

《数据可视化实验》

报告

**学 号： 22009200894**

**姓 名： 王越洋**

**《数据可视化实验》报告评分表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **课程名称** | **数据可视化** | | | | |
| **学 号** | **22009200894** | **姓名** | **王越洋** | **成绩** |  |
| **题 目** | 实验四：地理数据可视化 | | | | |
| **要 求** | 任务一：统计截止4月1日各省的累计确诊数量，并通过图表展示。  任务二：对每日各省的累计确诊患者数量进行统计，为图像添加时间轴，通过时间轴自动播放演示疫情发展的整个过程。 | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **评价项目** | **评价标准** | **满分** | **得分** |
| 方案合理性 | 能够根据题目的要求，对问题进行分析，利用了一定的基础理论和科学方法来解决实际工程问题，给出了完整的解决方案，报告中理论准备充分，有充分的分析论证过程。 | **15** |  |
| 结果正确性 | 报告数据与分析详实、正确、可信；  设计、实验及测试数据的充分性和可靠性；  理论推导或建模的严密性和完整性。 | **15** |  |
| 报告水平 | 全面了解本领域的动态，并能很好地评述研究背景（文献、市场、需求等）；  报告内容涉及较为深入的基础理论知识、专业技术知识、相关学科专业知识；  具有适当的技术难度，工作量较大。 | **25** |  |
| 写作能力 | 图表清晰，语言规范，符合实验报告要求；  设计、图表、软件的规范化和标准化；  文字表达的专业性、通顺性和概念的准确性；  论文构架的系统性、逻辑性和严谨性；  引用文献的真实性、贴切性和规范性。 | **15** |  |
| 演示视频 | 视频图像和声音清晰，能结合文档更好地展现作业的完成情况。 | **15** |  |
| 使用现代工具 | 能够针对复杂工程问题，开发、选择与使用恰当的技术、资源、现代工程工具和信息技术工具，包括对复杂工程问题的预测与模拟，并能够理解其局限性。 | **15** |  |
| **总 分** | | |  |

