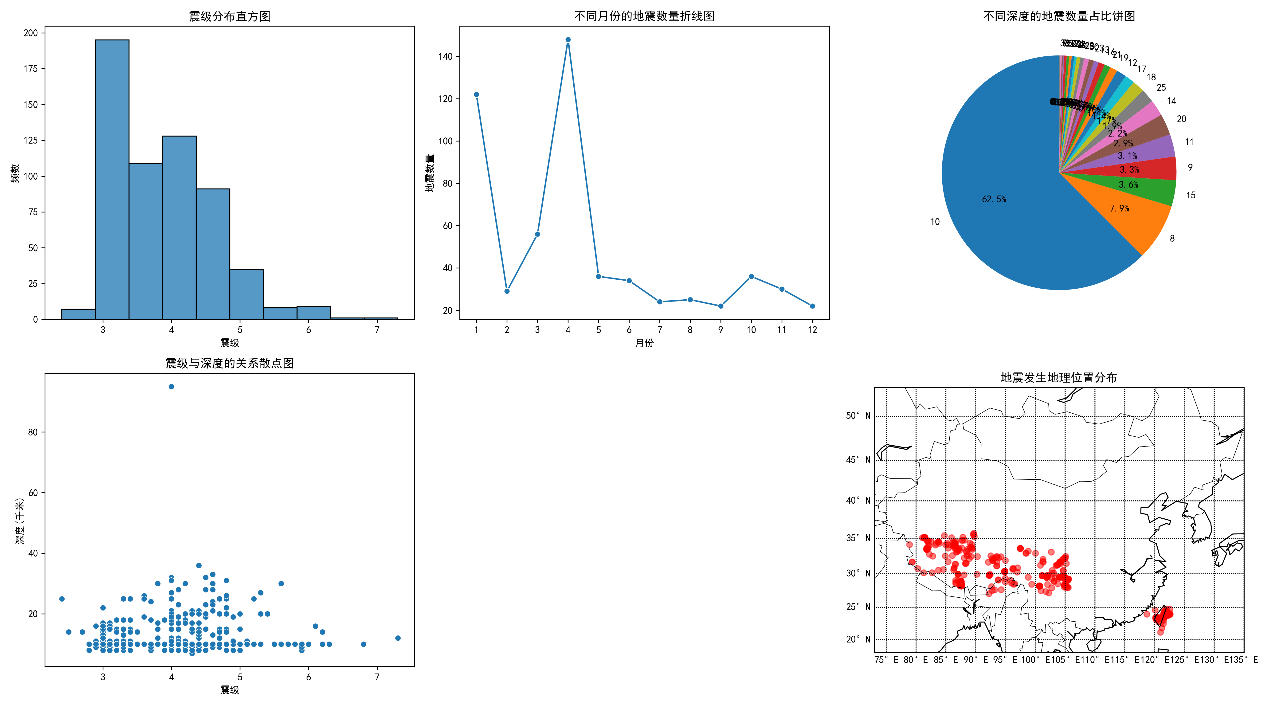
西藏自治区（尤其是日喀则市、那曲市）和四川省（内江、甘孜、凉山州）地震数量最多，与青藏高原东缘构造活动密切相关。台湾地区（花莲县、台南市、宜兰县）震级较高且频繁，因位于环太平洋地震带，地壳运动活跃。

8.分布规律：

地震集中在西南地区（西藏、四川）和东南沿海（台湾），符合中国地震带分布（青藏地震区、台湾地震区）。

然后是对去了重的数据进行可视化:

代码仍然是使用同一份，只不过是切换了数据,最终可视化后的结果如下所示：



可以看到，基本就只有数值的变化，比如第一张图从400变成了200，第二张图从300变成了140，第三张图的比例有点微小的变化，其余基本都和未去重的数据集差不多。

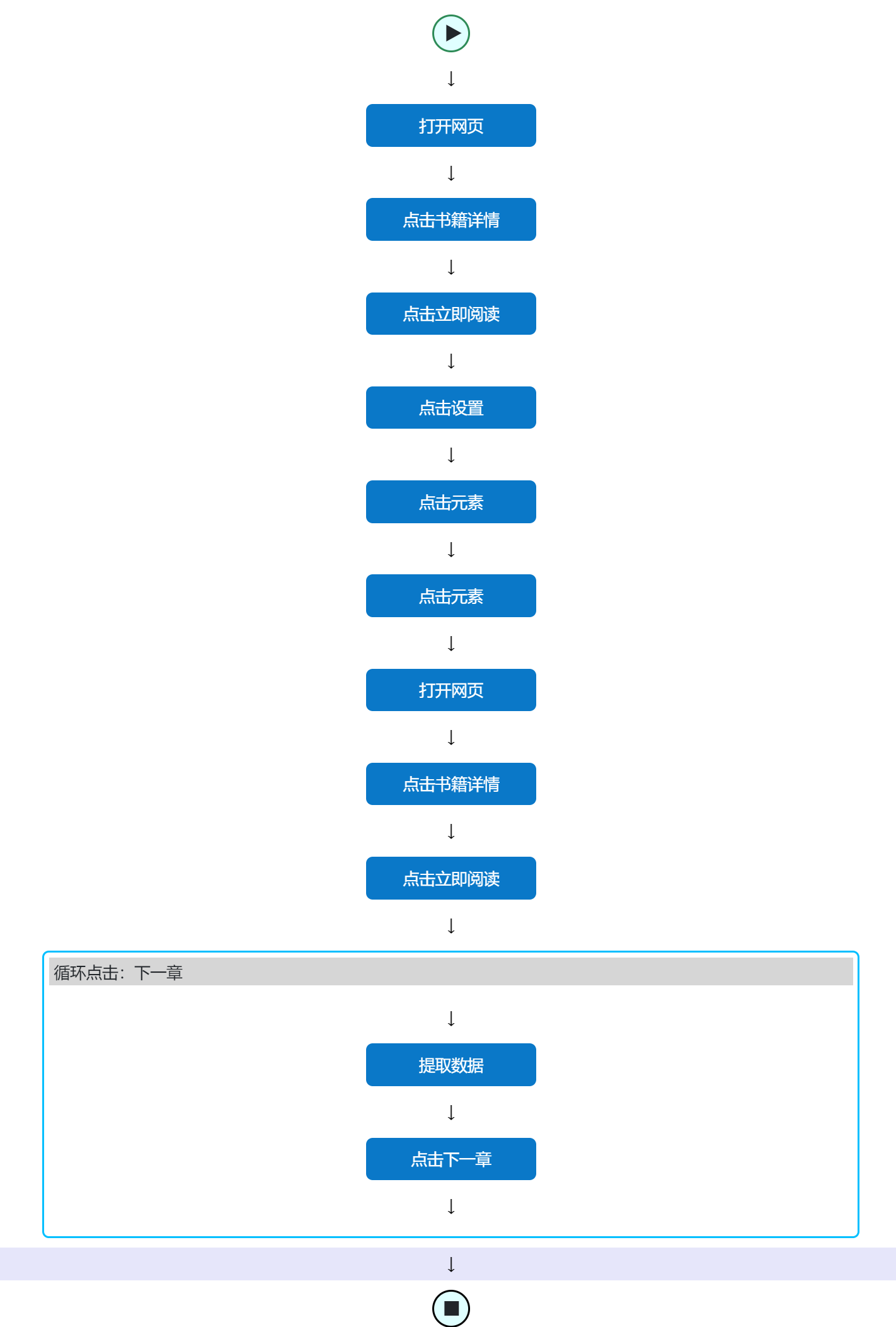
**Task 2:**

**爬取纵横中文网小说：**

1. **TOP10月票榜前五的小说；**
2. **每本小说爬取前50章（少于50章的爬取全部章节）；**
3. **将爬取的数据进行数据清洗和整理，以方便阅读的格式和文件分别保存5本小说；**
4. **为5本小说分别绘制词云并保存成图片格式。**

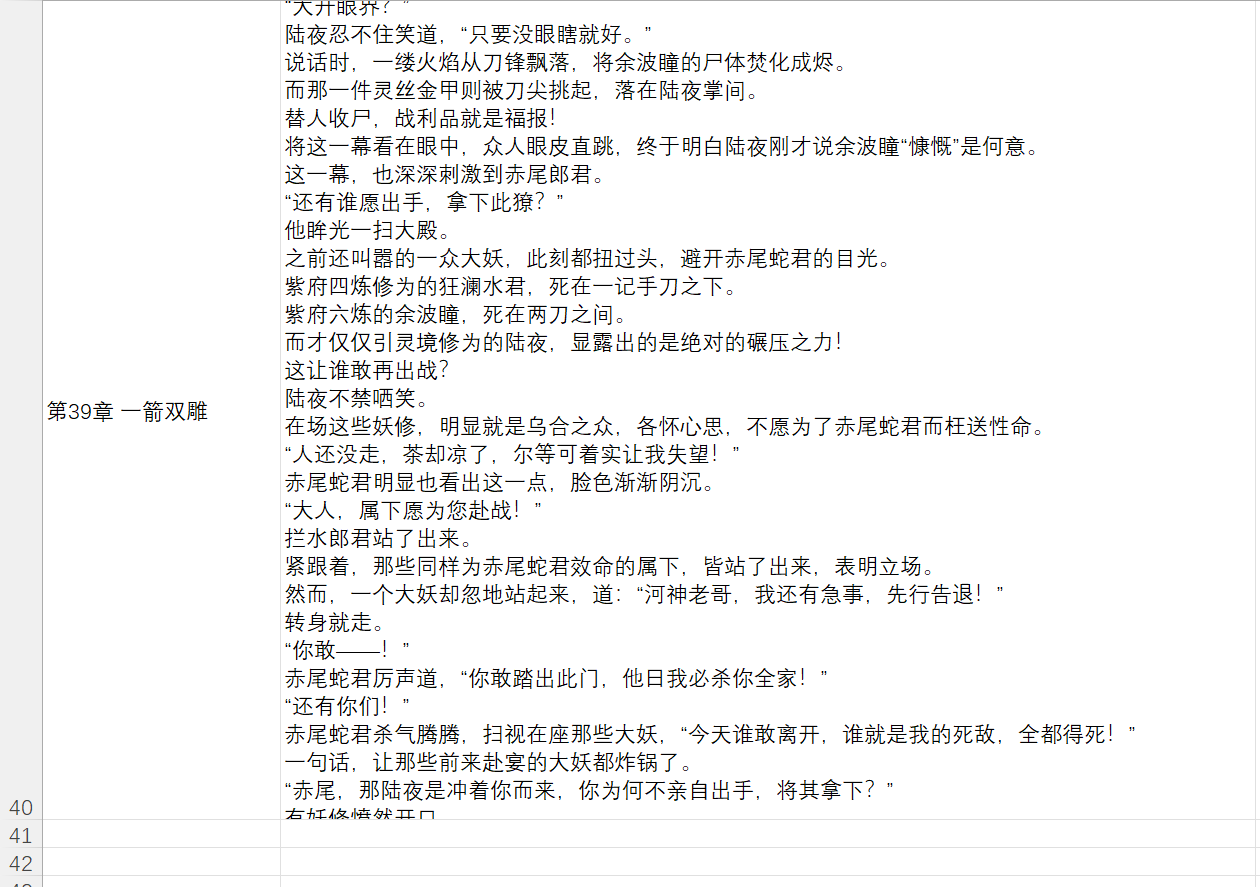
**报告中需要详细介绍设计思路和步骤（截图），并且展示5本小说的词云。将数据处理后的5本小说和词云图片文件一起打包为一个zip文件，文件名：小组名-Task2.zip，与报告一起上传至Moodle。**

首先设计easyspider数据爬取脚本:



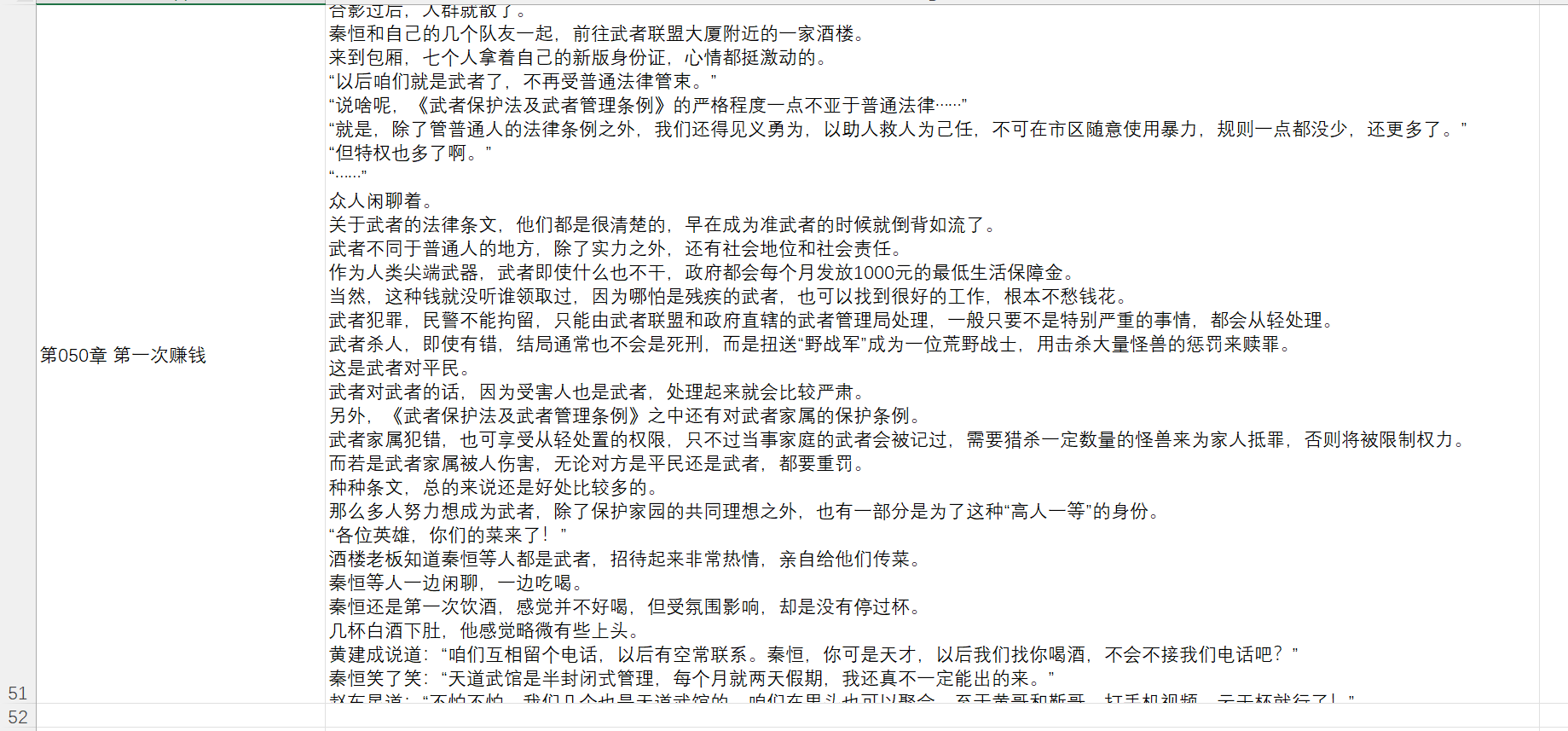
该脚本的逻辑为:打开纵横小说网，点击书籍详情，点击立即阅读，打开书籍右边的设置按钮，把显示段评给关闭，关闭后重新打开一开始的网页，重复打开操作一直回到原来的位置，这整一个过程就是为了把段评关闭，使得爬取文章内容时不会有额外的数字。接着就进行爬取数据并循环点击下一章的操作，把整本书爬取即可，另外4本书也是同样的逻辑，这边省略。

