## 实验二 《数据统计和可视化》

**组长：卢学文**

**组员：卢学文、 夏旭、 李鸿洋**

#### 题目

基于**实验一**中清洗后的数据练习统计和视化操作，100个同学（样本），每个同学有11门课程的成绩（11维的向量）；那么构成了一个100x11的数据矩阵。以你擅长的语言C/C++/Java/Python/Matlab，编程计算：

1. 请以课程1成绩为x轴，体能成绩为y轴，画出散点图。
2. 以5分为间隔，画出课程1的成绩直方图。
3. 对每门成绩进行z-score归一化，得到归一化的数据矩阵。
4. 计算出100x100的相关矩阵，并可视化出混淆矩阵。（为避免歧义，这里“协相关矩阵”进一步细化更正为100x100的相关矩阵，100为学生样本数目，视实际情况而定）
5. 根据相关矩阵，找到距离每个样本最近的三个样本，得到100x3的矩阵（每一行为对应三个样本的ID）输出到txt文件中，以\t,\n间隔。

#### 提示：

计算部分不能调用库函数；画图/可视化显示可可视化工具或API实现。

### 数据来源：

源于实验一所保存的result1.txt文件，直接在python中定义函数调用

以下是实现代码：

import pandas as pd

import numpy as np

from math import sqrt

import copy

import matplotlib.pyplot as plt

import pickle

#读取txt数据（读取实验一保存的数据）

def readTxt():

title=['SNO','SNAME','SCITY','SGENDER','SHEIGHT','C1','C2','C3','C4','C5','C6','C7','C8','C9','C10','SCONSTITUTION']

txtData\_set = pd.read\_csv('result1.txt',sep=' ',header=None)

txtData = np.array(txtData\_set)

return txtData,title

def Max\_3\_fun(S\_coefficient,ignore):

re=[0,0,0]

rank=[0,0,0]

index=0

for i in S\_coefficient:

if ignore==index:

index += 1

continue

for j in range(0, 3):

if i > re[j]:

re.insert(j,i)

rank.insert(j,index)

break

index += 1

return rank[0],rank[1],rank[2]

#实验1用的计算，因为其中有很多数据会用到所以照搬了

def Caculate(txtData,title):

Snum=0

total=0

for i in txtData:

if i[title.index('SCITY')]=='Beijing':

Snum+=1

# if i[1]=='Maxine':

# print(1)

for j in range(title.index('C1'),title.index('C9')):

if True!=np.isnan(i[j]):

total+=i[j]

# if True==np.isnan(total):

# print(1)

ans1=total/Snum

Snum = 0

Gnum=0

for i in txtData:

if i[title.index('SCITY')] == 'Guangzhou' and i[title.index('C1')]>80 and i[title.index('C9')]>=9 and i[title.index('SGENDER')]=='BOY':

Snum+=1

ans2=Snum

G\_girl=txtData[txtData[:,title.index('SCITY')]=='Guangzhou',title.index('SCONSTITUTION')]

S\_girl=txtData[txtData[:,title.index('SCITY')]=='Shanghai',title.index('SCONSTITUTION')]

G\_score=0

S\_score=0

score={'bad':1,'general':2,'good':4,'excellent':8}

for i in G\_girl:

G\_score +=score[i]

Gnum+=1

for i in S\_girl:

S\_score +=score[i]

Snum += 1

if S\_score/Snum>G\_score/Gnum:

ans3='Shanghai'

else:

ans3='Guangzhou'

Ax={5:[],6:[],7:[],8:[],9:[],10:[],11:[],12:[],13:[]}

Ax\_sum = {5: 0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0}

B\_sum = 0

Bx=[]

Snum=0

for i in txtData:

for x in range(title.index('C1'),title.index('C10')):

if np.isnan(i[x]) == True:

i[x]=68

Ax\_sum[x]+=i[x]

Ax[x].append(i[x])

if type(i[title.index('SCONSTITUTION')])==str:

B\_sum+=score[i[title.index('SCONSTITUTION')]]

