**《Python高级应用》实验报告**

实验名称 ： 实验2 科学计算实验1

实验日期 ： 　 2024/9/23

姓 名 ： 高心阳

学 号 ： 084623237

班 级 ： 计算机232

成 绩 ：

**人工智能与信息技术学院**

**南京中医药大学**

|  |
| --- |
| **实验目的：** |
| 1. 配置并熟悉科学计算的三个库 2. 掌握三种库的使用 |
| **实验内容和要求** |
| 1. **Numpy库练习：**   (1) 创建一个shape为(5,6)的二维全为0的ndarray对象，然后让第2,3行的第2-4列元素等于1;  (2) 使用np.random库创建一个10\*10的整型ndarray对象，并打印出最大最小元素;  (3) 对第2题中的矩阵，计算最后两列的和;  (4) 创建一个长度为10的一维随机数组并排序;  (5) 创建一个一维长度为15的随机矩阵和一个9随机矩阵，将前者使用reshape改为5\*3的矩阵，与3\*3的矩阵，求矩阵积。  ‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬   1. **Pandas库练习：**   （1）创建一个包含以下数据的 DataFrame：   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Name | Age | City | Salary | Department | Years\_of\_work | | Alice | 27 | Nanjing | 8000 | IT | 3 | | Bob | 24 | New York | 5000 | HR | 2 | | Charlie | 22 | Los Angeles | 6500 | IT | 2 | | David | 35 | Chicago | 5200 | Finance | 6 | | Ouki | 39 | Beijing | 20000 | Finance | 10 | | Rose | 25 | Xian | 6500 | HR | 3 |     （2）基于上述数据，找出所有年龄大于 25 且工资大于 6000 的员工的姓名和工资。  （3）基于题(1)计算每个部门的平均工资，并找出哪个部门的总工资最高。  **3、matplotlib库练习：**  （1）绘制题2中每个部门的平均工资条形图，展示不同部门的工资水平差异。  **要求：**   * X 轴表示部门名称，Y 轴表示平均工资。 * 条形图上每根柱子显示具体的平均工资数值。   （2）绘制一个热力图，显示每个部门与员工工作年限和工资之间的关系。  **要求：**   * 横轴表示 "Department"（部门），纵轴表示 "Years\_of\_work"（工作年限），每个方格的颜色深浅代表工资的高低。   **4、综合应用**  （1）缺失值处理：鸢尾花数据集共150个样本，每个样本5个字段，其中四个是特征变量，即萼片长度(Sepal\_len)、萼片宽度(Sepal\_with)、花瓣长度(Petal\_len)、花瓣宽度(Petal\_wid)，还有一个字段是其所属的品种的类别变量(species)。下表是鸢尾花数据集的前15个样本数据，其中有一个缺失值。要求分别使用均值、中位数或某种算法填充缺失值。并与原来的真实值4.8进行比较（通过方差、标准差等方法）。   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | Sepal\_len | Sepal\_with | Petal\_len | Petal\_wid | | 0 | 5.1 | 3.5 | 1.4 | 0.2 | | 1 | 4.9 | 3.0 | 1.4 | 0.2 | | 2 | NaN | 3.2 | 1.3 | 0.2 | | 3 | 4.6 | 3.1 | 1.5 | 0.2 | | 4 | 5.0 | 3.6 | 1.4 | 0.2 | | 5 | 5.4 | 3.9 | 1.7 | 0.4 | | 6 | 4.6 | 3.4 | 1.4 | 0.3 | | 7 | 5.0 | 3.4 | 1.5 | 0.2 | | 8 | 4.4 | 2.9 | 1.4 | 0.2 | | 9 | 4.9 | 3.1 | 1.5 | 0.1 | | 10 | 5.4 | 3.7 | 1.5 | 0.2 | | 11 | 4.8 | 3.4 | 1.6 | 0.2 | | 12 | 4.8 | 3.0 | 1.4 | 0.1 | | 13 | 4.3 | 3.0 | 1.1 | 0.1 | | 14 | 5.8 | 4.0 | 1.2 | 0.2 |   （2）对上述填充后的数据进行探索，画出各个属性对应的散点图及箱线图。分析上限、下限、异常值等。  （3）对上述填充后的数据进行规范化，可选最大最小规范化、零均值标准化、  小数定标规范化等方法。 |
| **运行结果（写清题号）** |
| 1. **Numpy库练习：**   **# 1-1**  **import numpy as np arr=np.zeros(shape=(5,6)) arr[1:3,1:4]=1 for row in range(0,5,1):  for col in range(0,6,1):  print(arr[row][col],end=' ')  print()**  **0.0 0.0 0.0 0.0 0.0 0.0**  **0.0 1.0 1.0 1.0 0.0 0.0**  **0.0 1.0 1.0 1.0 0.0 0.0**  **0.0 0.0 0.0 0.0 0.0 0.0**  **0.0 0.0 0.0 0.0 0.0 0.0**  **# 1-2 import numpy as np from random import randint arr=np.random.randint(low=randint(0,50), high=randint(50,100),size=(10,10)) print("最大值为：",arr.max()) print("最小值为：",arr.min())**  **最大值为： 80**  **最小值为： 48**  **# 1-3 sum\_last\_two\_column=arr[:,-2:].sum() print("最后两列的和：",sum\_last\_two\_column)**  **最后两列的和： 1240**  **# 1-4**  **import numpy as np # 创建一个长度为10的一维随机数组并排序 