1. 第一题
2. *# %% [markdown]*
3. *# ## 调用*
4. *# %%*
5. import pandas as pd
6. import matplotlib.pyplot as plt
7. excel = pd.read\_excel("B题数据.xlsx")
8. lon\_lat = pd.read\_excel('等高线图.xlsx')
9. x\_label = excel['断面名称']
10. y\_labels = {
11. '高锰酸盐指数(mg/L)': 'purple',
12. '氨氮(mg/L)': 'red',
13. '总磷(mg/L)': 'green',
14. '总氮(mg/L)': 'orange'
15. }
16. plt.rcParams['font.sans-serif'] = ['SimHei']  *#显示中文*
17. plt.rcParams['axes.unicode\_minus']=False *#用来正常显示负号*
19. *# %% [markdown]*
20. *# ## 第一题*
21. *#*
22. *# 1、分析北京地区水系中高锰酸盐指数、氨氮、总磷、总氮的总体分布特征。*
23. *# \*（分布特征分析）\**
24. *#*
25. *# 另外，为了绘图，我把广北滨河路（桥）的（桥）给去掉了，特此说明*
26. *# %% [markdown]*
27. *# ### 数据分析*
28. *# %%*
29. ana = excel[['高锰酸盐指数(mg/L)','氨氮(mg/L)','总磷(mg/L)','总氮(mg/L)']]
30. print(ana.describe())
31. *# %% [markdown]*
32. *# ### 地区分布条形图*
33. *# %%*
34. plt.figure(figsize=[27.50, 12.50])
35. *# 遍历y\_labels字典，为每个指标创建子图*
36. for i, (ylabel, color) in enumerate(y\_labels.items(), start=1):
37. plt.subplot(2, 2, i)  *# 根据循环的索引选择子图位置*
38. plt.bar(x\_label, excel[ylabel], color=color, label=ylabel)
39. plt.xticks(rotation=40, fontsize=10)
40. plt.title(ylabel,fontsize=20)
41. plt.gca().xaxis.set\_tick\_params(labelrotation=40)
43. *# 显示图例*
44. *# plt.legend()*
46. *# 显示图形*
47. plt.show()
48. *# %% [markdown]*
49. *# ### 等高线图*
50. *# %%*
51. point\_label = list(x\_label)
52. *# %%*
53. from scipy.interpolate import griddata
54. *# print(lon\_lat['经度'])*
55. *# 定义一维数组*
56. x = lon\_lat['经度']
57. y = lon\_lat['纬度']
58. point=np.vstack([x,y]).T
59. *# print(list(point))*
60. print(lon\_lat.columns)
61. xx,yy=np.meshgrid(np.linspace(x.min(),x.max(),200),np.linspace(y.min(),y.max(),200))
62. plt.figure(figsize=[27.50, 12.50])
63. for i, (ylabel, color) in enumerate(y\_labels.items(), start=1):
64. index = lon\_lat[ylabel]
65. zz=griddata(points=point, values=index, xi=(xx,yy), method="cubic")
66. plt.subplot(2, 2, i)
67. plt.xticks(fontsize=20)
68. plt.yticks(fontsize=20)
69. plt.title(ylabel,fontsize=25)
70. plt.scatter(x, y, c=color, s=30)
71. plt.contourf(xx, yy, zz, alpha=0.75, cmap=f'{color.title()}s')
72. for j, name in enumerate(x\_label):
73. plt.annotate(name, xy = list(point)[j], textcoords="offset points", xytext=(0,10), ha='center', fontsize=10)
74. plt.tight\_layout()
75. plt.show()
76. 第二题
77. *# %% [markdown]*
78. *# ## 调用*
79. *# %%*
80. import numpy as np
81. import pandas as pd
82. import matplotlib.pyplot as plt
83. import seaborn as sns
84. *# from scipy.stats import spearmanr*
85. import pingouin as pg
86. from sklearn.linear\_model import LinearRegression
87. from sklearn.preprocessing import StandardScaler
88. scaler = StandardScaler()
89. plt.rcParams['font.sans-serif'] = ['SimHei']  *#显示中文*
90. plt.rcParams['axes.unicode\_minus']=False *#用来正常显示负号*
91. plt.rcParams["figure.figsize"] = [27.50, 12.50]
92. excel = pd.read\_excel("B题数据.xlsx")
93. data = excel.iloc[0:,2:]
94. ana\_ff1 = pd.DataFrame(data)
95. *# %% [markdown]*
96. *# ## 第二题*
97. *#*
98. *# 2、分析高锰酸盐指数、氨氮、总磷、总氮的高低，与哪些因素具有相关性，仅限给出的数据项目，并据此评估呈现当前分布特征的原因。*
99. *# \*（相关性分析，评估原因）\**
100. *# %% [markdown]*
101. *# ### 主成分分析法*
102. *# %% [markdown]*
103. *# #### 创建类*
104. *# %%*
105. import numpy as np
106. from numpy import linalg
107. class PCA:
108. ''' dataset 形如array([样本1,样本2,...,样本m]),每个样本是一个n维的ndarray'''
