《大数据分析与处理》第二次作业

姓名： 李燕琴 学号： 20195633 日期： 2022年3月28日

*说明：作业提交电子版，绘图部分，提交绘图源代码（直接粘贴到word）。*

1. 数据质量可以从多方面评估，包括准确性、完整性和一致性。对于以上每个问题，讨论数据质量的评估如何依赖于数据的应用目的，给出例子。提出数据质量的两个其他度量尺度，并给出相应的例子。（15分）

①准确性：描述数据是否与其对应的客观实体的特征相一致，如绩点上限为4.0，但是给定数据却有40.6的，出现了明显的数据约束错误。

②完整性：描述数据是否存在缺失记录或缺失字段，如部分学生没有参加正考（而是补考），那么该生未参加补考前，成绩是不完整的。

③一致性：描述同一实体的同一属性的值在不同的系统或数据集中是否一致，如对于高等数学有工科类和电子类，统计某学院学生的高等数学成绩时，需要保证统计学生所修高等数学为同一类。

④有效性：描述数据是否满足用户定义的条件或在一定的域值范围内，例子同①中准确性的例子。

⑤唯一性：描述数据是否存在重复记录，对于同一个学生的高等数学成绩，是唯一的，如果有多份高等数学成绩，则数据错误。

数据质量的评估依赖于数据的应用，即对于给定的数据，不同的应用场景因其细粒度要求不同，可能会有完全不同的评估。如，一种是需要掌握股票数据，分析其上升趋势；另一种是需要掌握股票数据，分析股票的类型。前者如果缺失值或无用的参数过多，会导致分析不合理；后者，就算有一定量缺失值，但是大量数据是正确的，也能得到较好的分类效果。

2. 数据清洗是数据预处理中的重要步骤，而缺失值处理又是数据清洗中的典型任务，针对标称属性、二元属性、序数属性和数值属性，分别结合具体的例子，说明属性值缺失的具体处理方法。（15分）

①标称属性：用给定数据的该属性众数填充缺失值，或常量（选择一个值作为默认值）填充，如对于头发颜色= {黑色，棕色，灰色，白色}，如果统计地区在中国，可以将黑色作为默认值，此时黑色一般也为该地区该属性的众数；

②二元属性：用给定数据的该属性众数填充缺失值，或常量（选择一个值作为默认值）填充，如核酸检测的阴性和阳性，一般将阴性作为默认值；

③序数属性：用给定数据的该属性众数中位数填充缺失值，或常量（选择一个值作为默认值）填充，如成绩={高，中，差}，因为序数属性存在顺序，一般使用众数或中位数比较好；

④数值属性：使用最可能的值填充（如采用回归方法推断缺失值，或插值方法），如近日温度变化，可以利用回归或插值的方法进行填充。

3. 求下面矩阵的奇异值分解，并写出其外积展开式。（15分）



根据下列式子可以得到，U为的特征向量构成的矩阵。

同理，V为的特征向量构成的矩阵:

其中



特征值为：，

特征向量为：；



特征值为：，

特征向量为：；

则：







外积展开式为：



源代码如下：

import numpy as np

# 原始数据

A = np.array([

    [2,4],

    [1,3],

    [0,0],

    [0,0]

])

# 计算特征值，特征向量

W = np.dot(A,A.T)

valueW,vecW = np.linalg.eig(W)

X = np.dot(A.T,A)

valueX,vecX = np.linalg.eig(X)

print("A A^T\n",W)

print("A A^T的特征值\n",valueW,"\nA A^T的特征向量\n",vecW)

print("A^T A\n",X)

print("A^T A的特征值\n",valueX,"\nA^T A的特征向量\n",vecX)

# 计算奇异值分解，这里记得按照顺序排序

Sigma = np.array([

    [np.sqrt(valueW[0]),0],

    [0,np.sqrt(valueW[1])],

    [0,0],

    [0,0]

])

U = vecW

V = np.array([vecX[:,1],vecX[:,0]])

V = -1\*V

# 计算外积展开式

res = np.dot(np.dot(U,Sigma),V.T)

print("U\n",U)

print("V\n",V)

print("Sigma\n",Sigma)

print("外积展开式\n",res)

4. 对以下样本数据进行主成分分析。（15分）



解题结果如下：

文本

中度可信度描述已自动生成

源代码如下：

import numpy as np

# 原始数据

X = np.array([

    [2,3,3,4,5,7],

    [2,4,5,5,6,8]

])

print("原始数据\n",X)

# 原始维数

n = X.shape[1]

# 需要降到的维数

p = 1

# 中心化

X = X - np.mean(X,axis=1)[:,np.newaxis]

print("中心化结果\n",X)

# 协方差矩阵

sigma = np.dot(X,X.T)/(n-1) # or: sigma = np.cov(X)

print("协方差矩阵\n",sigma)

# 求特征值和特征向量

valueS, vecS = np.linalg.eig(sigma)

print("特征值\n",valueS,"\n特征向量\n",vecS)