Bx.append(score[i[title.index('SCONSTITUTION')]])

else:

B\_sum +=0

Bx.append(3)

Snum+=1

#这里对空值的处理是，替换成不计空值成员前的平均值取整（通过excel计算）

#替换后再重新计算平均值和方差等数值

A\_mean= {5: 0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0}

for k in Ax\_sum:

A\_mean[k]=(Ax\_sum[k]/Snum)

B\_mean =B\_sum/Snum

A\_variance={5: 0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0}

B\_variance=0

for i in Ax:

for j in Ax[i]:

A\_variance[i]+=(j-A\_mean[i])\*\*2

A\_variance[i]/=(Snum)

for i in Bx:

B\_variance+=(i-B\_mean)\*\*2

B\_variance/=(Snum)

coefficient={'C1': 0, 'C2':0, 'C3':0, 'C4':0, 'C5':0, 'C6':0, 'C7':0, 'C8':0, 'C9':0}

Cx=copy.copy(Ax)

Cx[14]=Bx

for i in Ax:

for z,j in zip(Ax[i],Bx):

coefficient[title[i]]+=((z-A\_mean[i])/sqrt(A\_variance[i]))\*((j-B\_mean)/sqrt(B\_variance))

ans4=coefficient

return Ax,Bx,Cx,A\_mean,B\_mean,A\_variance,B\_variance

def experiment(Ax,Bx,Cx,A\_mean,B\_mean,A\_variance,B\_variance):

Z\_Ax = {5: [], 6: [], 7: [], 8: [], 9: [], 10: [], 11: [], 12: [], 13: []}

Z\_Bx = []

# 实验二第1题

plt.scatter(Ax[5], Bx, marker='o')

plt.show()

# 实验二第2题

bin = [zx \* 5 for zx in range(0, 21)]

plt.hist(x=Ax[5], # 指定绘图数据

bins=bin,

edgecolor='black', # 指定直方图的边框色

align='mid',

)

plt.show()

# 实验二第3题

for i in Ax:

for j in Ax[i]:

Z\_Ax[i].append((j - A\_mean[i]) / sqrt(A\_variance[i]))

for i in Bx:

Z\_Bx.append((i - B\_mean) / sqrt(B\_variance))

Z\_Ax[14] = Z\_Bx

print()

orderedNames = [5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

dataMatrix = np.array([Z\_Ax[i] for i in orderedNames])

print("实验二第3题:",dataMatrix)

#将本题答案保存一下以供实验三使用

pickle\_file = open('z-scoreData.txt','wb')#必须是wb

pickle.dump(dataMatrix,pickle\_file)

pickle\_file.close()

# 实验二第4题 相关矩阵

S\_variance = []

S\_mean = []

for i in range(0, len(Ax[5])):

Mid = 0

for j in Cx:

Mid += Cx[j][i]

S\_mean.append(Mid / 10)

Mid = 0

for j in Cx:

Mid += (Cx[j][i] - S\_mean[i]) \*\* 2

S\_variance.append(Mid)

s = (len(Ax[5]), len(Ax[5]))

S\_coefficient = np.zeros(s)

for i in range(0, len(Ax[5])):

for j in range(0, len(Ax[5])):

for z in Cx:

S\_coefficient[i][j] += ((Cx[z][i] - S\_mean[i]) / sqrt(S\_variance[i])) \* (

(Cx[z][j] - S\_mean[j]) / sqrt(S\_variance[j]))

print("实验二第4题:",S\_coefficient)

# 实验二第5题

s = (len(Ax[5]), 3)

Max\_3 = np.zeros(s)

for i in range(0, len(Ax[5])):

Max\_3[i][0], Max\_3[i][1], Max\_3[i][2] = Max\_3\_fun(S\_coefficient[i], i)

Max\_3 = np.array(Max\_3)

np.savetxt('result2.txt', Max\_3, fmt='%d', delimiter="\t")

print("实验二第5题:",Max\_3)

def main():

txtData,title=readTxt()