第一本书的爬取内容：

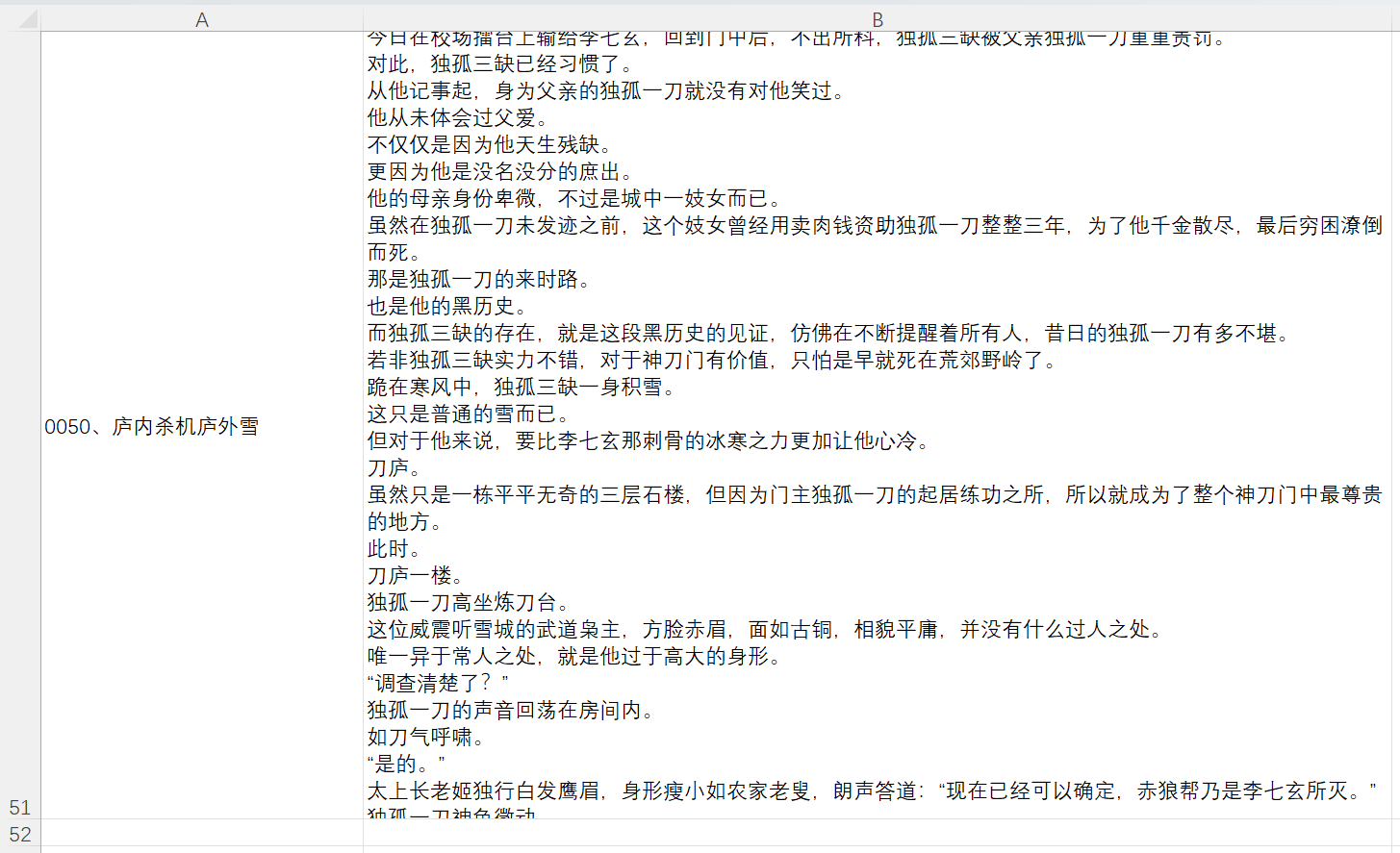


不足50章，这边爬取了最新的第39章

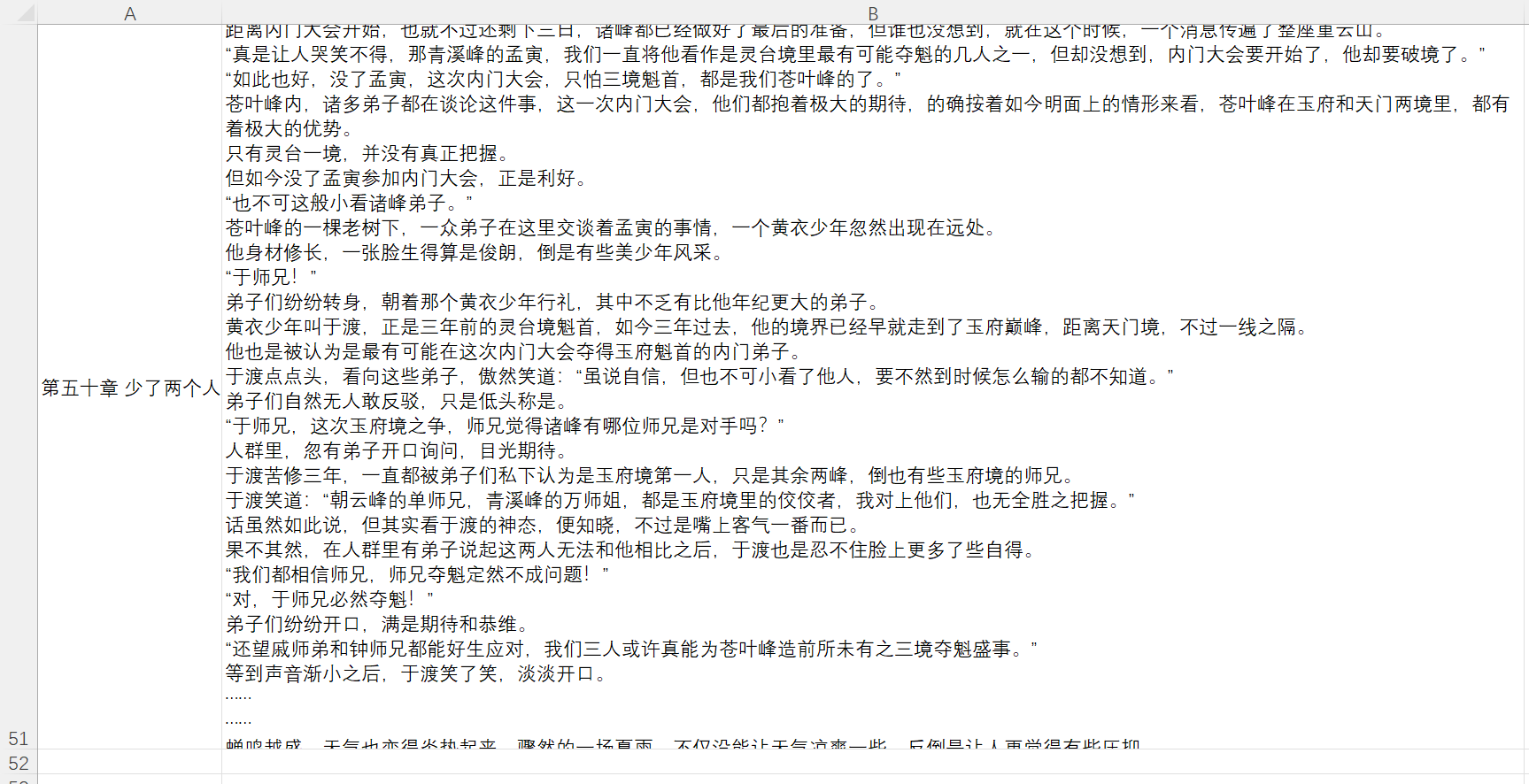
第二本书的爬取内容：



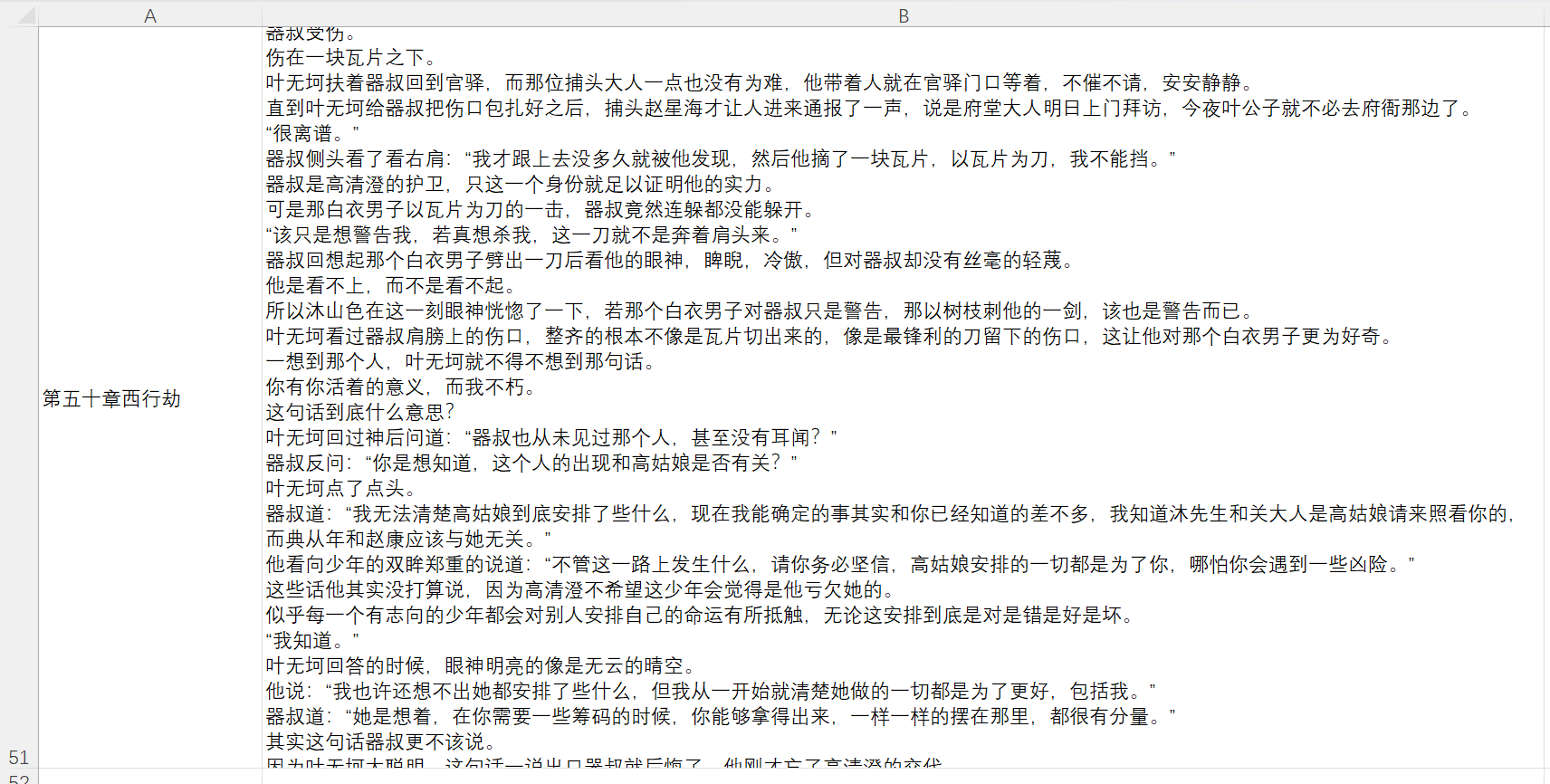
第三本书的爬取内容：



第四本书的爬取内容：



第五本书的爬取内容：



然后是将爬取的数据进行数据清洗和整理，以方便阅读的格式和文件分别保存5本小说

首先是数据清洗和整理，因为只有章节名和内容，所以这边不需要进行清洗，直接把该文件保存为方便阅读的格式即可。

因为要适合阅读，这边打算转为PDF格式

首先编写python代码，先把CSV文件的每一章存储为一个txt文件，然后再把这几章按顺序合并，具体的python代码如下:

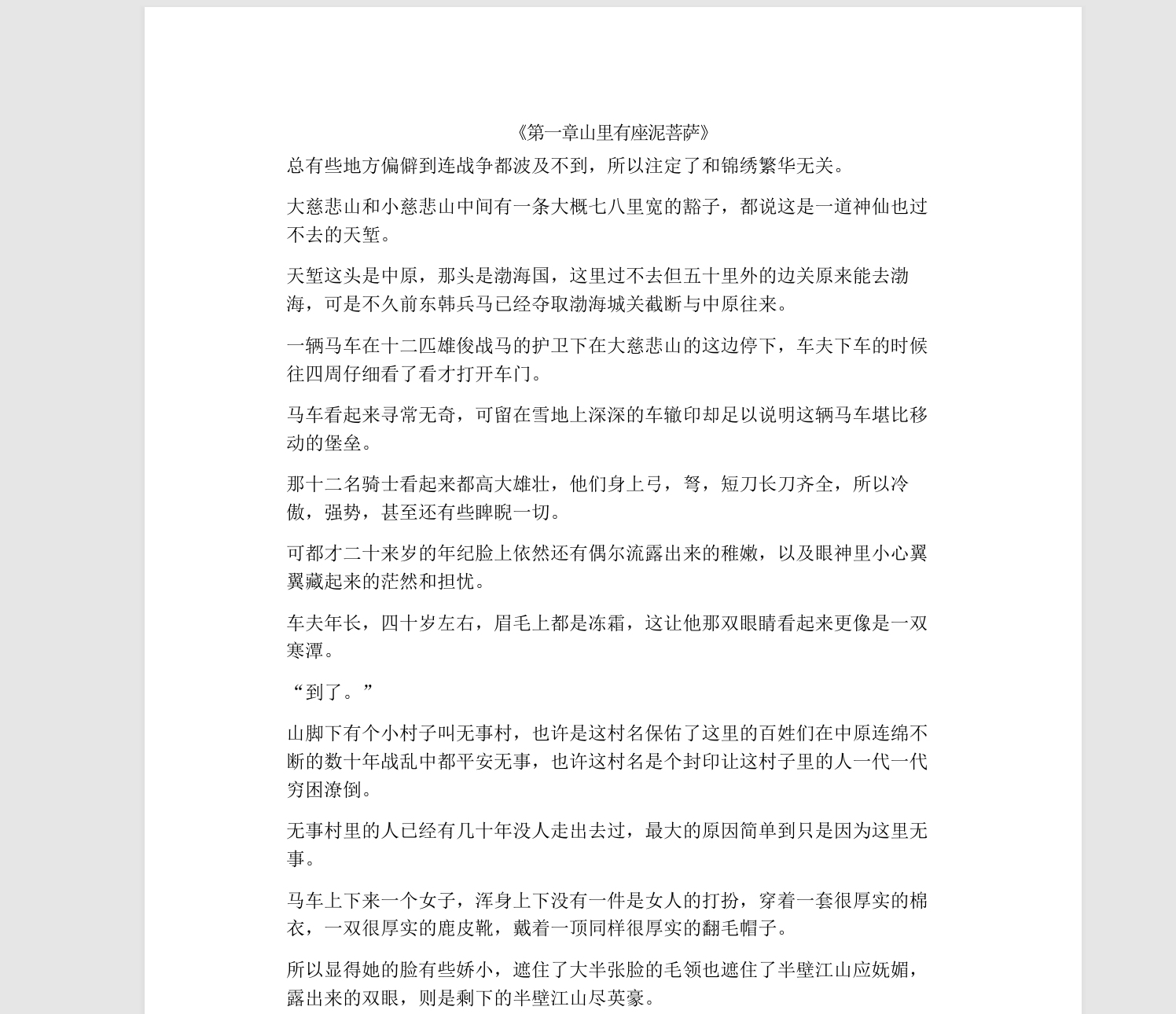
*import* csv  
*import* os  
*import* re  
  
data='AA-Task2-5.csv'  
name='天下长宁'  
  
  
file\_name=name+'.txt'  
*def* csv\_to\_txt(*csv\_path*, *output\_dir*=name):  
 os.makedirs(*output\_dir*, exist\_ok=*True*)  
 encodings = ['utf-8', 'gbk', 'gb2312']  
 detected\_enc = *None  
 for* enc *in* encodings:  
 *try*:  
 *with* open(*csv\_path*, 'r', encoding=enc) *as* f:  
 next(csv.reader(f))  
 detected\_enc = enc  
 *break  
 except* UnicodeDecodeError:  
 *continue  
 else*:  
 *raise* ValueError("无法检测文件编码")  
  
 *with* open(*csv\_path*, 'r', encoding=detected\_enc) *as* f:  
 reader = csv.DictReader(f)  
 *for* row *in* reader:  
 chapter\_name = re.sub(r'[\\/\*?:"<>|]', '\_', row['章节名'])  
 file\_path = os.path.join(*output\_dir*, f"{chapter\_name}.txt")  
 *with* open(file\_path, 'w', encoding='utf-8') *as* f\_out:  
 f\_out.write(f"《{row['章节名']}》\n\n")  
 f\_out.write(row['内容'].replace('。', '。\n'))  
  
  
*def* merge\_txt(*output\_dir*=name, *merged\_file*=file\_name):  
 *def* chinese\_to\_number(*cn\_num*):  
 num\_map = {  
 '零': 0, '一': 1, '二': 2, '三': 3, '四': 4,  
 '五': 5, '六': 6, '七': 7, '八': 8, '九': 9,  
 '十': 10, '百': 100, '千': 1000, '万': 10000,  
 }  
 *if cn\_num in* num\_map:  
 *return* num\_map[*cn\_num*]  
 *if cn\_num*.startswith('十') *and* len(*cn\_num*) > 1:  
 *return* 10 + num\_map.get(*cn\_num*[1], 0)  
 *if* '十' *in cn\_num*:  
 parts = *cn\_num*.split('十')  
 *if* len(parts) == 1: # 二十  
 *return* num\_map.get(parts[0], 0) \* 10  
 *elif* len(parts) == 2: # 二十一  
 *return* num\_map.get(parts[0], 0) \* 10 + num\_map.get(parts[1], 0)  
 *return* 0  
 *def* get\_chapter\_order(*file*):  
 match\_cn = re.match(r'第([一二三四五六七八九十零百]+)章[\_\-]?', *file*)  
 *if* match\_cn:  
 cn\_num = match\_cn.group(1)  
 *return* chinese\_to\_number(cn\_num)  
 match\_ar = re.match(r'第(\d+)章', *file*)  
 *if* match\_ar:  
 *return* int(match\_ar.group(1))  
 match\_num = re.match(r'^(\d+)', *file*.split('\_')[0])  
 *if* match\_num:  
 *return* int(match\_num.group(1))  
 *return* float('inf')  
 *with* open(*merged\_file*, 'w', encoding='utf-8') *as* f\_out:  
 files = sorted(os.listdir(*output\_dir*), key=*lambda x*: (  
 *not* get\_chapter\_order(x) == float('inf'),  
 get\_chapter\_order(x)  
 ))  
 *for* file *in* files:  
 *if* file.endswith('.txt'):  
 *with* open(os.path.join(*output\_dir*, file), 'r', encoding='utf-8') *as* f\_in:  
 f\_out.write(f\_in.read() + '\n\n')  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 csv\_to\_txt(data)  
 merge\_txt()