**评阅人（签字）： 2023年 月 日**

|  |
| --- |
| **一、实验目的及要求**  1.实验目的：  掌握地理数据可视化的原理和方法。  2.实验内容：  分析2020 新冠疫情。要求利用已有数据（CityData.xlsx）与已有工具，设计可视化方案，展示2020年全国新冠患者人数随时间的变化过程（右图仅供参考），并完成以下任务。  任务一：统计截止4月1日各省的累计确诊数量，并通过图表展示。  任务二：对每日各省的累计确诊患者数量进行统计，为图像添加时间轴，通过时间轴自动播放演示疫情发展的整个过程。  **二、实验环境配置**  1.编程语言：Python  2.所需库：  - pandas：用于数据处理和读取 Excel 文件。  - pyecharts：用于生成地理数据可视化地图。  - pyecharts.options：设置地图的视觉元素和颜色区间。  3.数据集：CityData.xlsx，包含疫情数据（日期、各省名称、确诊病例数量等）。  4.运行环境：Python 3.9，安装 pandas 和 pyecharts。  **三、实验步骤**  **任务一：**  1.数据读取：  使用pandas的read\_excel函数读取CityData.xlsx文件中的数据，该文件包含各省的疫情数据，包括省份名称、更新时间、抓取人数等信息。  利用groupby对数据按“updateTime”和“provinceName”进行分组，以方便后续的数据处理。  2.数据处理：  初始化列表：data和dates，分别用于两个存储分区后的省份精确数据和所有独立的日期。  遍历分组后的数据，提取省份名称、日期和精确数值，将其以[省份名称, 日期, 累计确诊人数]形式添加到data列表中，将日期添加到dates列表中以去重。这样处理后的data包含了每个省份在每个日期的累计精确值数据。  3.初始化省份估算：  为了记录每个省份的初始精确数，初始化一个num\_last列表，其中包含每个省份的名称和对应的精确数（初始值为0）。使用copy()方法对数据进行复制，确保不会直接修改原始数据，以便于绘图时保持数据的一致性。  4.获取4月1日的数据：  在data列表中找到日期为“2020-04-01”的数据，并更新num\_4\_1列表中的精确定位数据。这一步确保获取的是4月1日这一天所有省份的精确定位数据，用于相似该日期的柱状图。  5．绘制柱状图：  使用pyecharts.charts.Bar创建柱状图，以“2020年4月1日”y为轴数据，将x轴设置为各省名称。  调用add\_xaxis方法将省份名称列表确定，调用add\_yaxis方法将4月1日的精确数据确定。  通过set\_global\_opts设置柱状图的全局属性，如标题和横轴标签的旋转角度（小区名称重叠）。  使用render方法将生成的柱状图保存为HTML文件（bar.html）。  **任务二：**  1.数据读取：  利用 pandas 库读取 Excel 文件 CityData.xlsx，提取疫情相关数据。具体读取每一行的“省份名称”、“更新时间”和“确诊人数”。  2.数据处理：  通过 groupby 方法对数据按照 “更新时间”和“省份名称”进行分组，以获取各省每日的累计确诊人数。  将每个省份的名称、对应的日期、确诊人数存入列表 data 中。  将所有日期存入 dates 列表，并确保日期不重复。  3.初始化省份确诊人数：  遍历 data 列表，初始化每个省份的累计确诊人数，保存在 num\_last 列表中。此列表存储了每个省份的初始确诊人数，用于之后生成地图时的数据更新。  4.创建时间线图对象：  利用 pyecharts.charts.Timeline 创建时间线图，并设置播放间隔时间为 500 毫秒，以控制疫情地图的动态展示效果。  5.设置颜色区间：  配置地图的颜色区间 (pieces)，根据确诊人数的不同范围设置不同的颜色，以便直观显示各省的疫情严重程度。  6.生成地图数据并添加到时间线：  遍历每个日期，将各省份在该日期的确诊人数与 num\_last 中的初始人数进行匹配和更新。  使用 pyecharts.charts.Map 创建中国地图，并通过 .add() 方法将日期和数据对（省份名称和确诊人数）添加到地图中。  设置地图的视觉元素，包括颜色区间和布局位置。  将生成的地图通过 Timeline.add() 添加到时间线中，确保每个时间点的地图都在时间线中显示。  7.渲染结果：  调用 Timeline.render() 方法，将生成的疫情时间线地图渲染为 HTML 文件（citydata.html）。  **四、实验结果**  任务一：  通过柱状图，可以仔细观察各省在该日期的疫情严重程度。柱状图的横轴显示了省份名称，纵轴显示了相应的坐标  任务二：  生成了一个包含全国疫情数据变化的动态地图。该地图显示了各省份在不同日期的疫情精准定位，颜色深浅根据精准定位的多个疫情变化。地图包含了时间轴，可以通过播放或手动拖动时间轴，查看不同日期的疫情发展情况。  **五、源代码**  **任务一：**   1. import pandas as pd 2. from pyecharts import options as opts 3. from pyecharts.charts import Bar 4. import copy 5. # 读取 Excel 数据 6. df = pd.read\_excel("CityData.xlsx") 7. data = [] 8. dates = [] 9. # 数据处理 10. for line in df.groupby(["updateTime", "provinceName"]): 11. line\_list = line[1].values 12. *# 保持省份名称不做任何清理，保留原始数据* 13. data.append([line\_list[0][1], str(line\_list[0][7]).split(" ")[0], 14. line[1].city\_confirmedCount.sum()]) 15. date = str(line\_list[0][7]).split(" ")[0] 16. if date not in dates: 17. dates.append(date) 18. # 初始化每个省份的累计确诊人数 19. num\_last = [] 20. for i in data: 21. if i[0] not in (line[0] for line in num\_last): 22. num\_last.append([i[0], 0])  # 保留原始省份名称 23. # 获取4月1日的累计确诊人数 24. num\_4\_1 = num\_last.copy() 25. for i in data: 26. for line in num\_4\_1: 27. if i[1] == "2020-04-01" and line[0] == i[0]: 28. line[1] = int(i[2]) 29. # 绘制4月1日的各省累计确诊感染人数柱状图 30. province\_names = [line[0] for line in num\_4\_1]  # 提取省份名称 31. confirmed\_counts = [line[1] for line in num\_4\_1]  # 提取确诊人数 32. bar = ( 33. Bar(init\_opts=opts.InitOpts(width="1500px", height="650px")) 34. .add\_xaxis(province\_names) 35. .add\_yaxis("2020年4月1日", confirmed\_counts) 36. .set\_global\_opts( 37. title\_opts=opts.TitleOpts(title="2020年4月1日各省累计确诊人数"), 38. xaxis\_opts=opts.AxisOpts(axislabel\_opts=opts.LabelOpts(rotate=-30)) 39. ) 40. ) 41. # 渲染柱状图为 HTML 文件 42. bar.render("bar.html")   代码分析：   1. 