# 获取前p大的特征向量，并按行排练

idx = np.argmax(valueS)

pvecS = vecS.T[idx,:]

print("第一主成分\n",pvecS)

# 得到降维后的数据

res = np.dot(pvecS,X)

print("降维结果\n",res)

5. 综合编程题。给定鸢尾花IRIS数据集（数据及相关说明请参见作业附件“鸢尾花数据集.zip”），该数据集包含150个样本，每个样本包含4个属性以及1个类别标签，编写程序完成如下数据的分析和处理（40分）：

(a) 选择至少一种属性，绘制其分布直方图、盒图，并给出五数概括，根据直方图判断是否服从正态分布。将同一属性不同鸢尾花类别的多个直方图或盒图绘制到同一张图上，对比其属性值分布是否存在差异，并给出结论。

选择属性sepal.length，初步分析如下：

五数概括：[最小值，最大值，第一四分位数，第三四分位数，中位数] = [4.3, 7.9, 5.1, 6.4, 5.8]

由直方分布图可知，鸢尾花sepal.length 不服从正态分布。

图表, 直方图

描述已自动生成

图表, 箱线图

描述已自动生成

多类别分析，可以发现Setosa到Versicolor再到Virginica，其sepal.length值分布趋势呈现递增状态。

图表, 直方图

描述已自动生成图表, 箱线图

描述已自动生成

(b) 绘制不同属性之间的散点图矩阵，并计算相应的Pearson积矩系数和协方差，据此判断属性之间的相关性，并给出结论。

结果如下图，可以看到，sepal.length和petal.length，sepal.length和petal.width，petal.length和petal.width存在较强的相关性。

QR 代码

描述已自动生成

(c) 绘制不同属性之间的q-q图，判断不同属性的分布是否一致。

由下图可知，sepal.width和sepal.length，petal.width和petal.length的分布比较相似。

图片包含 文本

描述已自动生成

(d) 对属性进行最小-最大（min-max）归一化，将其取值归一化到[0,1]范围内，对归一化之后的数据，绘制不同属性之间的q-q图，判断不同属性的分布是否一致。对属性进行零均值（z-score）归一化，然后重复上述步骤绘制q-q图，观察结果是否一样，并给出结论。

由下图可知，结果同（c）中的结论，由此可以看出归一化并不改变数据的分布。

图片包含 文本

描述已自动生成图片包含 图表

描述已自动生成

(e) 调用scikit-learn的PCA包，对数据进行降维处理，将属性由4维降到2维，然后将降维之后的数据可视化到二维平面上，不同类别的数据点用不同颜色标记，观察不同类别的数据点是否能较好地区分开，并给出结论。