直接更改前两行要使用的csv文件名和要生成的书名就能更改要生成的书了。

得到的合并的txt文件如下所示:



因为要转为PDF格式，先复制到word中，然后利用word快速修改格式，接着直接转为PDF即可，最终的PDF如下所示:



具体可以查看一同提交的文件。

然后就是为5本小说分别绘制词云并保存成图片格式

因为要绘制词云，所以使用CSV文件的内容那一列即可，具体实现的python代码如下:

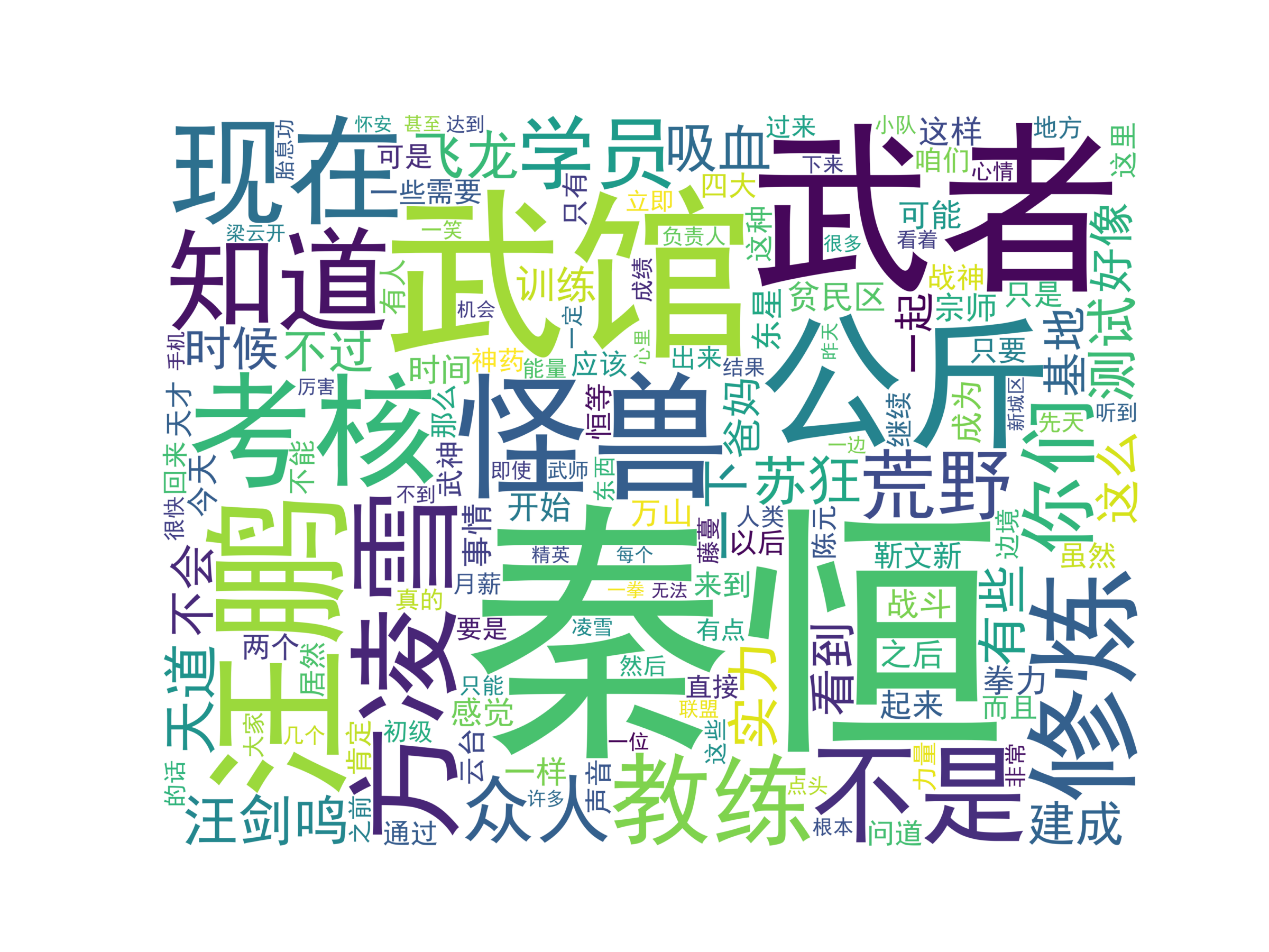
# -\*- coding: utf-8 -\*-  
*import* pandas *as* pd  
*import* jieba  
*import* matplotlib.pyplot *as* plt  
*import* os  
*from* wordcloud *import* WordCloud  
*from* collections *import* Counter  
*from* PIL *import* ImageFont  
*def* detect\_chinese\_font():  
 *if* os.name == 'nt':  
 font\_candidates = [  
 'C:/Windows/Fonts/simhei.ttf', # 黑体  
 'C:/Windows/Fonts/msyh.ttc', # 微软雅黑  
 'C:/Windows/Fonts/simfang.ttf', # 仿宋  
 'C:/Windows/Fonts/simkai.ttf', # 楷体  
 ]  
 *else*:  
 font\_candidates = [  
 '/usr/share/fonts/truetype/simhei.ttf',  
 '/usr/share/fonts/opentype/noto/NotoSansCJK-Regular.ttc'  
 ]  
 *for* font\_path *in* font\_candidates:  
 *if* os.path.exists(font\_path):  
 *return* font\_path  
 *raise* FileNotFoundError("未找到可用的中文字体，请手动安装或指定路径")  
*def* main():  
 *try*:  
 # ==================== 配置部分 ====================  
 csv\_file = 'AA-Task2-1.csv'  
 encoding = 'gbk'  
 output\_image = 'novel\_wordcloud.png'  
 max\_words = 200  
 # ================================================  
 font\_path = detect\_chinese\_font()  
 print(f"使用的字体路径: {font\_path}")  
 ImageFont.truetype(font\_path, 12)  
 df = pd.read\_csv(csv\_file, encoding=encoding, on\_bad\_lines='warn')  
 text = ' '.join(df['内容'].astype(str)) # 合并所有文本  
 # 自定义停用词列表  
 stopwords = {  
 '说道', '一个', '没有', '就是', '可以', '还是', '已经', '我们', '他们', '这个', '那个',  
 '自己', '什么', '怎么', '因为', '如果', '但是', '所以', '这是', '还有', '...', '，', '。',  
 '！', '？', '、', '的', '了', '在', '是', '我', '你', '他', '她', '它', '就', '不', '都',  
 '着', '有', '也', '人', '到', '说', '要', '去', '会', '对', '把', '又', '而', '从', '让',  
 '被', '之', '等', '与', '年', '日', '月', '上', '下', '中', '出', '过', '给', '只', '时'  
 }  
 words = jieba.lcut(text)  
 filtered\_words = [word *for* word *in* words *if* len(word) > 1 *and* word *not in* stopwords]  
 word\_counts = Counter(filtered\_words)  
 top\_words = word\_counts.most\_common(150)  
 wc = WordCloud(  
 font\_path=font\_path,  
 background\_color='white',  
 max\_words=max\_words,  
 width=1600,  
 height=1200,  
 collocations=*False*,  
 margin=2,  
 scale=2  
 )  
 wordcloud = wc.generate\_from\_frequencies(dict(top\_words))  
 plt.figure(figsize=(16, 12), dpi=150)  
 plt.imshow(wordcloud, interpolation='bilinear')  
 plt.axis('off')  
 plt.show()  
 wordcloud.to\_file(output\_image)  
 print(f"词云已保存至: {os.path.abspath(output\_image)}")  
 *except* UnicodeDecodeError *as* e:  
 print(f"编码错误: {e}\n请尝试修改encoding参数为: gb18030 或 utf-8")  
 *except* FileNotFoundError *as* e:  
 print(f"文件错误: {e}\n请确认: 1.CSV文件存在 2.字体路径正确")  
 *except* Exception *as* e:  
 print(f"发生未预期错误: {str(e)}")  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 main()

通过这份代码修改所使用的csv文件即可生成不同小说的词云，生成后的五本小说的词云分别如下:

第一本:



第二本:



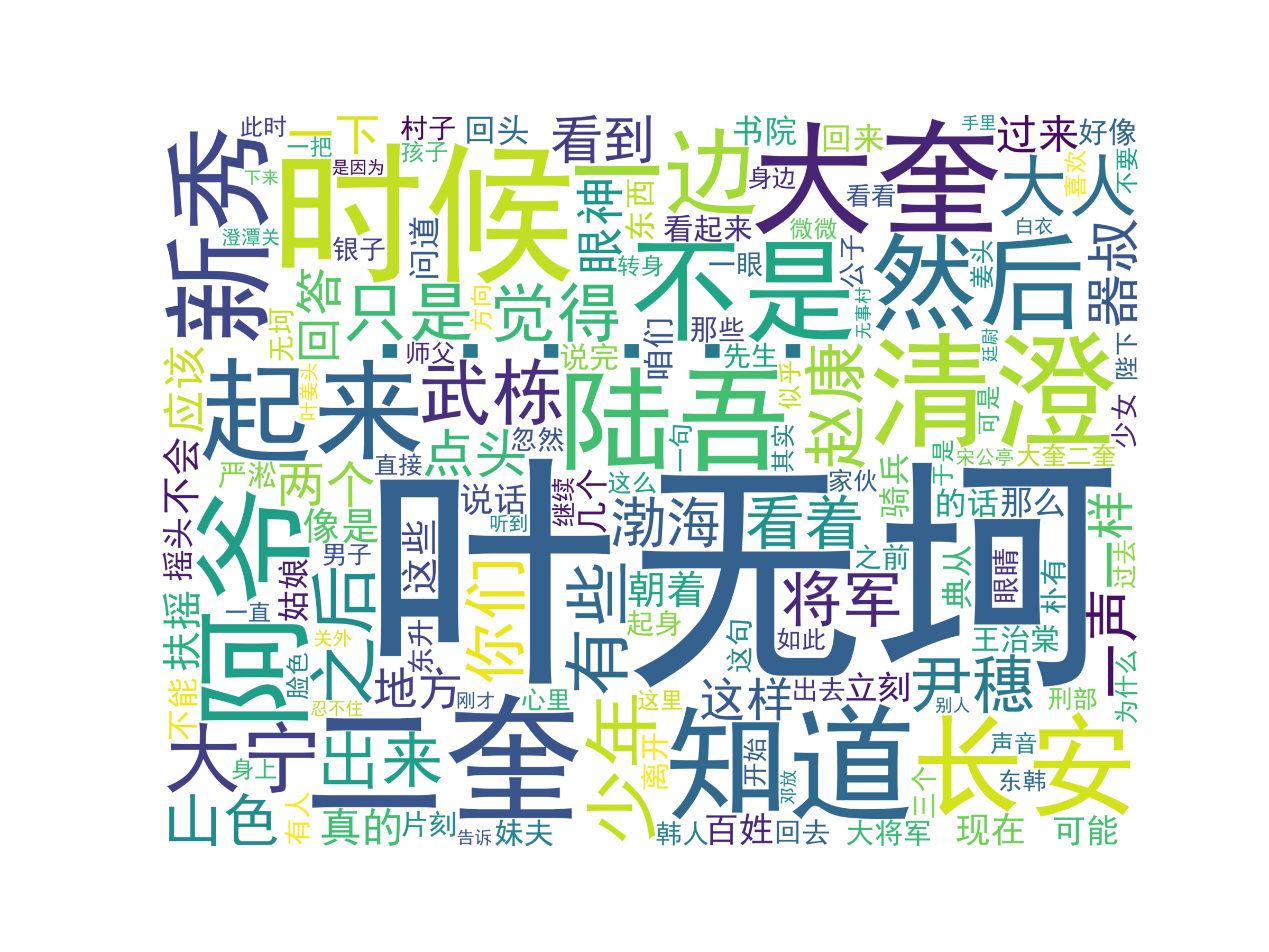
第三本:



第四本:



第五本:



**Task 3:**

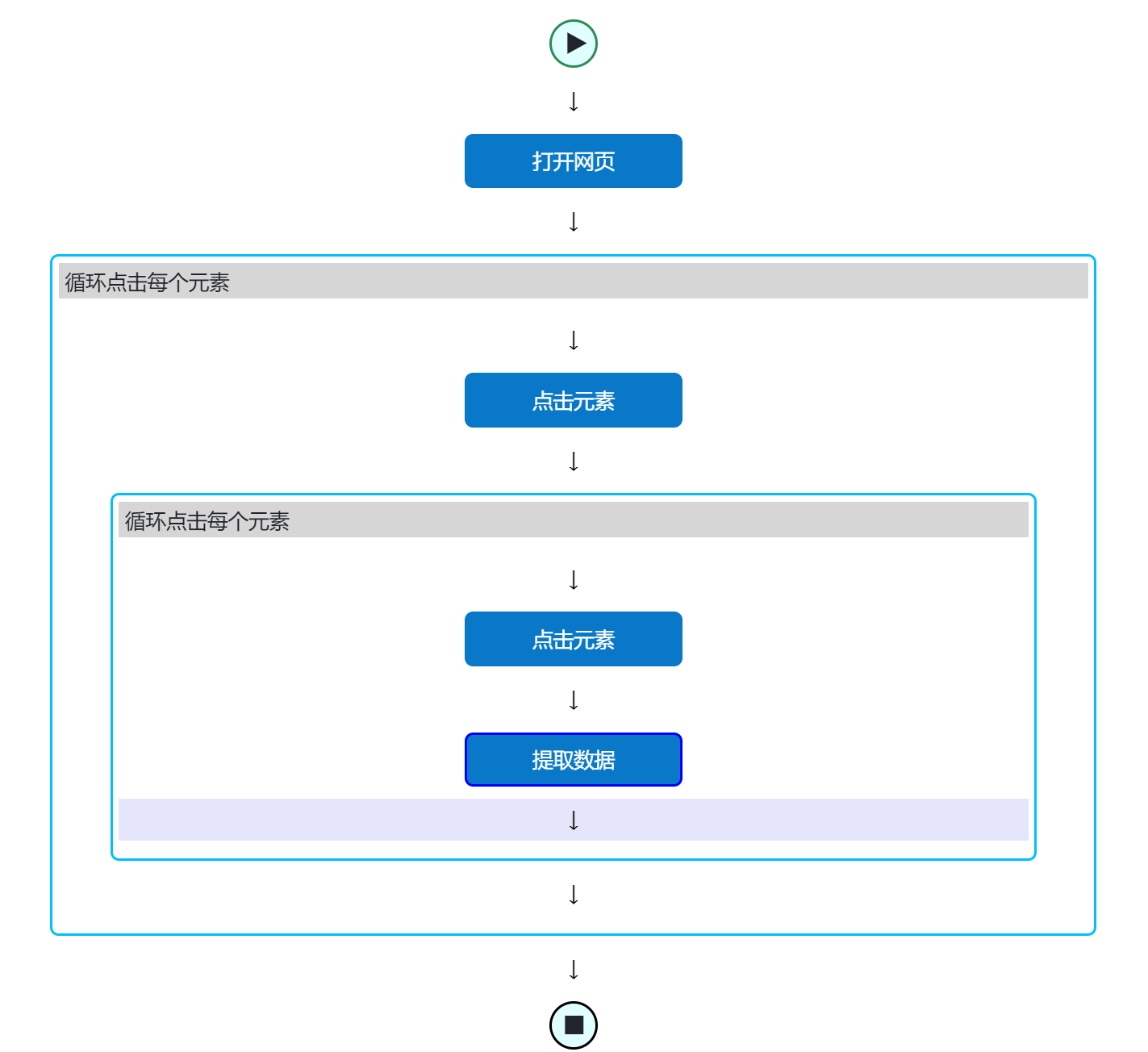
**尝试用大数据分析技术分析出杭州、宁波、温州哪个城市更宜居？**

1. **本任务要求运用大数据分析技术，通过收集、处理和分析多源数据，构建一个科学的评价体系，找出杭州、宁波和温州三个城市中哪个更宜居。需要完成从数据采集、清洗、分析到可视化的全流程工作，并最终形成分析报告；**
2. **需要用到的技术：数据采集（爬虫）、数据预处理、数据分析、数据可视化等；**
3. **需要分析的数据维度参考：**
   1. **经济维度：人均GDP、就业率、房价收入比、消费水平指数等；**
   2. **环境维度：空气质量指数（AQI）、水质状况、绿化覆盖率、年平均气温、极端天气频率等；**
   3. **基础设施维度：公共交通便利性（地铁里程、公交线路）、医疗资源(每千人床位数、三甲医院数量)、教育资源(中小学质量、高校数量)、商业设施密度等；**
   4. **社会文化维度：文化场馆数量(博物馆、图书馆、剧院)、休闲娱乐设施、城市安全指数(犯罪率)、人口密度等；**
   5. **生活便利维度：外卖/快递便利度、便利店/超市密度、社区服务设施、数字政务便利度等。**
4. **任务要求：**
   1. **数据收集：从多个不同来源收集数据，确保数据的全面性和代表性**
   2. **数据处理：清洗不一致、缺失的数据，进行必要的标准化处理，构建综合评价指标体系；**
   3. **分析建模：设计合理的权重分配方案，应用适当的分析模型，进行敏感性分析验证结果稳健性；**
   4. **可视化呈现：制作交互式可视化图表，创建城市宜居度地图，展示关键指标对比。**
5. **推荐数据来源网站：**
   1. **空气质量在线监测分析平台：**[**https://www.aqistudy.cn/**](https://www.aqistudy.cn/)
   2. **浙江统计局：<http://tjj.zj.gov.cn>**
   3. **国家地表水质自动监测实时数据发布系统：<https://szzdjc.cnemc.cn:8070/GJZ/Business/Publish/Main.html>**
   4. **浙江省住建厅：**[**https://jst.zj.gov.cn/**](https://jst.zj.gov.cn/)
   5. **国家气象科学数据中心：**[**http://data.cma.cn**](http://data.cma.cn)