110. def \_\_init\_\_(self, dataset):
111. *# 这里的参数跟上文是反着来的(每行是一个样本)，需要转置一下*
112. self.dataset = np.matrix(dataset, dtype='float64').T
113. *# def standard2(self, dataset):*
114. *#     scaler = StandardScaler()*
115. *#     scaled\_data = scaler.fit\_transform(np.asarray(dataset))*
116. *#     # scaled\_df = pd.DataFrame(scaled\_data)*
117. *#     # return scaled\_df*
119. def standard(self, dataset):
120. '''标准化'''
121. for (index, line) in enumerate(dataset):
122. dataset[index] -= np.mean(line)
123. *# np.std(line, ddof = 1)即样本标准差(分母为n - 1)*
124. dataset[index] /= np.std(line, ddof = 1)
125. def get\_eig\_vector(self, data, Cov):
126. '''求特征值和特征向量'''
127. eigs, vectors = linalg.eig(Cov)
128. *# print(vectors)*
129. for i in range(len(eigs)):
130. data.append((eigs[i], vectors[:, i]))
131. *# 按照特征值从大到小排序*
132. data.sort(key = lambda x: x[0], reverse = True)
133. return eigs
135. def principal\_comps(self, threshold = 0.85):
136. '''求主成分。threshold可选参数表示方差累计达到threshold后就不再取后面的特征向量。
137. 返回值是特征值、特征向量、方差贡献率、累计方差贡献率。'''
138. ret = []
139. data = []
140. self.standard(self.dataset)
141. Cov = np.cov(self.dataset)
142. eigs = self.get\_eig\_vector(data, Cov)
143. sum = 0
144. for comp in data:
145. sum += comp[0] / np.sum(eigs)
146. ret.append(
147. tuple(
148. map(lambda x: np.round(x, 5)
149. *# 特征值、特征向量、方差贡献率、累计方差贡献率*
150. , (comp[0], comp[1], comp[0] / np.sum(eigs), sum))
151. )
152. )
153. if sum > threshold:
154. return ret
155. return ret
156. *# PMx = PCA(anal)*
157. *# op = PMx.principal\_comps(threshold=0.85)*
158. *# %% [markdown]*
159. *# #### 数据准备*
160. *# %%*
161. *#准备原始数据*
162. anal = excel.drop(['高锰酸盐指数(mg/L)', '氨氮(mg/L)','总磷(mg/L)', '断面名称', '总氮(mg/L)', '水质类别'], axis=1)
163. index = ['水温(℃)','pH(无量纲)','溶解氧(mg/L)','电导率(μS/cm)','浊度(NTU)']
164. *#输出用*
165. output = [['高锰酸盐', '高锰酸盐指数(mg/L)'], ['氨氮', '氨氮(mg/L)'], ['总磷', '总磷(mg/L)'],['总氮', '总氮(mg/L)']]
167. *# print(excel.columns)*
168. *#绘制影响因子图用*
169. x\_label = pd.read\_excel('影响因子图.xlsx', sheet\_name='Sheet5', header=None).squeeze()
170. y\_labels = {
171. '高锰酸盐指数(mg/L)': 'purple',
172. '氨氮(mg/L)': 'red',
173. '总磷(mg/L)': 'green',
174. '总氮(mg/L)': 'orange'
175. }
176. def standard(data):
177. '''标准化'''
178. opt = (data - data.mean()) / data.std()
179. return opt
180. *# %% [markdown]*
181. *# #### 分析*
182. *# %%*
183. with pd.ExcelWriter('output.xlsx') as writer:
184. for ii in output:
185. PMx = PCA(anal)
186. op = PMx.principal\_comps(threshold=0.85)
187. opd = pd.DataFrame(op, columns=['特征值', '特征向量', '方差贡献率', '累计方差贡献率'])
188. *# print(opd)*
189. opd.to\_excel('analyse.xlsx', index=True)
190. matrix = pd.DataFrame(index=index)
191. for i in range(len(op)):
192. matrix[i] = op[i][1]
193. *#result是主成分*
194. result = np.dot(anal, matrix)
195. *# print(result)*
196. model = LinearRegression()
197. y = np.array(excel[ii[1]])
198. x = np.array(result)
199. model.fit(y=standard(y), X=x)
200. result\_mid = np.dot(matrix, model.coef\_.T)
201. pd.set\_option('display.float\_format', lambda x: '%.6f' % x)
202. *# print(result2)*
203. matrix['主成分回归系数'] = result\_mid
204. *# print(matrix)*
205. matrix.to\_excel(writer, sheet\_name=ii[0], float\_format='%.6f', index=True)
206. *# print(result)*
207. *# 载荷系数=原始变量与主成分之间的相关系数*
208. for i in range(3):
209. for j in index:
210. load = pd.DataFrame([result[:, i], anal[j]])
211. *# print(load)*
212. print(load.iloc[0].corr(load.iloc[1]))
213. print('\n')
214. *# %% [markdown]*
215. *# #### 影响因子*
216. *# %%*
217. for i, (ylabel, color) in enumerate(y\_labels.items(), start=1):
218. inf = pd.read\_excel('影响因子图.xlsx', sheet\_name=f'Sheet{i}', header=None).squeeze()
219. *# print(list(inf))*
220. *# print(list(x\_label))*
221. plt.subplot(2, 2, i)  *# 根据循环的索引选择子图位置*
222. plt.bar(list(x\_label[i-1]), list(inf), color=color, label=ylabel)
223. plt.xticks(rotation=40, fontsize=15)
224. plt.title(ylabel,fontsize=20)
225. plt.gca().xaxis.set\_tick\_params(labelrotation=40)
226. plt.tight\_layout()
227. plt.show()
228. *# %% [markdown]*
229. *# ## 调用*
230. *# %%*
231. import numpy as np
232. import pandas as pd
233. *# plt.rcParams["figure.figsize"] = [27.50, 12.50]*
234. excel = pd.read\_excel("B题数据.xlsx")
235. ana = pd.DataFrame(excel).iloc[:, 2:]
236. *# print(ana)*
237. 第三题
238. *# %% [markdown]*
239. *# ## 第三题*
240. *#*
241. *# 3、国家标准的I-V分类主要考虑水质本身对人和环境的影响，请从居民、游客生活体验的角度出发，对各个监测站的水质重新分类，并给出分类的依据。*
242. *# \*（分类，分类依据制定）\**
243. *# %% [markdown]*
244. *# ### 熵权法计算原始数据打分到二级标准（实际是一级）的权重*
245. *#*
246. *# 注：不包括水质类别相对应的数据列。一方面是制定打分标准时忘记了，另一方面是后来反思也认为数据给出的水质类别与其他列是相关的，不必算进新的打分标准中*
247. *# %%*
248. stdana = pd.DataFrame()
249. weight = []
250. k = 1 / np.log(len(ana.index))
251. for i, col in enumerate(ana):
252. stdana[i] = ((ana[col]-min(ana[col]))/max(ana[col]-min(ana[col])))
253. stdana[i] /= sum(stdana[i])
254. for ji, j in enumerate(stdana[i]):
255. if j != 0:
256. stdana.iloc[ji, i] \*= np.log(j)
257. else:
258. stdana.iloc[ji, i] = 0
259. weight.append(1-(-k\*sum(stdana[i])))
260. for k in range(len(weight)):
261. weight[k] /= sum(weight)
262. weight = {k:v for k,v in zip(ana.columns, weight)}
263. *# print(weight)*
264. *# %% [markdown]*
265. *# ### 分数计算*
266. *# %%*
267. weight1 = [weight['水温(℃)'], weight['浊度(NTU)'], weight['溶解氧(mg/L)']]
268. weight2 = [weight['高锰酸盐指数(mg/L)'], weight['电导率(μS/cm)'], weight['pH(无量纲)'], weight['氨氮(mg/L)']]
269. weight3 = [weight['氨氮(mg/L)'], weight['溶解氧(mg/L)'], weight['总磷(mg/L)'], weight['总氮(mg/L)']]
270. weight1 /= sum(weight1)
271. weight2 /= sum(weight2)
272. weight3 /= sum(weight3)
273. weightn = [weight1, weight2, weight3]
274. s = pd.DataFrame(columns=['舒适度', '安全性', '清洁度'])
275. for i in range(3):
276. score = pd.read\_excel('分.xlsx', sheet\_name=i)
277. s.iloc[:, i] = np.dot(weightn[i], score.iloc[0:, 1:])
278. weightupper = [0.2, 0.4 ,0.4]
279. s\_last = pd.concat([excel['断面名称'], pd.Series(np.dot(s, weightupper))], axis=1)
280. s\_last.columns = ['断面名称', '总分']
281. s\_last\_sorted = sorted(s\_last.values, key=lambda x: x[1], reverse=True)
282. *# print(excel['断面名称'])*
283. *# print(s\_last\_sorted)*
284. total\_scores = np.array([item[1] for item in s\_last\_sorted])
285. standardized\_scores = (total\_scores - min(total\_scores)) / (max(total\_scores) - min(total\_scores))
286. levels = 6 - np.digitize(standardized\_scores, np.linspace(-0.00001, 1.00001, 6), right=True)
287. s\_last\_sorted\_with\_scores = [np.append(item, [score, level]) for item, score, level in zip(s\_last\_sorted, standardized\_scores, levels)]
288. for item in s\_last\_sorted\_with\_scores:
289. print(item)
290. print(s\_last\_sorted)