arr=np.random.randint(0,100,10) arr.sort() print(arr)**  **[ 7 20 20 37 38 41 58 77 83 85]**  **import numpy as np a = np.random.rand(15) b = np.random.rand(9) a = a.reshape(5, 3) b = b.reshape(3, 3) c = np.dot(a, b) print(c)**  **[[0.42817001 1.39976975 1.09759084]**  **[0.57751759 0.92432039 1.25439084]**  **[0.33366037 0.79194117 0.71574901]**  **[0.33953903 1.07345113 0.86492962]**  **[0.44295003 0.96212255 0.83687772]]**   1. **Pandas库练习：**   **import pandas as pd data = {  'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Ouki', 'Rose'],  'Age': [27, 24, 22, 35, 39, 25],  'City': [None, 'New York', 'Los Angeles', 'Chicago', 'Beijing', 'Xian'],  'Salary': [8000, 5000, 6500, 5200, 20000, 6500],  'Department': ['IT', 'HR', 'IT', 'Finance', 'Finance', 'HR'],  'Years\_of\_work': [3, 2, 2, 6, 10, 3] } df = pd.DataFrame(data) print("problem 2-1 result:\n",df)**  **Name Age City Salary Department Years\_of\_work**  **0 Alice 27 None 8000 IT 3**  **1 Bob 24 New York 5000 HR 2**  **2 Charlie 22 Los Angeles 6500 IT 2**  **3 David 35 Chicago 5200 Finance 6**  **4 Ouki 39 Beijing 20000 Finance 10**  **5 Rose 25 Xian 6500 HR 3**  **# 2-2 result = df[(df['Age'] > 25) & (df['Salary'] > 6000)] print("problem 2-2 result:\n",result)**  **Name Age City Salary Department Years\_of\_work**  **0 Alice 27 None 8000 IT 3**  **4 Ouki 39 Beijing 20000 Finance 10**  **# 2-3 # 基于题(1)计算每个部门的平均工资，并找出哪个部门的总工资最高。 avg\_salary = df.groupby('Department')['Salary'].mean() max\_salary = avg\_salary.idxmax() print("problem 2-3 result:\n",avg\_salary)**  **Department**  **Finance 12600.0**  **HR 5750.0**  **IT 7250.0**  **Name: Salary, dtype: float64**   1. **matplotlib库练习：**   **import matplotlib.pyplot as plt plt.rcParams['font.sans-serif'] = ['SimHei'] plt.rcParams['axes.unicode\_minus'] = False avg\_salary.plot(kind='bar', title='平均薪资表') for x, y in enumerate(avg\_salary):  plt.text(x, y, '%.2f' % y, ha='center', va='bottom') plt.show()**    **# 3-2 import seaborn as sns import numpy as np plt.rcParams['font.sans-serif'] = ['SimHei'] plt.rcParams['axes.unicode\_minus'] = False df = pd.DataFrame(data) df = df.pivot\_table(index='Department', columns='Years\_of\_work', values='Salary', aggfunc=np.mean) sns.heatmap(df, cmap='YlGnBu', annot=True) plt.show()**     1. **综合应用**   **# 4-1 import numpy as np import pandas as pd # 方差 def variance(data):  mean = np.mean(data)  return np.mean((data - mean) \*\* 2) # 标准差 def standard\_deviation(data):  return np.sqrt(variance(data))  data = {  'Sepal\_len': [5.1, 4.9, np.nan, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.8, 4.8, 4.3, 5.8],  'Sepal\_with': [3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1, 3.7, 3.4, 3.0, 3.0, 4.0],  'Petal\_len': [1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.6, 1.4, 1.1, 1.2],  'Petal\_wid': [0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.1, 0.2] } true\_data=4.8 df = pd.DataFrame(data) # 均值 