**报告中需要详细介绍设计思路和步骤（截图），并且用数据可视化方式展示分析和展示最后的结论。将所有爬取的数据、数据分析代码、可视化图表文件一起打包为一个zip文件，文件名：小组名-Task3.zip，与报告一起上传至Moodle。**

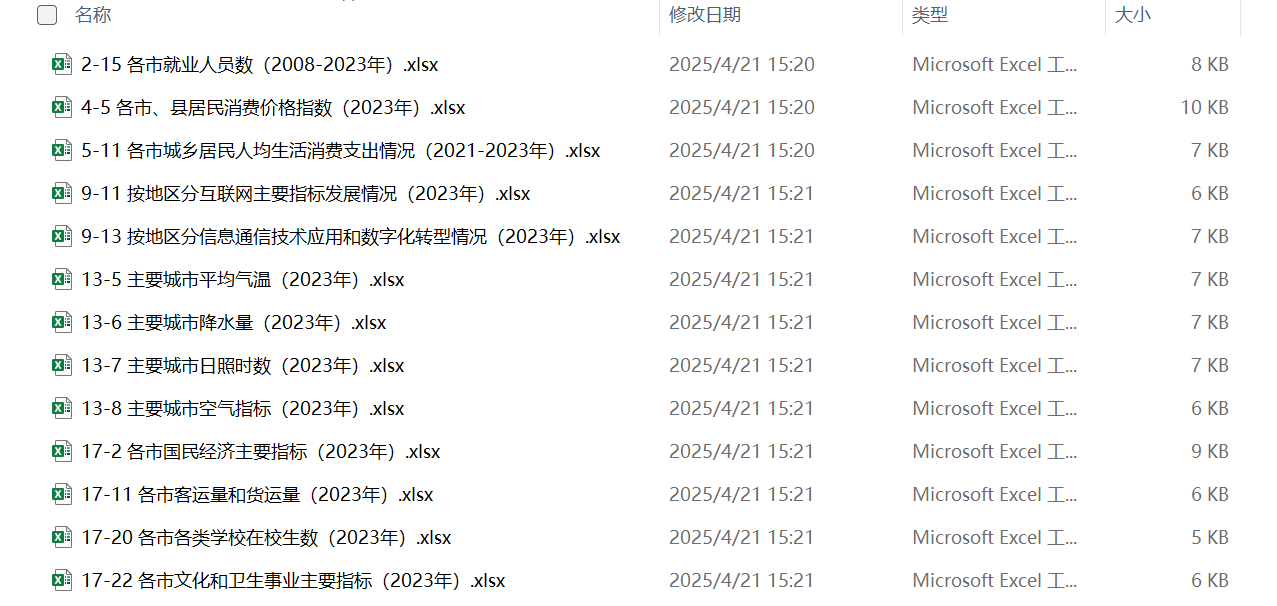
这里选择爬取https://tjj.zj.gov.cn/的数据

设计easyspider数据爬取脚本如下:

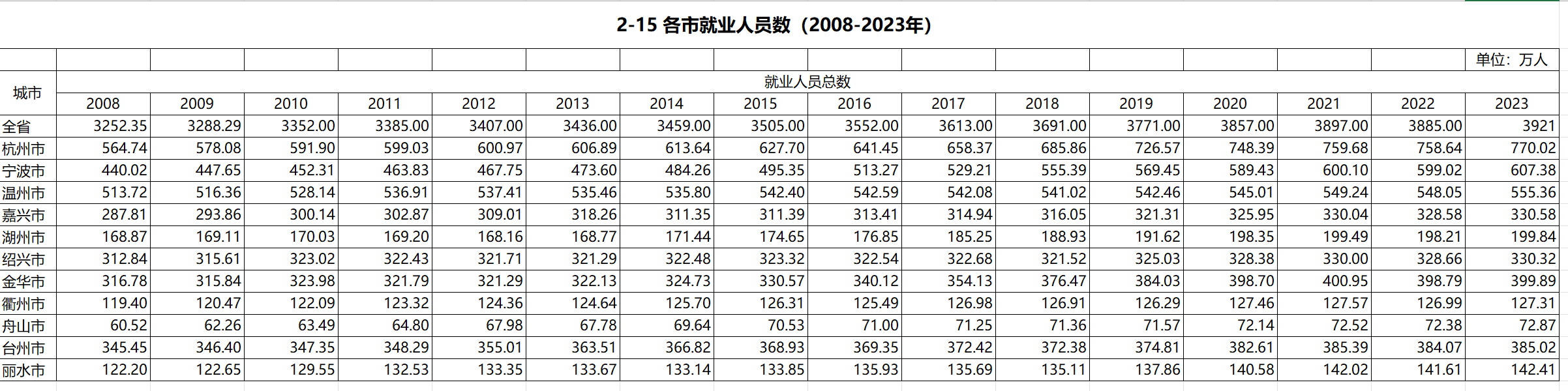


该脚本的逻辑为:首先打开浙江省统计局的网页，然后点击各个大标签，各个大标签下有许多个小标签，对于各个小标签，循环爬取数据即可。

最终爬取到的，这里接下来要用到的数据如下:



这里打开其中一个表格进行查看:



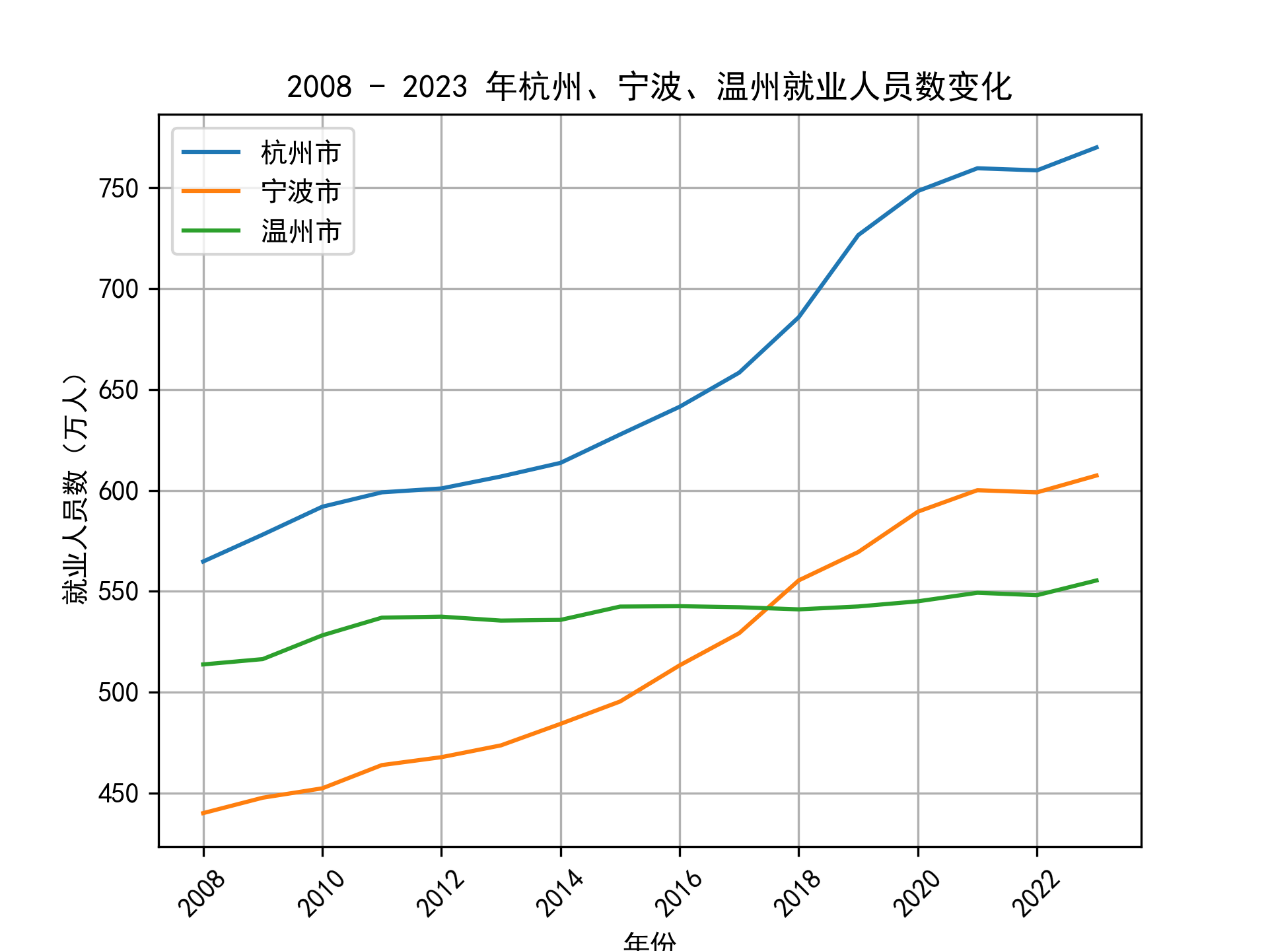
可以看到，除了杭州，宁波，温州三个城市，还有别的城市，所以接下来要写的python可视化代码在读取数据后，要进行数据预处理，只关注杭州，宁波，温州三个城市。

因为各个表格统计的数据不同，适合的可视化方法也不同，所以要对每个表格进行不同的可视化操作，以下是各个可视化表格的代码:

2-15的可视化代码:

*import* matplotlib.pyplot *as* plt  
*import* pandas *as* pd  
# 设置图片清晰度  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simhei']  
# 提取杭州、宁波、温州三个城市 2008 - 2023 年的就业人员数数据  
data = pd.DataFrame({  
 '年份': [2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023],  
 '杭州市': [564.74, 578.08, 591.9, 599.03, 600.97, 606.89, 613.64, 627.7, 641.45, 658.37, 685.86, 726.57, 748.39, 759.68, 758.64, 770.02],  
 '宁波市': [440.02, 447.65, 452.31, 463.83, 467.75, 473.6, 484.26, 495.35, 513.27, 529.21, 555.39, 569.45, 589.43, 600.1, 599.02, 607.38],  
 '温州市': [513.72, 516.36, 528.14, 536.91, 537.41, 535.46, 535.8, 542.4, 542.59, 542.08, 541.02, 542.46, 545.01, 549.24, 548.05, 555.36]  
})  
data.set\_index('年份', inplace=*True*)  
data.plot(title='2008 - 2023 年杭州、宁波、温州就业人员数变化')  
plt.xlabel('年份')  
plt.xticks(rotation=45)  
plt.ylabel('就业人员数（万人）')  
plt.legend()  
plt.grid(*True*)  
plt.show()

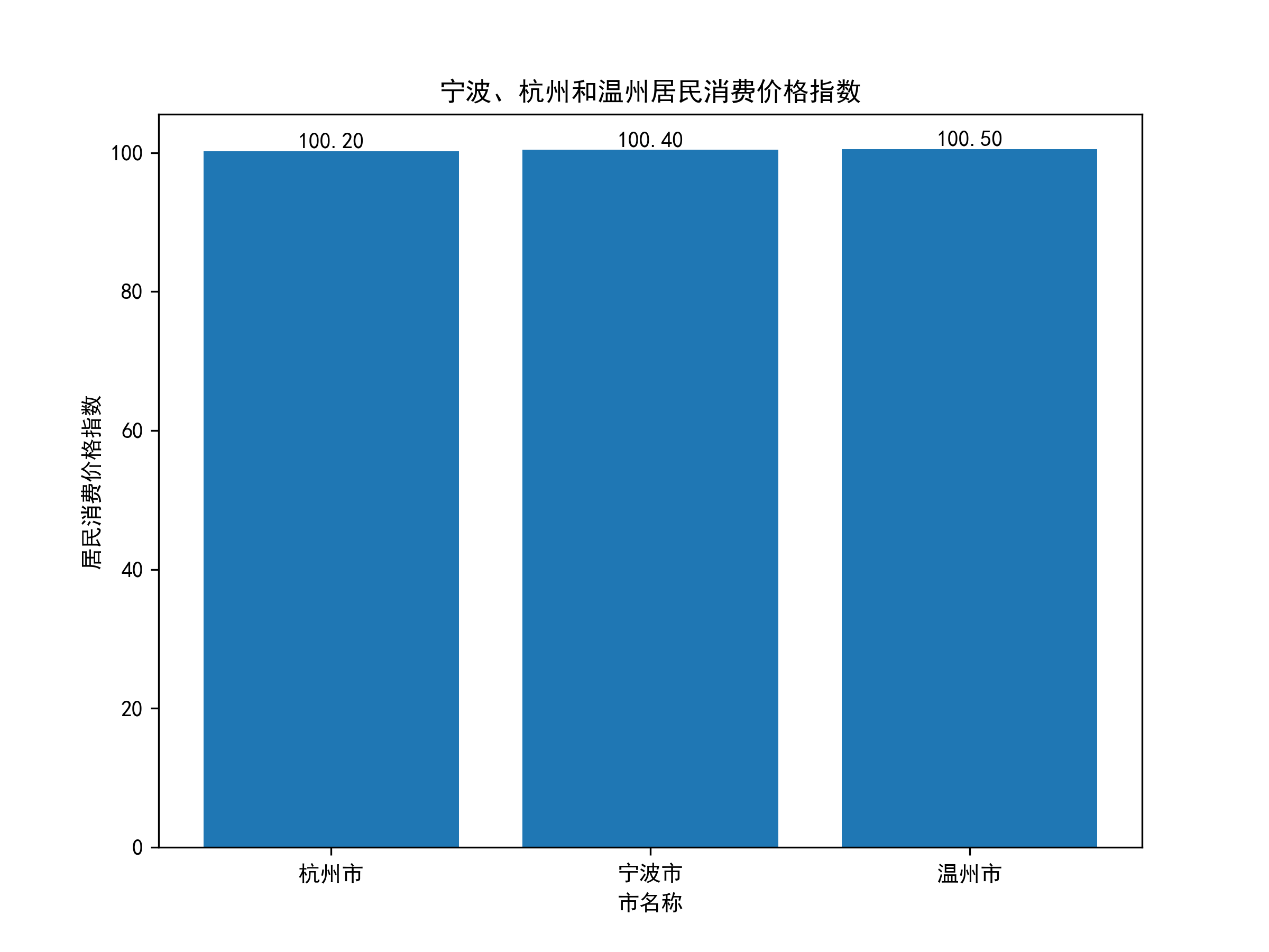
可视化的结果如下:



4-5的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
excel\_file = pd.ExcelFile('4-5 各市、县居民消费价格指数（2023年）.xlsx')  
# 获取所有表名  
sheet\_name = excel\_file.sheet\_names  
  
# 读取表格的数据  
df = excel\_file.parse('4-5 各市、县居民消费价格指数（2023年）')  
df.columns = ['市（县）名称', '居民消费价格指数', '食品烟酒', '食品烟酒\_食品', '食品烟酒\_食品\_粮食', '食品烟酒\_食品\_菜', '食品烟酒\_食品\_畜肉类', '食品烟酒\_食品\_禽肉类', '食品烟酒\_食品\_水产品', '食品烟酒\_食品\_蛋类', '食品烟酒\_食品\_奶类', '衣着', '居住', '生活用品及服务', '交通和通信', '教育文化和娱乐', '医疗保健', '其他用品和服务']  
df = df[4:]  
df = df.reset\_index(drop=*True*)  
df['居民消费价格指数'] = pd.to\_numeric(df['居民消费价格指数'])  
selected\_cities = ['杭州市', '宁波市', '温州市']  
selected\_df = df[df['市（县）名称'].isin(selected\_cities)]  
plt.figure(figsize=(8, 6))  
bars = plt.bar(selected\_df['市（县）名称'], selected\_df['居民消费价格指数'])  
*for* bar *in* bars:  
 height = bar.get\_height()  
 plt.text(bar.get\_x() + bar.get\_width() / 2., height,  
 '%.2f' % height,  
 ha='center', va='bottom')  
plt.title('宁波、杭州和温州居民消费价格指数')  
plt.xlabel('市名称')  
plt.ylabel('居民消费价格指数')  
plt.show()