数据处理    通过 groupby 函数按 updateTime 和 provinceName 对数据进行分组。   将每个省份在每个日期的累计确诊人数、日期和省份名称添加到 data 列表中。日期使用 .split() 只提取年、月、日部分，以确保格式统一。   将所有不重复的日期添加到 dates 列表中。   1. 初始化每个省份的累计确诊人数    初始化 num\_last 列表，用于记录每个省份的累计确诊人数，并初始化为 0。确保省份名称在不做任何清理的情况下被保留下来。   1. 获取 4 月 1 日的累计确诊人数    通过复制 num\_last 列表得到 num\_4\_1，用于记录 4 月 1 日的累计确诊人数。   遍历 data 列表，匹配日期为 "2020-04-01" 的数据，并更新 num\_4\_1 中每个省份的确诊人数。   1. 绘制柱状图    提取 num\_4\_1 中的省份名称和累计确诊人数，分别存入 province\_names 和 confirmed\_counts 列表中。   使用 pyecharts 的 Bar 类创建柱状图，设置宽度和高度，并将数据添加到图表的 x 和 y 轴。   使用 set\_global\_opts 设置全局属性，包括图表标题和横轴标签的旋转角度，以避免省份名称重叠  (5) 渲染柱状图为 HTML 文件  **任务二：**   1. import pandas as pd 2. from pyecharts import options as opts 3. from pyecharts.charts import Map, Timeline 4. *# 读取 Excel 数据* 5. df = pd.read\_excel("CityData.xlsx") 6. data = [] 7. dates = [] 8. *# 数据处理* 9. for line in df.groupby(["updateTime", "provinceName"]): 10. line\_list = line[1].values 11. *# 这里保持省份名称不做任何清理，保留原始数据* 12. data.append([line\_list[0][1], str(line\_list[0][7]).split(" ")[0], 13. line[1].city\_confirmedCount.sum()]) 14. date = str(line\_list[0][7]).split(" ")[0] 15. if date not in dates: 16. dates.append(date) 17. *# 初始化每个省份的累计确诊人数* 18. num\_last = [] 19. for i in data: 20. if i[0] not in (line[0] for line in num\_last): 21. num\_last.append([i[0], 0])  *# 保留原始省份名称* 22. *# 创建时间线图对象* 23. t = Timeline(init\_opts=opts.InitOpts(width="1400px", height="700px")) 24. t.add\_schema(play\_interval=500) 25. *# 配置颜色区间* 26. pieces = [ 27. {'max': 1, 'label': '0', 'color': 'white'}, 28. {'min': 1, 'max': 20, 'label': '1-20', 'color': '#FFFAFA'}, 29. {'min': 20, 'max': 50, 'label': '20-50', 'color': '#FAF0E6'}, 30. {'min': 50, 'max': 100, 'label': '50-100', 'color': '#FFEFD5'}, 31. {'min': 100, 'max': 200, 'label': '100-200', 'color': '#FFDAB9'}, 32. {'min': 200, 'max': 500, 'label': '200-500', 'color': '#FFB6C1'}, 33. {'min': 500, 'max': 1000, 'label': '500-1000', 'color': '#FF69B4'}, 34. {'min': 1000, 'max': 5000, 'label': '1000-5000', 'color': '#FF4500'}, 35. {'min': 5000, 'max': 10000, 'label': '5000-10000', 'color': '#DC143C'}, 36. {'min': 10000, 'max': 50000, 'label': '10000-50000', 'color': '#B22222'}, 37. {'min': 50000, 'max': 100000, 'label': '50000-100000', 'color': '#800000'} 38. ] 39. *# 生成地图数据并添加到时间线* 40. for date in dates: 41. num = num\_last.copy()  *# 保留原始省份名称和累积值* 42. for i in data: 43. for line in num: 44. if i[1] == date and line[0] == i[0]: 45. *# 使用完整省份名称进行匹配* 46. line[1] = int(i[2]) 47. m = ( 48. Map() 49. .add(series\_name=date, data\_pair=num, maptype="china", 50. is\_map\_symbol\_show=False) 51. .set\_global\_opts( 52. visualmap\_opts=opts.VisualMapOpts(is\_piecewise=True, 53. pieces=pieces, pos\_top="25%", pos\_left="3%") 54. ) 55. ) 56. t.add(m, date) 57. t.render("地理图可视化.html")   代码分析：   1. 数据处理    使用 groupby 对数据按照 "updateTime" 和 "provinceName" 进行分组，获取每个省份每天的累计确诊人数，并将结果存入 data 列表中。   提取日期并存入 dates 列表，确保不重复。   1. 初始化每个省份的累计确诊人数   初始化 num\_last 列表，记录每个省份的累计确诊人数，初始值为 0。  (3) 创建时间线图对象  使用 pyecharts 中的 Timeline 类创建时间线图对象，设置图表的宽度、高度和播放间隔。  (4) 配置颜色区间  设置颜色区间，根据确诊人数的范围设置不同的颜色，便于地图中显示疫情的严重程度。  (5) 生成地图数据并添加到时间线   遍历每个日期，创建地图数据。使用 num\_last.copy() 初始化当天的确诊人数列表 num，然后根据日期匹配更新数据。   使用 pyecharts 的 Map 类生成中国地图，将日期和数据对添加到地图中，并将地图添加到时间线中。  (6) 渲染为 HTML 文件 |