df.fillna(df.mean(), inplace=True) print("补充值：", df['Sepal\_len'][3], "方差：", variance(df['Sepal\_len']), "标准差：", standard\_deviation(df['Sepal\_len'])) # 中位数 df.fillna(df.median(), inplace=True) print("补充值：", df['Sepal\_len'][3], "方差：", variance(df['Sepal\_len']), "标准差：", standard\_deviation(df['Sepal\_len'])) # 众数 df.fillna(df.mode().iloc[0], inplace=True) print("补充值：", df['Sepal\_len'][3], "方差：", variance(df['Sepal\_len']), "标准差：", standard\_deviation(df['Sepal\_len']))**  **补充值： 4.6 方差： 0.1445714285714286 标准差： 0.3802254970033291**  **补充值： 4.6 方差： 0.1445714285714286 标准差： 0.3802254970033291**  **补充值： 4.6 方差： 0.1445714285714286 标准差： 0.3802254970033291**  **# 4-2 import matplotlib.pyplot as plt import seaborn as sns # 散点图 sns.pairplot(df) plt.show() # 箱线图 df.boxplot() plt.show()print("Sepal\_len上限：", df['Sepal\_len'].max(), "Sepal\_len下限：", df['Sepal\_len'].min()) print("Sepal\_with上限：", df['Sepal\_with'].max(), "Sepal\_with下限：", df['Sepal\_with'].min()) print("Petal\_len上限：", df['Petal\_len'].max(), "Petal\_len下限：", df['Petal\_len'].min()) print("Petal\_wid上限：", df['Petal\_wid'].max(), "Petal\_wid下限：", df['Petal\_wid'].min())**  **print("Sepal\_len异常值：", df['Sepal\_len'][df['Sepal\_len'] > df['Sepal\_len'].max()]) print("Sepal\_with异常值：", df['Sepal\_with'][df['Sepal\_with'] > df['Sepal\_with'].max()]) print("Petal\_len异常值：", df['Petal\_len'][df['Petal\_len'] > df['Petal\_len'].max()]) print("Petal\_wid异常值：", df['Petal\_wid'][df['Petal\_wid'] > df['Petal\_wid'].max()])**      **Sepal\_len上限： 5.8 Sepal\_len下限： 4.3**  **Sepal\_with上限： 4.0 Sepal\_with下限： 2.9**  **Petal\_len上限： 1.7 Petal\_len下限： 1.1**  **Petal\_wid上限： 0.4 Petal\_wid下限： 0.1**  **Sepal\_len异常值： Series([], Name: Sepal\_len, dtype: float64)**  **Sepal\_with异常值： Series([], Name: Sepal\_with, dtype: float64)**  **Petal\_len异常值： Series([], Name: Petal\_len, dtype: float64)**  **Petal\_wid异常值： Series([], Name: Petal\_wid, dtype: float64)**  **# 4-3 # 最大最小规范化 def max\_min\_normalization(data):  return (data - data.min()) / (data.max() - data.min()) df\_max\_min = df.copy() df\_max\_min.iloc[:, :] = max\_min\_normalization(df\_max\_min) print("最大最小规范化：\n", df\_max\_min) df\_zero\_mean = df.copy()**  **Sepal\_len Sepal\_with Petal\_len Petal\_wid**  **0 0.533333 0.545455 0.500000 0.333333**  **1 0.400000 0.090909 0.500000 0.333333**  **2 0.419048 0.272727 0.333333 0.333333**  **3 0.200000 0.181818 0.666667 0.333333**  **4 0.466667 0.636364 0.500000 0.333333**  **5 0.733333 0.909091 1.000000 1.000000**  **6 0.200000 0.454545 0.500000 0.666667**  **7 0.466667 0.454545 0.666667 0.333333**  **8 0.066667 0.000000 0.500000 0.333333**  **9 0.400000 0.181818 0.666667 0.000000**  **10 0.733333 0.727273 0.666667 0.333333**  **11 0.333333 0.454545 0.833333 0.333333**  **12 0.333333 0.090909 0.500000 0.000000**  **13 0.000000 0.090909 0.000000 0.000000**  **14 1.000000 1.000000 0.166667 0.333333** |
| **实验的体会与建议** |
|  |