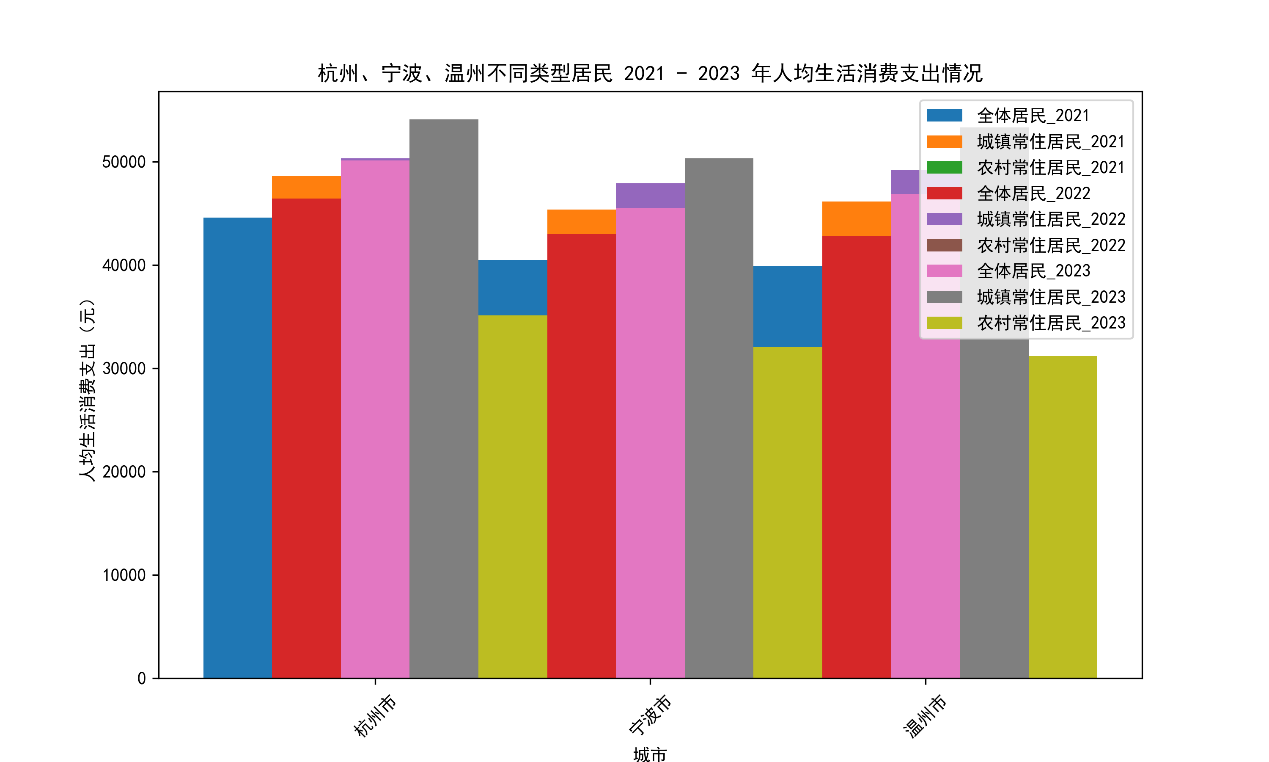
可视化的结果如下:



5-11的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
*import* numpy *as* np  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
excel\_file = pd.ExcelFile('5-11 各市城乡居民人均生活消费支出情况（2021-2023年）.xlsx')  
sheet\_name = excel\_file.sheet\_names  
df = excel\_file.parse(sheet\_name[0])  
df.columns = ['城市', '全体居民\_2021', '全体居民\_2022', '全体居民\_2023', '城镇常住居民\_2021', '城镇常住居民\_2022', '城镇常住居民\_2023', '农村常住居民\_2021', '农村常住居民\_2022', '农村常住居民\_2023']  
df = df[3:]  
df = df.reset\_index(drop=*True*)  
selected\_cities = ['杭州市', '宁波市', '温州市']  
df = df[df['城市'].isin(selected\_cities)]  
plt.figure(figsize=(10, 6))  
width = 0.25  
years = ['2021', '2022', '2023']  
num\_years = len(years)  
num\_cities = len(df['城市'])  
x = np.arange(num\_cities)  
*for* i, year *in* enumerate(years):  
 all\_residents = df[f'全体居民\_{year}']  
 urban\_residents = df[f'城镇常住居民\_{year}']  
 rural\_residents = df[f'农村常住居民\_{year}']  
 pos = x + (i - 1) \* width  
 plt.bar(pos, all\_residents, width, label=f'全体居民\_{year}')  
 plt.bar(pos + width, urban\_residents, width, label=f'城镇常住居民\_{year}')  
 plt.bar(pos + 2 \* width, rural\_residents, width, label=f'农村常住居民\_{year}')  
plt.xticks(x + width, df['城市'], rotation=45)  
plt.title('杭州、宁波、温州不同类型居民 2021 - 2023 年人均生活消费支出情况')  
plt.xlabel('城市')  
plt.ylabel('人均生活消费支出（元）')  
plt.legend()  
plt.show()

可视化的结果如下:



9-11的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
excel\_file = pd.ExcelFile('9-11 按地区分互联网主要指标发展情况（2023年）.xlsx')  
sheet\_name = excel\_file.sheet\_names  
df = excel\_file.parse(sheet\_name[0])  
df.columns = ['地区', '固定互联网宽带接入端口(万个)', '移动互联网用户(万户)', '移动互联网用户接入流量(万GB)', '(固定)互联网宽带接入用户(万户)']  
df = df[2:]  
df = df.reset\_index(drop=*True*)  
selected\_cities = ['杭州市', '宁波市', '温州市']  
selected\_df = df[df['地区'].isin(selected\_cities)]  
fig, axs = plt.subplots(2, 2, figsize=(15, 10))  
bars\_1 = axs[0, 0].bar(selected\_df['地区'], selected\_df['固定互联网宽带接入端口(万个)'])  
axs[0, 0].set\_title('固定互联网宽带接入端口(万个)')  
*for* bar *in* bars\_1:  
 height = bar.get\_height()  
 axs[0, 0].text(bar.get\_x() + bar.get\_width() / 2., height, f'{height:.1f}', ha='center', va='bottom')  
bars\_2 = axs[0, 1].bar(selected\_df['地区'], selected\_df['移动互联网用户(万户)'])  
axs[0, 1].set\_title('移动互联网用户(万户)')  
*for* bar *in* bars\_2:  
 height = bar.get\_height()  
 axs[0, 1].text(bar.get\_x() + bar.get\_width() / 2., height, f'{height:.1f}', ha='center', va='bottom')  
bars\_3 = axs[1, 0].bar(selected\_df['地区'], selected\_df['移动互联网用户接入流量(万GB)'])  
axs[1, 0].set\_title('移动互联网用户接入流量(万GB)')  
*for* bar *in* bars\_3:  
 height = bar.get\_height()  
 axs[1, 0].text(bar.get\_x() + bar.get\_width() / 2., height, f'{height:.1f}', ha='center', va='bottom')  
bars\_4 = axs[1, 1].bar(selected\_df['地区'], selected\_df['(固定)互联网宽带接入用户(万户)'])  
axs[1, 1].set\_title('(固定)互联网宽带接入用户(万户)')  
*for* bar *in* bars\_4:  
 height = bar.get\_height()  
 axs[1, 1].text(bar.get\_x() + bar.get\_width() / 2., height, f'{height:.1f}', ha='center', va='bottom')  
plt.tight\_layout()  
plt.show()

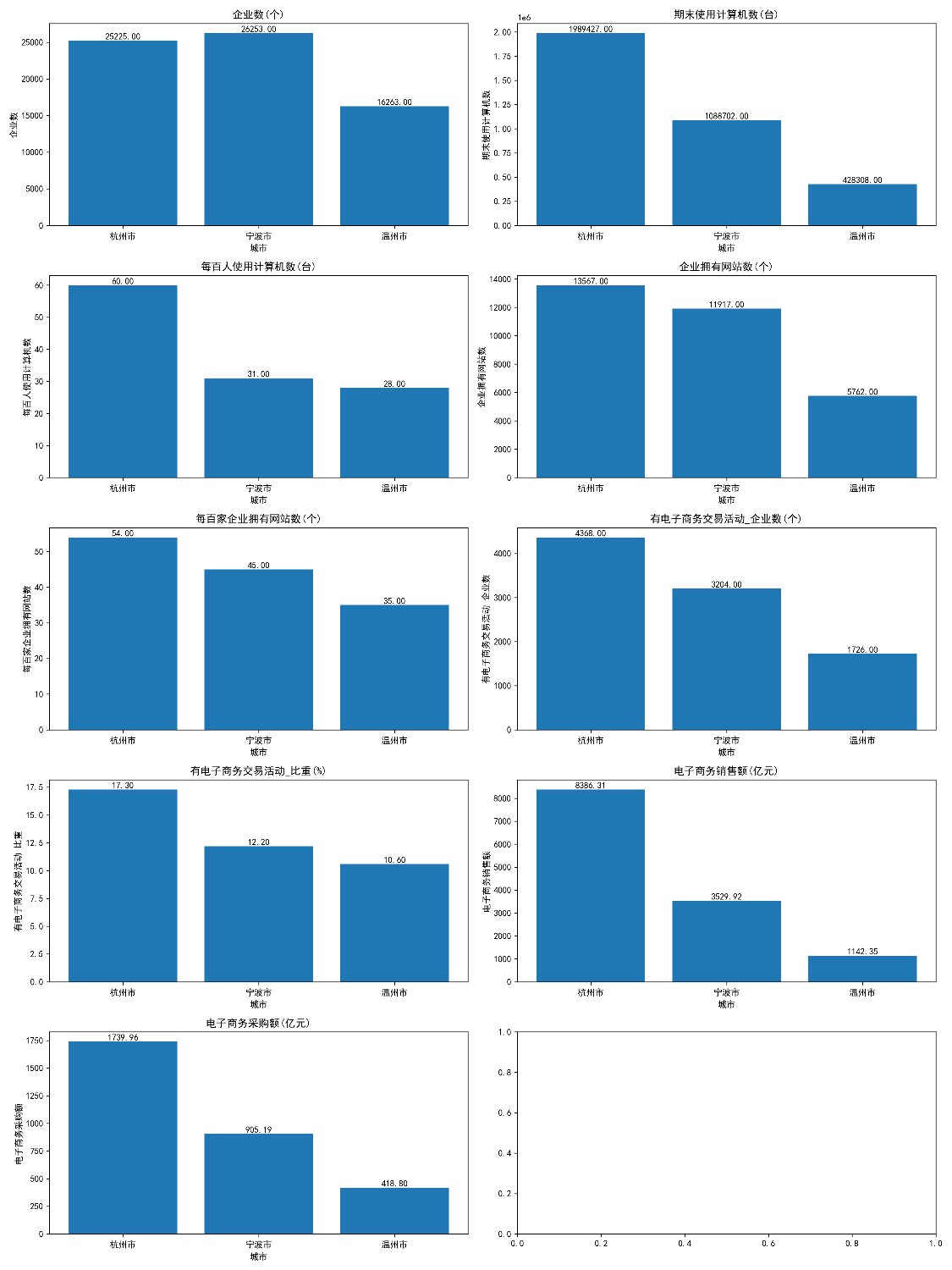
可视化的结果如下:



9-13的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
excel\_file = pd.ExcelFile('9-13 按地区分信息通信技术应用和数字化转型情况（2023年）.xlsx')  
sheet\_name = excel\_file.sheet\_names  
df = excel\_file.parse(sheet\_name[0])  
df.columns = ['行业', '企业数(个)', '期末使用计算机数(台)', '每百人使用计算机数(台)', '企业拥有网站数(个)', '每百家企业拥有网站数(个)', '有电子商务交易活动\_企业数(个)', '有电子商务交易活动\_比重(%)', '电子商务销售额(亿元)', '电子商务采购额(亿元)']  
df = df[4:14]  
df = df.reset\_index(drop=*True*)  
selected\_cities = ['杭州市', '宁波市', '温州市']  
selected\_df = df[df['行业'].isin(selected\_cities)]  
fig, axes = plt.subplots(5, 2, figsize=(15, 20))  
metrics = ['企业数(个)', '期末使用计算机数(台)', '每百人使用计算机数(台)', '企业拥有网站数(个)', '每百家企业拥有网站数(个)', '有电子商务交易活动\_企业数(个)', '有电子商务交易活动\_比重(%)', '电子商务销售额(亿元)', '电子商务采购额(亿元)']  
*for* i, metric *in* enumerate(metrics):  
 row = i // 2  
 col = i % 2  
 axes[row, col].bar(selected\_df['行业'], selected\_df[metric])  
 axes[row, col].set\_title(metric)  
 axes[row, col].set\_xlabel('城市')  
 axes[row, col].set\_ylabel(metric.split('(')[0])  
 *for* bar *in* axes[row, col].patches:  
 height = bar.get\_height()  
 axes[row, col].text(bar.get\_x() + bar.get\_width() / 2, height, f'{height:.2f}', ha='center', va='bottom')  
plt.tight\_layout()  
plt.show()

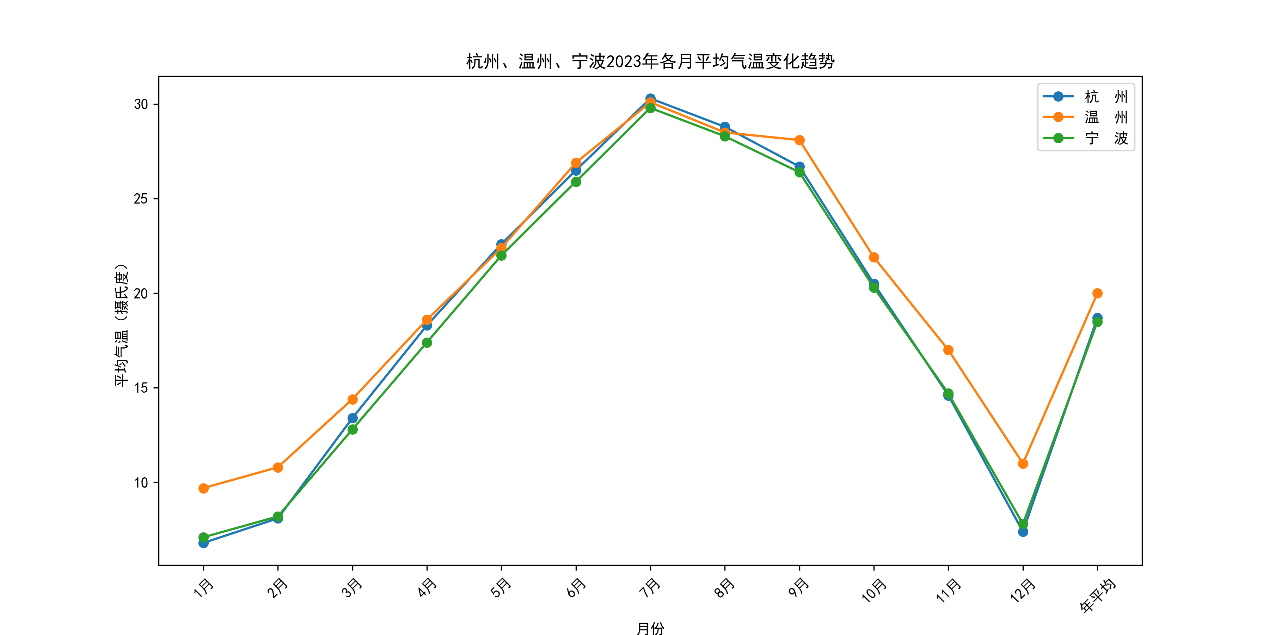
可视化的结果如下:



13-5的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('13-5 主要城市平均气温（2023年）.xlsx')  
df = excel\_file.parse('13-5 主要城市平均气温（2023年）', header=2)  
selected\_cities = ['杭 州', '温 州', '宁 波']  
filtered\_df = df[df['城市'].isin(selected\_cities)]  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
melted\_df = pd.melt(filtered\_df, id\_vars='城市', var\_name='月份', value\_name='平均气温')  
plt.figure(figsize=(12, 6))  
*for* city *in* selected\_cities:  
 city\_data = melted\_df[melted\_df['城市'] == city]  
 plt.plot(city\_data['月份'], city\_data['平均气温'], marker='o', label=city)  
plt.title('杭州、温州、宁波2023年各月平均气温变化趋势')  
plt.xlabel('月份')  
plt.ylabel('平均气温（摄氏度）')  
plt.xticks(rotation=45)  
plt.legend()  
plt.show()