经过PCA降维后，得到的二维数据点可视化如下：可以发现，PCA降维后，去除了一些相关属性，使得数据能更好的区分开。

图表, 散点图

描述已自动生成

绘图源代码如下：

1. # 读取数据
2. **import** pandas as pd
4. %matplotlib inline
5. **import** matplotlib as mpl
6. **import** matplotlib.pyplot as plt
7. # 支持中文
8. plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签
9. plt.rcParams['axes.unicode\_minus'] = False  # 用来正常显示负号
11. iris = pd.read\_csv("./数据集1\_iris鸢尾花数据集.csv")
12. col = 'sepal.length'
13. # 1)
14. data = iris[col].values
16. title = "鸢尾花%s直方分布图"%(col)
17. plt.hist(data,label='all')
18. plt.title(title)
19. plt.legend()
20. plt.xlabel(col)
21. plt.grid(alpha=0.4)
22. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
23. plt.show()
25. title = "鸢尾花%s盒图"%(col)
26. plt.boxplot([data])
27. plt.title(title)
28. plt.xlabel(col)
29. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
30. plt.show()
32. **def** get\_five\_summary(data):
33. **return** [np.min(data),np.max(data),np.percentile(data,25),np.percentile(data,75),np.median(data)]
34. **print**("%s的五数概括如下:"%(col))
35. **print**("[最小值，最大值，第一四分位数，第三四分位数，中位数] =",get\_five\_summary(data))
36. # 1)
37. group = list(iris.groupby('variety'))
39. title = "多类别鸢尾花%s直方分布图"%(col)
40. color = ['r','b','g']
41. i = 0
42. **for** name,df **in** group:
43. plt.hist(df[col],
44. histtype='bar',
45. stacked=False,
46. fill=True,
47. label=name,
48. alpha=0.8, # opacity of the bars
49. color=color[i],
50. edgecolor = "k")
51. i+=1
52. plt.legend()
53. plt.title(title)
54. plt.xlabel(col)
55. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
56. plt.show()
58. title = "多类别鸢尾花%s盒图"%(col)
59. data = []
60. names = []
61. **for** name,df **in** group:
62. data.append(df[col])
63. names.append(name)
64. plt.boxplot(data,labels=names)
65. plt.title(title)
66. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
67. plt.show()
68. # 2)
69. cols = ['sepal.length', 'sepal.width', 'petal.length', 'petal.width']
70. title = "不同属性之间的散点图"
72. fig, ax = plt.subplots(nrows=1, ncols=6, figsize=(20, 5))
74. rho\_noise = np.corrcoef(iris[cols].values.T)
75. cov = np.cov(iris[cols].values.T)
76. curr = 0
77. **for** i **in** range(0,4):
78. col\_i = cols[i]
79. **for** j **in** range(i+1,4):
80. col\_j = cols[j]
81. ax[curr].scatter(iris[col\_i], iris[col\_j])
82. ax[curr].set\_xticks([])
83. **if** j!=0:
84. ax[curr].set\_yticks([])
85. ax[curr].title.set\_text('Pearson = %.3f\nCov = %.3f'%(rho\_noise[i,j],cov[i,j]))
86. ax[curr].set(xlabel=cols[i],ylabel=cols[j])
87. curr+=1
89. plt.suptitle(title,fontsize=15,x=0.5,y=1.02)
90. fig.subplots\_adjust(wspace=0.3,hspace=0.4)
91. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
93. fig.show()
94. # 3)
95. **import** seaborn as sns
96. cols = ['sepal.length', 'sepal.width', 'petal.length', 'petal.width']
97. title = "不同属性之间的QQ图"
99. fig, ax = plt.subplots(nrows=1, ncols=6, figsize=(20, 5))
101. curr = 0
102. **for** i **in** range(0,4):
103. col\_i = cols[i]
104. **for** j **in** range(i+1,4):
105. col\_j = cols[j]
106. s\_i = sorted([np.percentile(iris[col\_i],k) **for** k **in** np.linspace(1,100,500)], reverse=True)
107. s\_j = sorted([np.percentile(iris[col\_j],k) **for** k **in** np.linspace(1,100,500)], reverse=True)
108. sns.regplot(s\_i,s\_j,ci=68,color='b',line\_kws={'color':'r'},ax=ax[curr])
109. ax[curr].set(xlabel=cols[i],ylabel=cols[j])
110. curr+=1
112. plt.suptitle(title,fontsize=15,x=0.5,y=1.02)
113. fig.subplots\_adjust(wspace=0.3,hspace=0.4)
114. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
115. fig.show()
116. # 4)
117. **from** sklearn.preprocessing **import** MinMaxScaler,StandardScaler
118. mms = MinMaxScaler()
119. data = mms.fit\_transform(iris[cols])
121. title = "不同属性最小最大归一化后的QQ图"
122. fig, ax = plt.subplots(nrows=1, ncols=6, figsize=(20, 5))
124. curr = 0
125. **for** i **in** range(0,4):
126. **for** j **in** range(i+1,4):
127. s\_i = sorted([np.percentile(data[:,i],k) **for** k **in** np.linspace(1,100,500)], reverse=True)
128. s\_j = sorted([np.percentile(data[:,j],k) **for** k **in** np.linspace(1,100,500)], reverse=True)
129. sns.regplot(s\_i,s\_j,ci=68,color='b',line\_kws={'color':'r'},ax=ax[curr])
130. ax[curr].set(xlabel=cols[i],ylabel=cols[j])
131. curr+=1
133. plt.suptitle(title,fontsize=15,x=0.5,y=1.02)
134. fig.subplots\_adjust(wspace=0.3,hspace=0.4)
135. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
136. fig.show()
138. sds = StandardScaler()
139. data = sds.fit\_transform(iris[cols])
140. title = "不同属性标准归一化后的QQ图"
141. fig, ax = plt.subplots(nrows=1, ncols=6, figsize=(20, 5))
142. curr = 0
143. **for** i **in** range(0,4):
144. **for** j **in** range(i+1,4):
145. s\_i = sorted([np.percentile(data[:,i],k) **for** k **in** np.linspace(1,100,500)], reverse=True)
146. s\_j = sorted([np.percentile(data[:,j],k) **for** k **in** np.linspace(1,100,500)], reverse=True)
147. sns.regplot(s\_i,s\_j,ci=68,color='b',line\_kws={'color':'r'},ax=ax[curr])
148. ax[curr].set(xlabel=cols[i],ylabel=cols[j])
149. curr+=1
151. plt.suptitle(title,fontsize=15,x=0.5,y=1.02)
152. fig.subplots\_adjust(wspace=0.3,hspace=0.4)
153. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
154. fig.show()
155. # 5)
156. **from** sklearn.decomposition **import** PCA
157. data = iris[cols].values
158. label = iris['variety']
160. pca = PCA(n\_components=2)
161. new\_data = pca.fit\_transform(data)
162. **print**("降维后，维度如下：",new\_data.shape)
164. c\_dic = {
165. 'Setosa':'r',
166. 'Virginica':'b',
167. 'Versicolor':'g',
168. }
169. label\_color = [c\_dic[i] **for** i **in** label.values]
170. title = 'PCA降维后的二维可视化'
171. plt.scatter(new\_data[:,0],new\_data[:,1],c=label\_color)
172. plt.title(title)
173. plt.savefig(title+'.png', dpi=300, bbox\_inches="tight", pad\_inches=0.1)
174. plt.show()