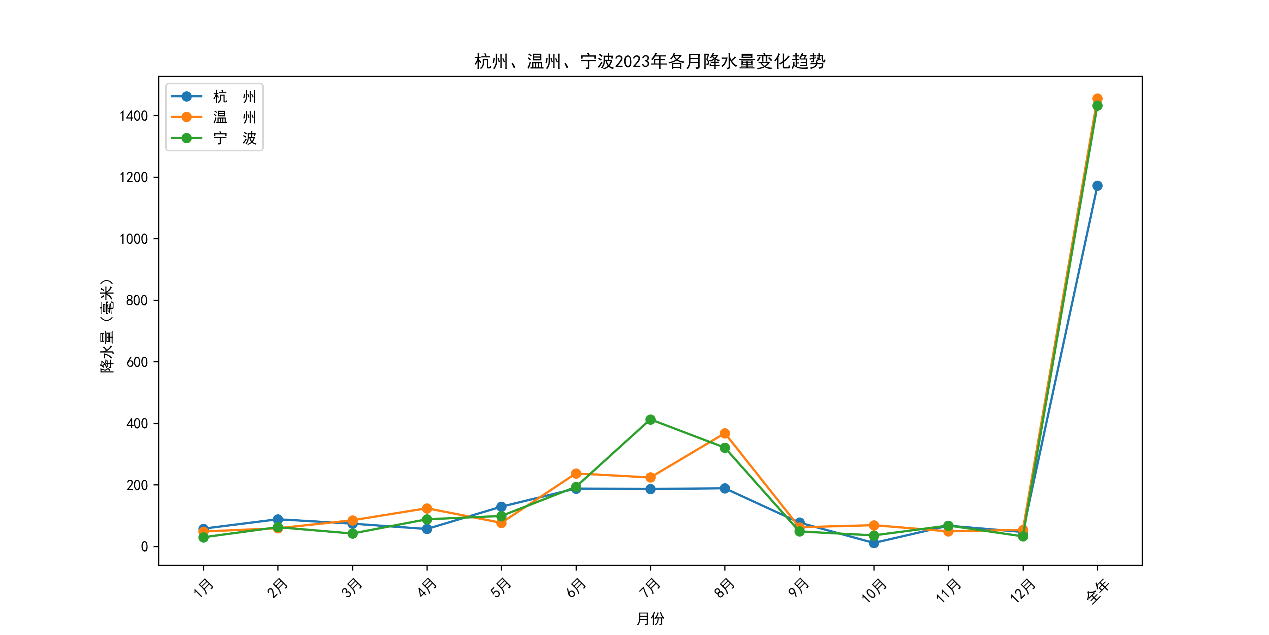
可视化的结果如下:



13-6的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
excel\_file = pd.ExcelFile('13-6 主要城市降水量（2023年）.xlsx')  
df = excel\_file.parse('13-6 主要城市降水量（2023年）', header=2)  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
selected\_cities = ['杭 州', '温 州', '宁 波']  
filtered\_df = df[df['城市'].isin(selected\_cities)]  
melted\_df = pd.melt(filtered\_df, id\_vars='城市', var\_name='月份', value\_name='降水量')  
plt.figure(figsize=(12, 6))  
*for* city *in* selected\_cities:  
 city\_data = melted\_df[melted\_df['城市'] == city]  
 plt.plot(city\_data['月份'], city\_data['降水量'], marker='o', label=city)  
plt.title('杭州、温州、宁波2023年各月降水量变化趋势')  
plt.xlabel('月份')  
plt.ylabel('降水量（毫米）')  
plt.xticks(rotation=45)  
plt.legend()  
plt.show()

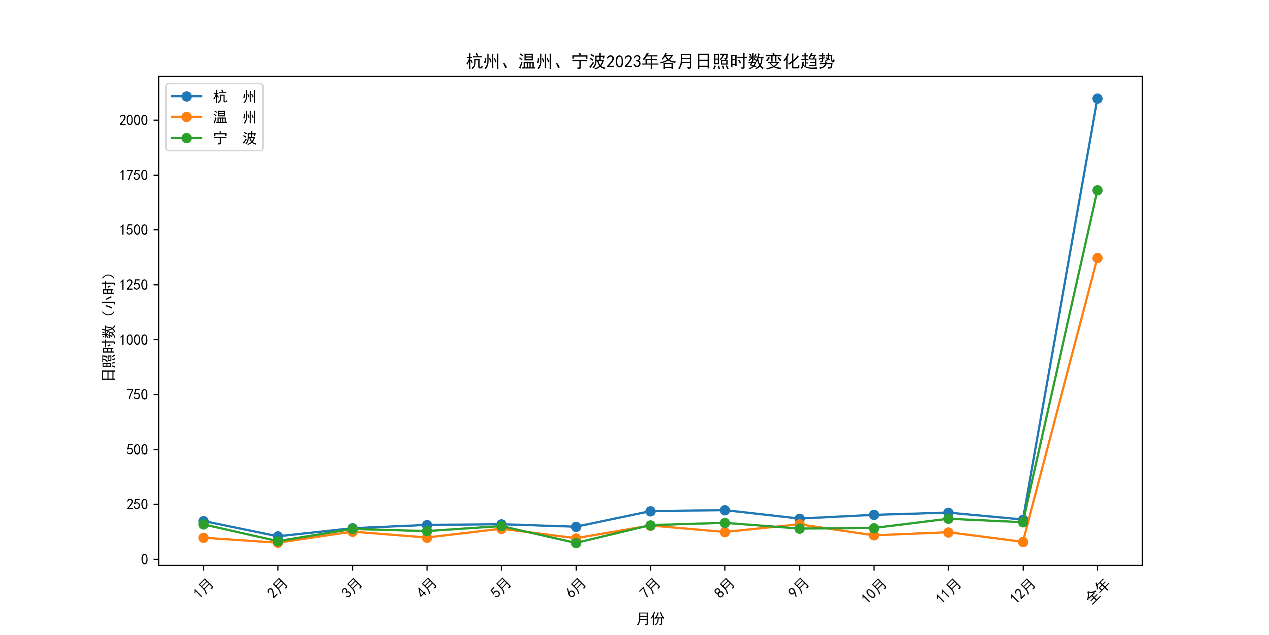
可视化的结果如下:



13-7的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('13-7 主要城市日照时数（2023年）.xlsx')  
df = excel\_file.parse('13-7 主要城市日照时数（2023年）', header=2)  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
selected\_cities = ['杭 州', '温 州', '宁 波']  
filtered\_df = df[df['城市'].isin(selected\_cities)]  
melted\_df = pd.melt(filtered\_df, id\_vars='城市', var\_name='月份', value\_name='日照时数')  
plt.figure(figsize=(12, 6))  
*for* city *in* selected\_cities:  
 city\_data = melted\_df[melted\_df['城市'] == city]  
 plt.plot(city\_data['月份'], city\_data['日照时数'], marker='o', label=city)  
plt.title('杭州、温州、宁波2023年各月日照时数变化趋势')  
plt.xlabel('月份')  
plt.ylabel('日照时数（小时）')  
plt.xticks(rotation=45)  
plt.legend()  
plt.show()

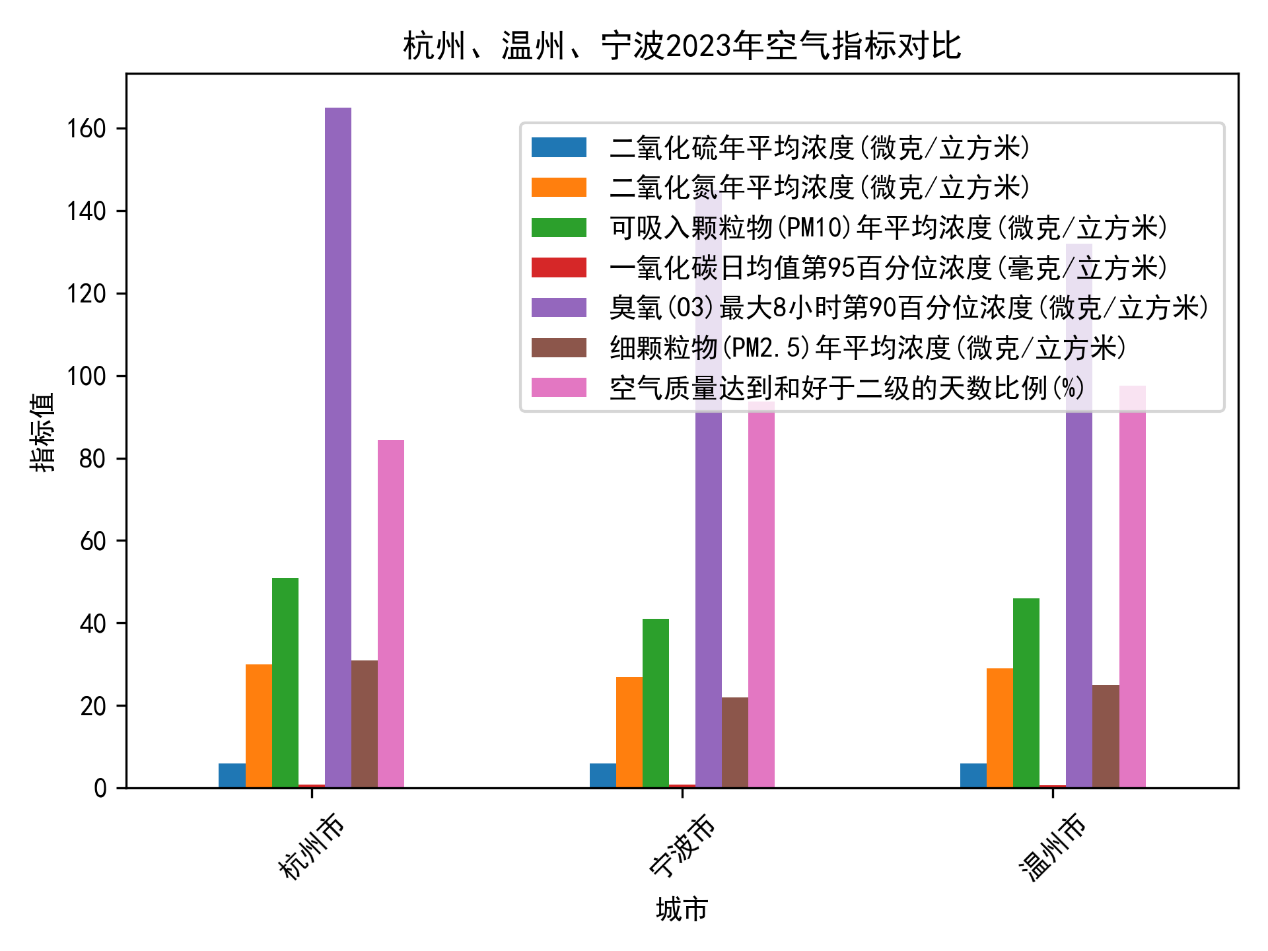
可视化的结果如下:



13-8的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('13-8 主要城市空气指标（2023年）.xlsx')  
df = excel\_file.parse('13-8 主要城市空气指标（2023年）', header=1)  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
selected\_df = df.loc[1:3]  
selected\_df.set\_index('城市', inplace=*True*)  
plt.figure(figsize=(15, 8))  
ax = selected\_df.plot(kind='bar')  
plt.title('杭州、温州、宁波2023年空气指标对比')  
plt.xlabel('城市')  
plt.ylabel('指标值')  
plt.xticks(rotation=45)  
ax.legend(bbox\_to\_anchor=(1, 0.95), loc='upper right')  
plt.tight\_layout()  
plt.show()

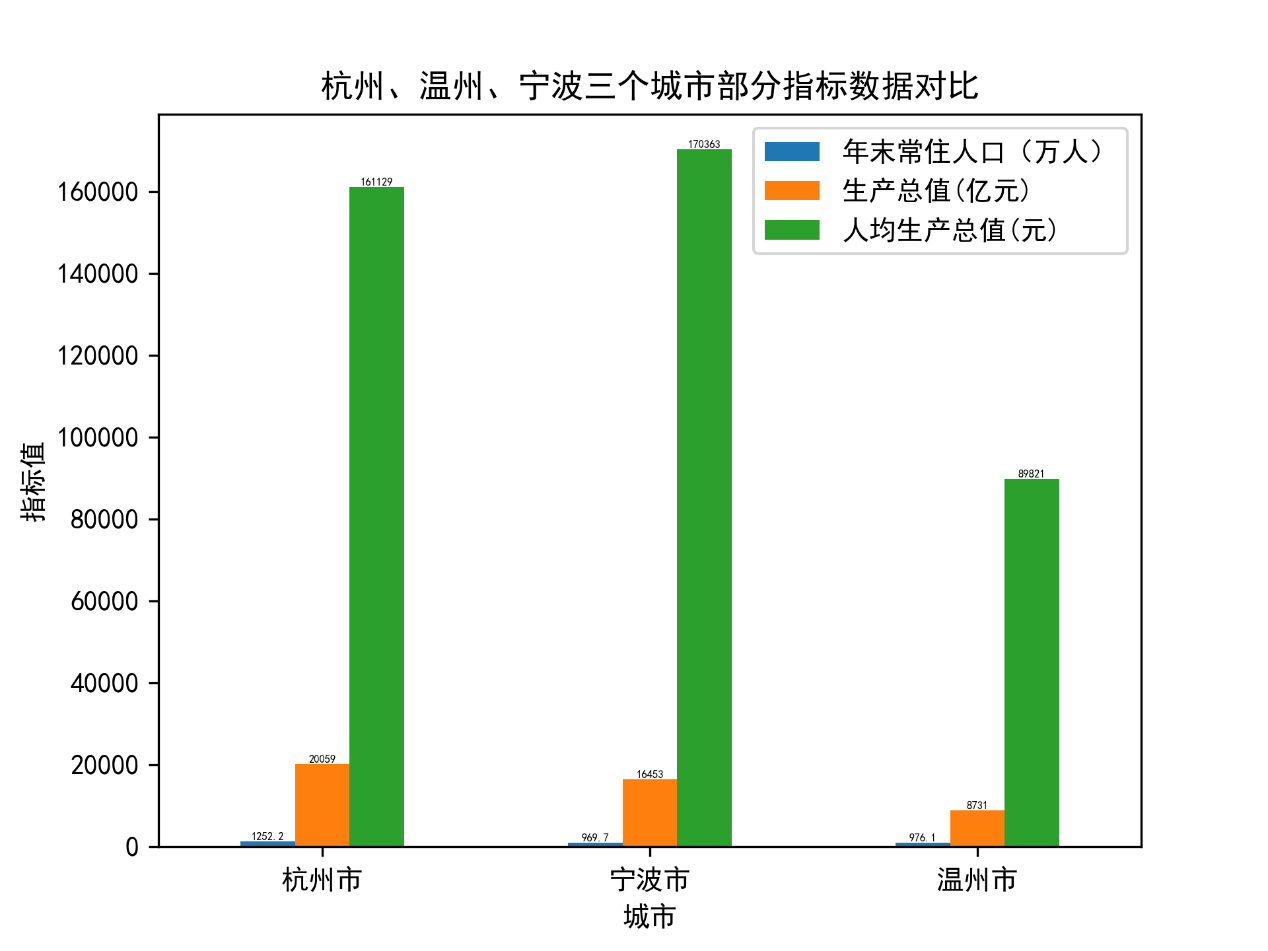
可视化的结果如下:



17-2的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('17-2 各市国民经济主要指标（2023年）.xlsx')  
df = excel\_file.parse('17-2 各市国民经济主要指标（2023年）')  
df.columns = ['城市', '年末常住人口（万人）', '生产总值(亿元)', '生产总值(亿元)\_第一产业', '生产总值(亿元)\_第二产业', '生产总值(亿元)\_第三产业', '生产总值(亿元)\_工业', '人均生产总值(元)', '社会消费品零售总额 (亿元)', '进口总额 (亿元)', '出口总额 (亿元)', '财政总收入 (亿元)', '一般公共预算收入 （亿元', '一般公共预算支出 （亿元', '住户存款年末余额 (亿元)', '城镇居民人均可支配收入 (元)', '农村居民人均可支配收入 (元)']  
df = df[2:]  
df = df.reset\_index(drop=*True*)  
filter\_df = df[df['城市'].isin(['杭州市', '温州市', '宁波市'])]  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
bar = filter\_df.set\_index('城市')[['年末常住人口（万人）', '生产总值(亿元)', '人均生产总值(元)']].plot(kind='bar')  
*for* container *in* bar.containers:  
 *for* index, rect *in* enumerate(container):  
 x\_coord = rect.get\_x() + rect.get\_width() / 2  
 y\_coord = rect.get\_height()  
 week\_day = filter\_df['城市'].iloc[index]  
 time\_period = bar.get\_xticklabels()[index].get\_text()  
 plt.text(x\_coord, y\_coord, str(y\_coord), ha='center', va='bottom', fontsize=4)  
plt.xticks(rotation=360)  
plt.title('杭州、温州、宁波三个城市部分指标数据对比')  
plt.xlabel('城市')  
plt.ylabel('指标值')  
plt.show()

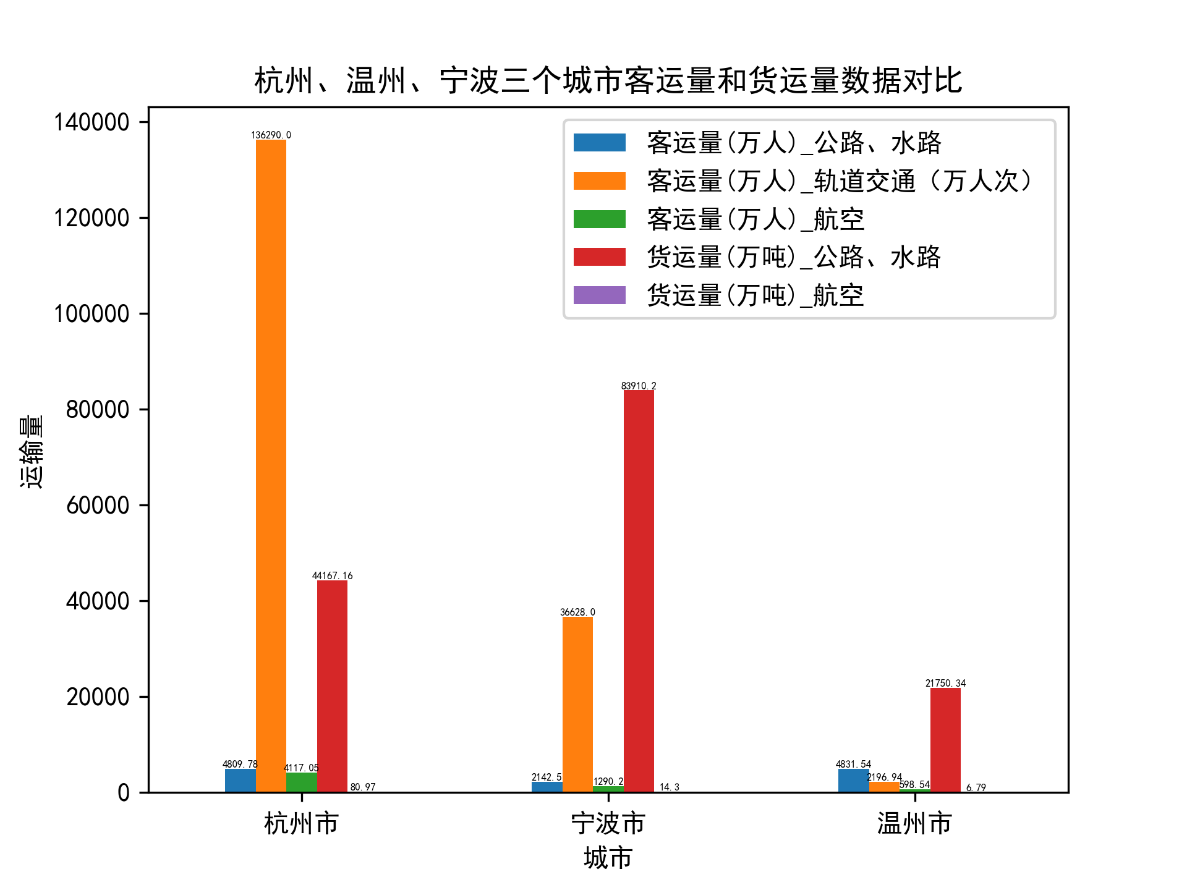
可视化的结果如下:



17-11的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('17-11 各市客运量和货运量（2023年）.xlsx')  
df = excel\_file.parse('17-11 各市客运量和货运量（2023年）')  
df.columns = ['城市', '客运量(万人)\_公路、水路', '客运量(万人)\_轨道交通（万人次）', '客运量(万人)\_航空', '货运量(万吨)\_公路、水路', '货运量(万吨)\_航空']  
df = df[2:]  
df = df.reset\_index(drop=*True*)  
filter\_df = df[df['城市'].isin(['杭州市', '温州市', '宁波市'])]  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
ax = filter\_df.set\_index('城市').plot(kind='bar')  
*for* container *in* ax.containers:  
 *for* index, rect *in* enumerate(container):  
 x\_coord = rect.get\_x() + rect.get\_width() / 2  
 y\_coord = rect.get\_height()  
 plt.text(x\_coord, y\_coord, str(y\_coord), ha='center', va='bottom', fontsize=4)  
plt.xticks(rotation=360)  
plt.title('杭州、温州、宁波三个城市客运量和货运量数据对比')  
plt.xlabel('城市')  
plt.ylabel('运输量')  
plt.show()

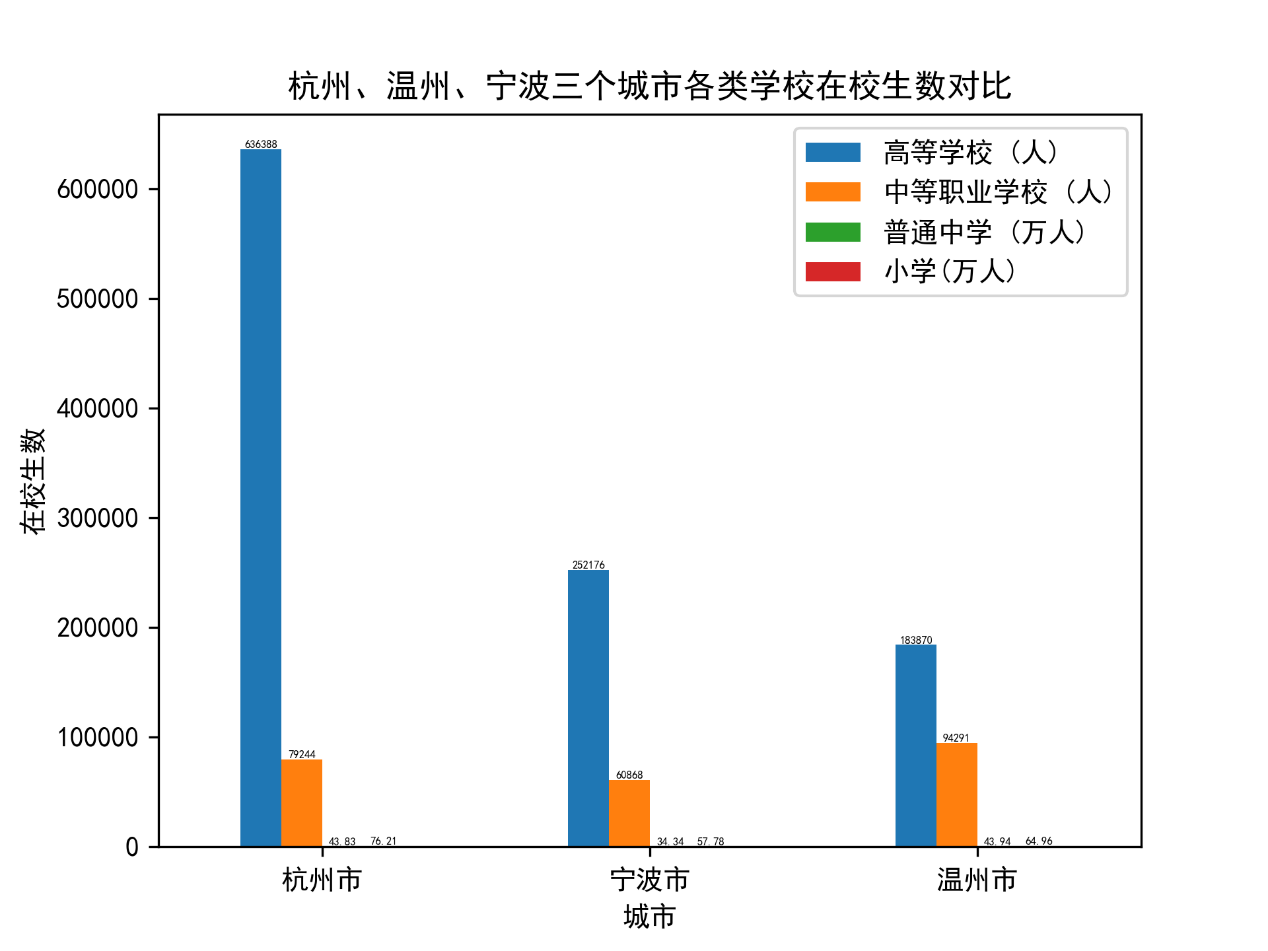
可视化的结果如下:



17-20的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('17-20 各市各类学校在校生数（2023年）.xlsx')  
df = excel\_file.parse('17-20 各市各类学校在校生数（2023年）')  
filter\_df = df[df['17-20 各市各类学校在校生数（2023年）'].isin(['杭州市', '温州市', '宁波市'])]  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
filter\_df.columns = ['城市', '高等学校 (人)', '中等职业学校 (人)', '普通中学 (万人)', '小学(万人)']  
ax = filter\_df.set\_index('城市').plot(kind='bar')  
*for* container *in* ax.containers:  
 *for* index, rect *in* enumerate(container):  
 x\_coord = rect.get\_x() + rect.get\_width() / 2  
 y\_coord = rect.get\_height()  
 plt.text(x\_coord, y\_coord, str(y\_coord), ha='center', va='bottom', fontsize=4)  
plt.xticks(rotation=360)  
plt.title('杭州、温州、宁波三个城市各类学校在校生数对比')  
plt.xlabel('城市')  
plt.ylabel('在校生数')  
plt.show()

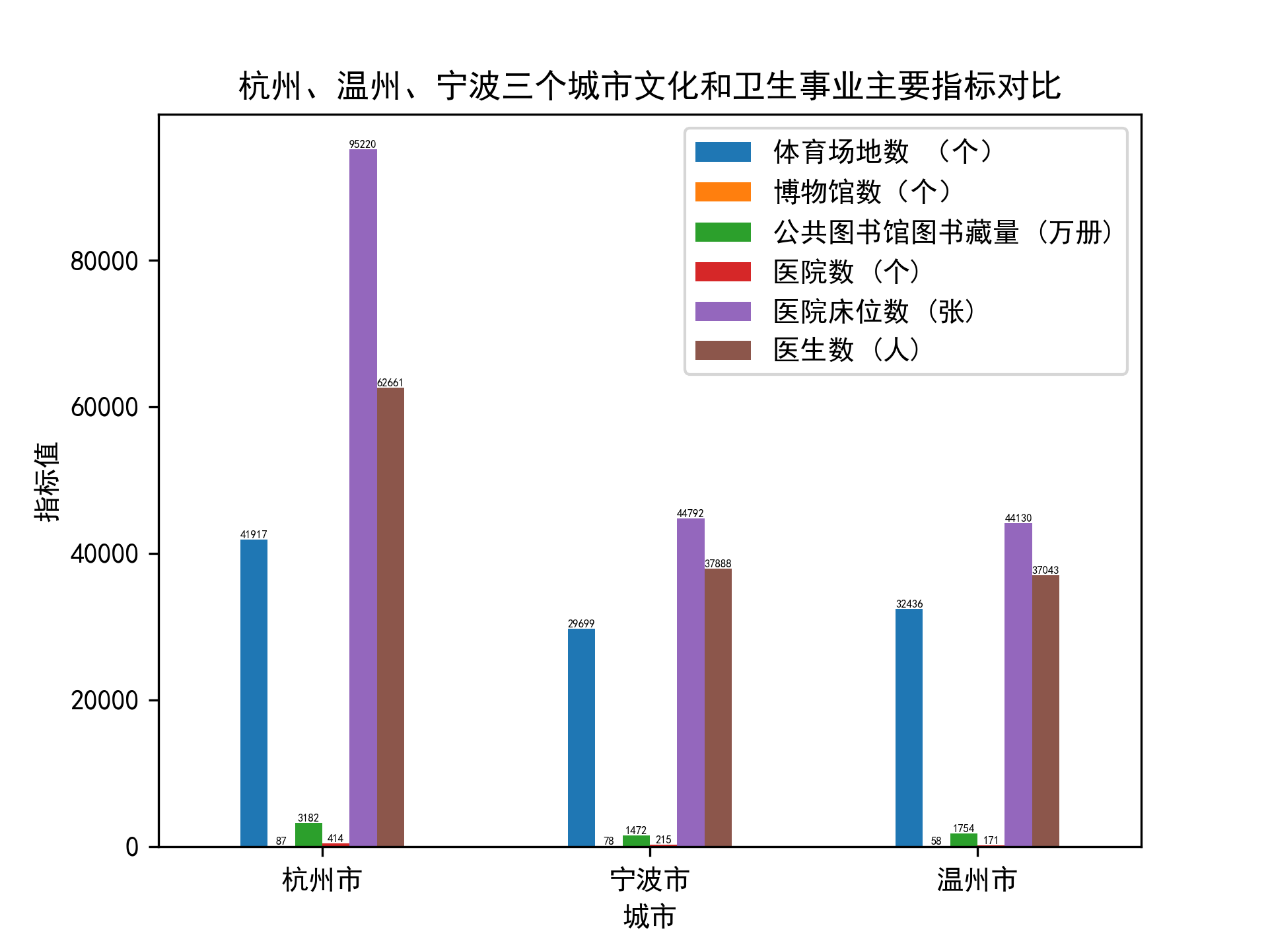
可视化的结果如下:



17-22的可视化代码:

*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
  
excel\_file = pd.ExcelFile('17-22 各市文化和卫生事业主要指标（2023年）.xlsx')  
df = excel\_file.parse('17-22 各市文化和卫生事业主要指标（2023年）')  
filter\_df = df[df['17-22 各市文化和卫生事业主要指标（2023年）'].isin(['杭州市', '温州市', '宁波市'])]  
plt.rcParams['figure.dpi'] = 300  
plt.rcParams['font.sans-serif'] = ['simHei']  
filter\_df.columns = ['城市', '体育场地数 （个）', '博物馆数（个）', '公共图书馆图书藏量 (万册)', '医院数 (个)', '医院床位数 (张)', '医生数 (人)']  
ax = filter\_df.set\_index('城市').plot(kind='bar')  
*for* container *in* ax.containers:  
 *for* index, rect *in* enumerate(container):  
 x\_coord = rect.get\_x() + rect.get\_width() / 2  
 y\_coord = rect.get\_height()  
 plt.text(x\_coord, y\_coord, str(y\_coord), ha='center', va='bottom', fontsize=4)  
plt.xticks(rotation=360)  
plt.title('杭州、温州、宁波三个城市文化和卫生事业主要指标对比')  
plt.xlabel('城市')  
plt.ylabel('指标值')  
plt.show()

可视化的结果如下:

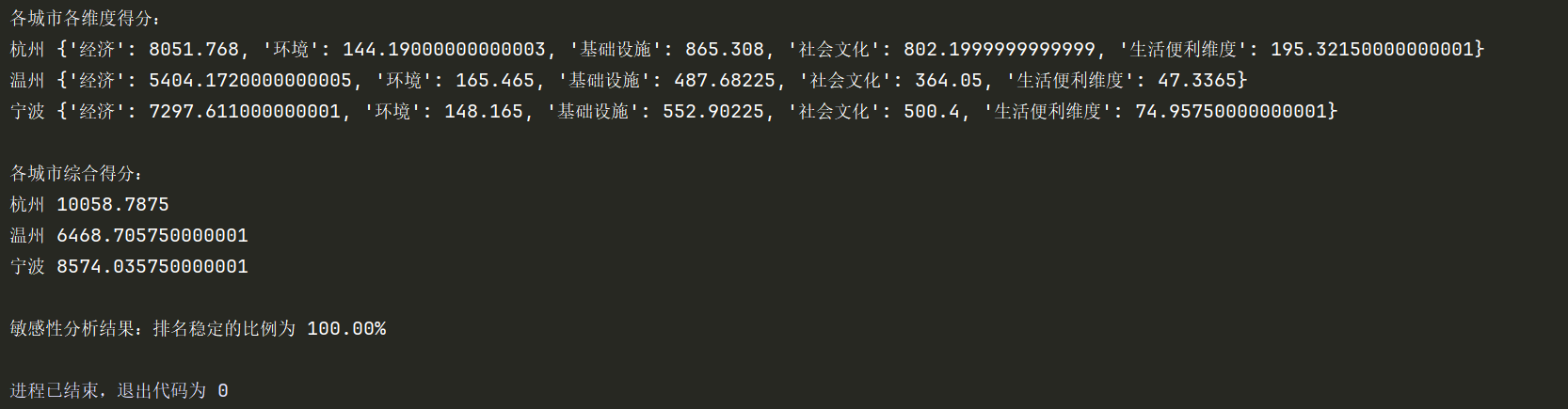


然后是设计合理的权重分配方案，应用适当的分析模型，进行敏感性分析验证结果稳健性

这里直接使用部分我爬取的数据，具体实现的代码如下:

*import* numpy *as* np  
*from* scipy.stats *import* spearmanr  
  
# 提取杭州、温州、宁波的数据，并按维度整理  
data = {  
 '经济': {  
 '杭州': {  
 '就业人员': 770.02, # 2023年数据  
 '消费价格指数': 100.2,  
 '人均生活消费支出': 58879, # 2023年城镇居民数据  
 '国民经济主要指标': 20768.46 # 2023年GDP数据  
 },  
 '温州': {  
 '就业人员': 555.36,  
 '消费价格指数': 100.5,  
 '人均生活消费支出': 45356,  
 '国民经济主要指标': 8029.86  
 },  
 '宁波': {  
 '就业人员': 607.38,  
 '消费价格指数': 100.4,  
 '人均生活消费支出': 56564,  
 '国民经济主要指标': 15704.33  
 }  
 },  
 '环境': {  
 '杭州': {  
 '平均气温': 18.7,  
 '降水量': 1335.4,  
 '日照时数': 1440.0,  
 '空气指标': 89.7  
 },  
 '温州': {  
 '平均气温': 19.2,  
 '降水量': 1736.2,  
 '日照时数': 1461.6,  
 '空气指标': 92.3  
 },  
 '宁波': {  
 '平均气温': 18.5,  
 '降水量': 1313.0,  
 '日照时数': 1541.6,  
 '空气指标': 90.2  
 }  
 },  
 '基础设施': {  
 '杭州': {  
 '学校在校生': 217.44,  
 '客运量': 11320  
 },  
 '温州': {  
 '学校在校生': 166.43,  
 '客运量': 6336  
 },  
 '宁波': {  
 '学校在校生': 137.03,  
 '客运量': 7235  
 }  
 },  
 '社会文化': {  
 '杭州': {  
 '文化和卫生事业主要指标': 5348  
 },  
 '温州': {  
 '文化和卫生事业主要指标': 2427  
 },  
 '宁波': {  
 '文化和卫生事业主要指标': 3336  
 }  
 },  
 '生活便利维度': {  
 '杭州': {  
 '互联网主要指标': 2222.22,  
 '信息通信技术': 1684.21  
 },  
 '温州': {  
 '互联网主要指标': 504.62,  
 '信息通信技术': 442.11  
 },  
 '宁波': {  
 '互联网主要指标': 867.57,  
 '信息通信技术': 631.58  
 }  
 }  
}  
# 设计权重分配方案（这里简单假设各维度内指标等权重，各维度权重根据经验设定）  
weights = {  
 '经济': 0.4,  
 '环境': 0.2,  
 '基础设施': 0.15,  
 '社会文化': 0.15,  
 '生活便利维度': 0.1  
}  
# 计算各城市各维度得分  
city\_scores = {}  
*for* city *in* ['杭州', '温州', '宁波']:  
 city\_scores[city] = {}  
 *for* dimension *in* weights.keys():  
 num\_metrics = len(data[dimension][city])  
 dimension\_weight = weights[dimension]  
 metric\_weight = dimension\_weight / num\_metrics  
 dimension\_score = 0  
 *for* metric *in* data[dimension][city].keys():  
 dimension\_score += metric\_weight \* data[dimension][city][metric]  
 city\_scores[city][dimension] = dimension\_score  
# 计算各城市综合得分  
city\_overall\_scores = {}  
*for* city *in* city\_scores.keys():  
 overall\_score = 0  
 *for* dimension *in* city\_scores[city].keys():  
 overall\_score += city\_scores[city][dimension]  
 city\_overall\_scores[city] = overall\_score  
print("各城市各维度得分：")  
*for* city *in* city\_scores.keys():  
 print(city, city\_scores[city])  
print("\n各城市综合得分：")  
*for* city *in* city\_overall\_scores.keys():  
 print(city, city\_overall\_scores[city])  
# 进行敏感性分析（改变权重，看综合得分排名是否稳定）  
num\_trials = 100 # 试验次数  
stability\_count = 0 # 排名稳定的次数  
original\_ranking = np.argsort(list(city\_overall\_scores.values()))[::-1]  
*for* \_ *in* range(num\_trials):  
 # 随机生成权重（各维度权重在±20%范围内变动）  
 new\_weights = {}  
 *for* dimension *in* weights.keys():  
 new\_weight = weights[dimension] \* (1 + np.random.uniform(-0.2, 0.2))  
 new\_weights[dimension] = new\_weight  
 total\_weight = sum(new\_weights.values())  
 *for* dimension *in* new\_weights.keys():  
 new\_weights[dimension] = new\_weights[dimension] / total\_weight  
 # 重新计算各城市综合得分  
 new\_city\_overall\_scores = {}  
 *for* city *in* city\_scores.keys():  
 new\_overall\_score = 0  
 *for* dimension *in* city\_scores[city].keys():  
 new\_overall\_score += new\_weights[dimension] \* city\_scores[city][dimension]  
 new\_city\_overall\_scores[city] = new\_overall\_score  
 new\_ranking = np.argsort(list(new\_city\_overall\_scores.values()))[::-1]  
 # 计算斯皮尔曼等级相关系数  
 rho, p\_value = spearmanr(original\_ranking, new\_ranking)  
 *if* rho > 0.8 *and* p\_value < 0.05: # 假设相关系数大于0.8且p值小于0.05认为排名稳定  
 stability\_count += 1  
stability\_ratio = stability\_count / num\_trials  
print(f"\n敏感性分析结果：排名稳定的比例为 {stability\_ratio \* 100:.2f}%")

代码运行后的结果如下所示：



然后是最后的可视化呈现,制作交互式可视化图表，创建城市宜居度地图，展示关键指标对比

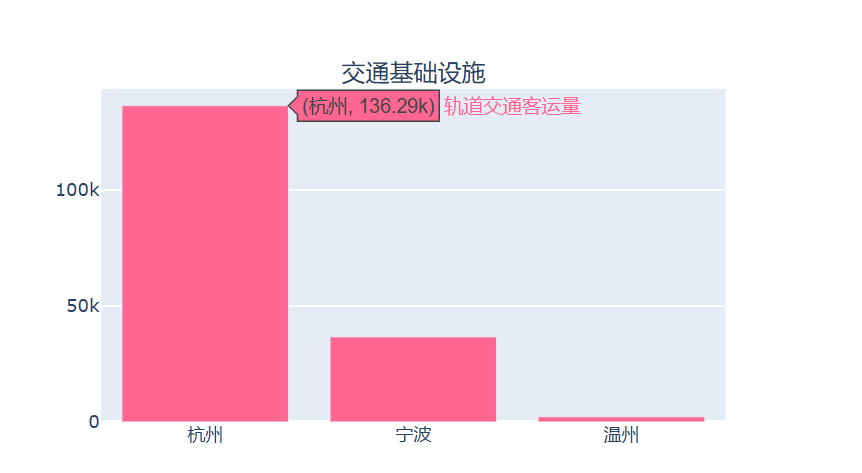
用python读取每个表的数据，读取其中一些有用的数据，然后利用plotly库来制作简单的可交互式的表格，以此来展示各项关键指标的对比，具体实现的代码如下所示:

*import* pandas *as* pd  
*import* plotly.express *as* px  
*import* plotly.subplots *as* sp  
*from* plotly.subplots *import* make\_subplots  
*import* plotly.graph\_objects *as* go  
  
# 读取所有数据表（替换为实际文件路径）  
data\_path = {  
 '经济': '17-2 各市国民经济主要指标（2023年）.xlsx',  
 '环境\_气温': '13-5 主要城市平均气温（2023年）.xlsx',  
 '环境\_降水': '13-6 主要城市降水量（2023年）.xlsx',  
 '环境\_空气': '13-8 主要城市空气指标（2023年）.xlsx',  
 '基础设施\_教育': '17-20 各市各类学校在校生数（2023年）.xlsx',  
 '基础设施\_交通': '17-11 各市客运量和货运量（2023年）.xlsx',  
 '社会文化': '17-22 各市文化和卫生事业主要指标（2023年）.xlsx',  
 '生活便利\_互联网': '9-11 按地区分互联网主要指标发展情况（2023年）.xlsx',  
 '生活便利\_ICT': '9-13 按地区分信息通信技术应用和数字化转型情况（2023年）.xlsx'  
}  
  
  
# 数据预处理函数  
*def* load\_data(*file*, *sheet*=0, *skiprows*=2, *usecols*=*None*):  
 # 动态跳过表头行，确保列名正确读取  
 df = pd.read\_excel(*file*, sheet\_name=*sheet*, skiprows=*skiprows*, usecols=*usecols*, header=0)  
  
 # 统一处理列名：移除空格和特殊字符  
 df.columns = df.columns.str.replace(r'[\s　]+', '', regex=*True*)  
 *if* '城市' *not in* df.columns:  
 # 尝试模糊匹配列名  
 possible\_columns = [col *for* col *in* df.columns *if* '市' *in* col]  
 *if* possible\_columns:  
 df = df.rename(columns={possible\_columns[0]: '城市'})  
 *else*:  
 *raise* KeyError(f"文件 {*file*} 中未找到有效的城市列")  
  
 # 统一城市名称格式：移除所有空格并去掉末尾的"市"  
 df['城市'] = df['城市'].str.replace(r'[\s　]+', '', regex=*True*).str.rstrip('市')  
 allowed\_cities = ['杭州', '宁波', '温州']  
 df = df[df['城市'].isin(allowed\_cities)]  
 *return* df  
  
  
# --------------------------  
# 经济维度  
# --------------------------  
经济 = load\_data(data\_path['经济'], skiprows=1, usecols="A,H,I,P,Q")  
# 经济 = pd.read\_excel(data\_path['经济'],skiprows=1,usecols="A,B,G,H,M,O,P")  
经济 = 经济.rename(columns={  
 '人均生产总值(元)': '人均GDP',  
 '社会消费品零售总额(亿元)': '消费总额',  
 '城镇居民人均可支配收入(元)': '城镇居民收入',  
 '农村居民人均可支配收入(元)': '农村居民收入'  
})  
# --------------------------  
# 环境维度  
# --------------------------  
# 气温  
气温 = load\_data(data\_path['环境\_气温'], usecols="A,N")  
气温.columns = ['城市', '年平均气温']  
  
降水 = load\_data(data\_path['环境\_降水'], usecols="A,N")  
降水.columns = ['城市', '年降水量']  
  
空气 = load\_data(data\_path['环境\_空气'], skiprows=1)  
空气 = 空气[['城市', '细颗粒物(PM2.5)年平均浓度(微克/立方米)', '空气质量达到和好于二级的天数比例(%)']]  
空气.columns = ['城市', 'PM2.5', '空气质量优良天数比例']  
# 合并环境数据  
环境 = pd.merge(气温, 降水, on='城市').merge(空气, on='城市')  
print(环境)  
  
# --------------------------  
# 基础设施维度  
# --------------------------  
教育 = load\_data(data\_path['基础设施\_教育'],skiprows=1)  
教育 = 教育.rename(columns={'普通中学(万人)': '普通中学在校生', '小学(万人)': '小学在校生'})  
交通 = load\_data(data\_path['基础设施\_交通'],skiprows=1)  
交通.rename(columns={'客运量(万人)':'公路、水路','Unnamed:2':'轨道交通（万人次）','Unnamed:3':'航空','货运量(万吨)':'公路、水路.1','Unnamed:5':'航空.1'},inplace=*True*)  
交通 = 交通[['城市', '公路、水路' , '轨道交通（万人次）', '航空']] # 选取关键指标  
# --------------------------  
# 社会文化维度  
# --------------------------  
文化卫生 = load\_data(data\_path['社会文化'],skiprows=1)  
文化卫生 = 文化卫生.rename(columns={  
 '博物馆数（个）': '博物馆数量',  
 '医院床位数(张)': '医院床位数',  
 '医生数(人)': '医生数量'  
})  
# --------------------------  
# 生活便利维度  
# --------------------------  
互联网 = load\_data(data\_path['生活便利\_互联网'],skiprows=1)  
互联网 = 互联网.rename(columns={  
 '移动互联网用户(万户)': '移动互联网用户',  
 '（固定）互联网宽带接入用户(万户)': '宽带用户'  
})  
ICT = load\_data(data\_path['生活便利\_ICT'],skiprows=1)  
ICT = ICT[['城市', '每百人使用计算机数(台)', '每百家企业拥有网站数(个)']]  
# --------------------------  
# 可视化  
# --------------------------  
*def* create\_visualization():  
 # 调整子图布局为4行2列  
 fig = make\_subplots(  
 rows=4, cols=2,  
 specs=[  
 [{"type": "bar"}, {"type": "bar"}], # 经济与环境指标分两列  
 [{"type": "bar"}, {"type": "bar"}], # 基础设施  
 [{"type": "bar"}, {"type": "bar"}], # 社会文化  
 [{"type": "bar"}, {"type": "bar"}], # 生活便利  
 ],  
 subplot\_titles=(  
 "经济指标对比", "环境指标-气温与降水",  
 "教育基础设施", "交通基础设施",  
 "医疗资源", "文化资源",  
 "互联网普及", "数字化程度"  
 )  
 )  
 # 经济指标 (第一行左)  
 metrics = ['人均GDP', '消费总额', '城镇居民收入']  
 *for* metric *in* metrics:  
 fig.add\_trace(  
 go.Bar(x=经济['城市'], y=经济[metric], name=metric),  
 row=1, col=1  
 )  
 # 环境指标 (第一行右)  
 fig.add\_trace(  
 go.Bar(x=环境['城市'], y=环境['年平均气温'], name='年平均气温(℃)'),  
 row=1, col=2  
 )  
 fig.add\_trace(  
 go.Bar(x=环境['城市'], y=环境['年降水量'], name='年降水量(mm)'),  
 row=1, col=2  
 )  
 # 教育基础设施 (第二行左)  
 fig.add\_trace(  
 go.Bar(x=教育['城市'], y=教育['普通中学在校生'], name='普通中学在校生(万人)'),  
 row=2, col=1  
 )  
 # 交通基础设施 (第二行右)  
 fig.add\_trace(  
 go.Bar(x=交通['城市'], y=交通['轨道交通（万人次）'], name='轨道交通客运量'),  
 row=2, col=2  
 )  
 # 医疗资源 (第三行左)  
 fig.add\_trace(  
 go.Bar(x=文化卫生['城市'], y=文化卫生['医院床位数'], name='医院床位数'),  
 row=3, col=1  
 )  
 # 文化资源 (第三行右)  
 fig.add\_trace(  
 go.Bar(x=文化卫生['城市'], y=文化卫生['博物馆数量'], name='博物馆数量'),  
 row=3, col=2  
 )  
 # 互联网普及 (第四行左)  
 fig.add\_trace(  
 go.Bar(x=互联网['城市'], y=互联网['移动互联网用户'], name='移动互联网用户(万户)'),  
 row=4, col=1  
 )  
 # 数字化程度 (第四行右)  
 fig.add\_trace(  
 go.Bar(x=ICT['城市'], y=ICT['每百人使用计算机数(台)'], name='每百人计算机数'),  
 row=4, col=2  
 )  
 # 调整布局  
 fig.update\_layout(  
 height=1600,  
 width=1200,  
 title\_text="杭州、宁波、温州宜居性对比分析",  
 showlegend=*True*,  
 barmode='group' # 同类指标分组显示  
 )  
 fig.show()  
# 执行可视化  
create\_visualization()

运行后会弹出一个网站，网站的效果如下:



当鼠标放上时会显示具体的数值:



至此实现了交互式的图表

最后是对数据进行分析:

**一、经济维度分析**

从经济实力和居民生活水平来看，**杭州**表现最为突出。根据《各市国民经济主要指标》数据，杭州 2023 年生产总值达 20059 亿元，人均生产总值 161129 元，均大幅领先宁波（16453 亿元，170363 元）和温州（8731 亿元，89821 元），经济规模和效率优势显著。其社会消费品零售总额 7670.57 亿元，反映消费市场活跃度高，居民购买力强；财政总收入和一般公共预算收入分别为 5338.72 亿元和 2616.81 亿元，政府财政实力雄厚，可支撑更多公共服务投入。  
宁波的外贸优势突出，出口总额 8287.82 亿元，远超杭州和温州，显示其外向型经济特征；人均生产总值略高于杭州，反映产业附加值较高。温州经济规模较小，但民营经济活跃，社会消费品零售总额 4257.08 亿元，居民消费需求旺盛，但人均收入和财政指标弱于杭甬。  
**结论**：杭州经济综合实力最强，宁波外贸和效率优势显著，温州民营经济活跃但整体规模较弱。

**二、环境维度分析**

从城市规划和基础设施间接关联来看，杭州和宁波在绿化覆盖率、生态保护投入（如财政支出中的环保相关领域）等方面通常领先全省，温州近年来也在推进 “美丽温州” 建设。

**三、基础设施维度分析**

**杭州**在教育和交通基础设施上优势明显。教育方面，高等学校在校生 63.64 万人，远超宁波（25.22 万人）和温州（18.39 万人），普通中学和小学在校生数量也居首位，优质教育资源富集。交通领域，杭州客运量（公路、水路、轨道交通、航空）合计 145,206.83 万人次，其中轨道交通 136290 万人次，航空 4117.05 万人次，立体化交通网络发达，通勤便利性高。  
宁波货运能力突出，货运量 83910.20 万吨，依托港口优势成为物流枢纽；温州客运量以公路为主（4831.54 万人），但轨道交通和航空规模较小。  
**结论**：杭州教育和交通基础设施全面领先，宁波货运物流优势显著，温州基础设施均衡但规模较小。

**四、社会文化维度分析**

在文化和卫生事业上，**杭州**再度领先。体育场地数 41917 个、博物馆 87 个、公共图书馆藏书 3182 万册，均为三城之首，文化设施丰富度高；医疗资源方面，医院数 414 个、床位数 95220 张、医生数 62661 人，远超宁波（215 个医院，44792 张床位）和温州（171 个医院，44130 张床位），居民享受更高质量的医疗服务。  
宁波博物馆数 78 个、藏书 1472 万册，卫生资源较均衡；温州文化设施数量较少，但医疗资源基本满足需求。  
**结论**：杭州文化和卫生资源优势显著，宁波次之，温州稍弱。

**五、生活便利维度分析**

从间接指标看，杭州作为数字经济中心，社会消费品零售总额中线上消费占比更高，信息基础设施建设通常领先。宁波和温州在电商和数字化服务上也较为发达，但缺乏具体数据支撑对比。

**综合评价与宜居性结论**

**杭州**在经济实力、教育资源、文化卫生设施、交通便利性等核心维度全面领先，适合对就业、教育、生活品质要求高的人群。

**宁波**外贸经济强劲，人均收入高，港口物流发达，宜居性体现在产业均衡和生活成本相对可控（房价、通勤压力低于杭州）。

**温州**民营经济活跃，生活氛围浓厚，但经济规模、公共资源总量弱于杭甬，适合偏好商业环境和本地生活的人群。

**最终结论**：若以综合公共服务和发展机会为核心，**杭州**宜居性最优；若考虑产业特色和生活成本，**宁波**更具竞争力；温州则在民营经济活力上独具优势。

|  |  |  |
| --- | --- | --- |
| 组员姓名 | 分工 | 工作量比例 |
| 何丰毅 | Task1,Task2,Task3数据爬取,代码编写,数据解析 | 100% |
|  |  |  |
|  |  |  |