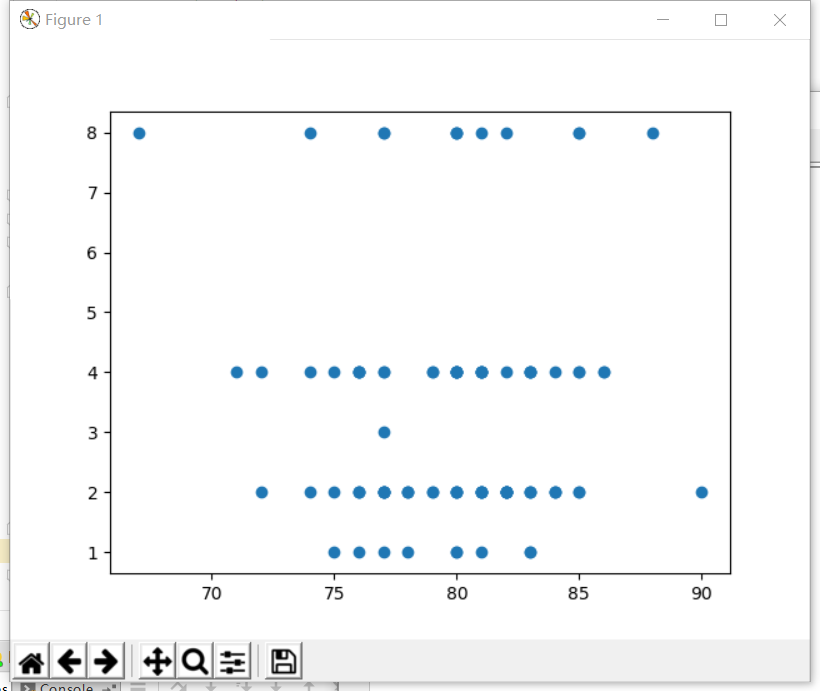
Ax,Bx,Cx,A\_mean,B\_mean,A\_variance,B\_variance=Caculate(txtData,title)

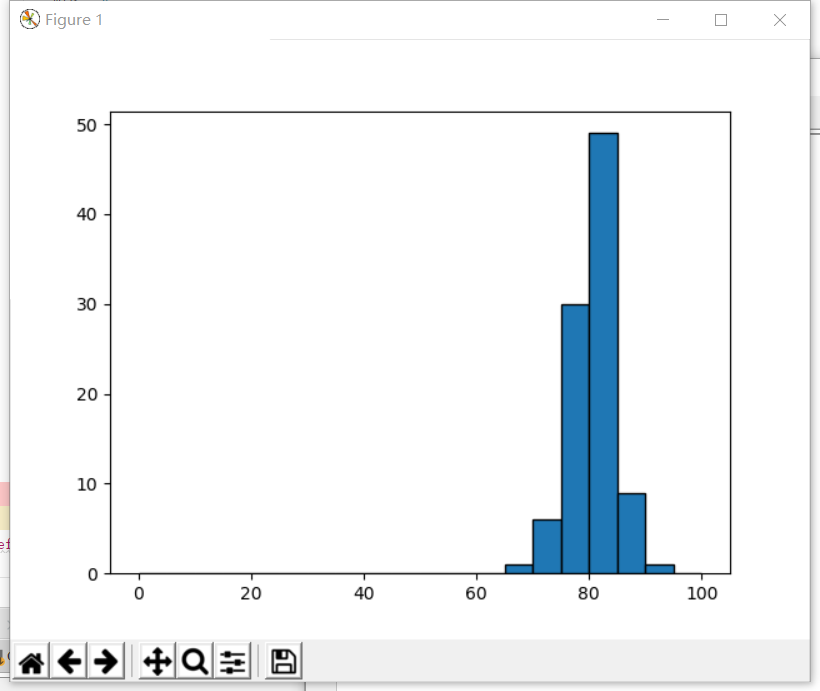
experiment(Ax,Bx,Cx,A\_mean,B\_mean,A\_variance,B\_variance)

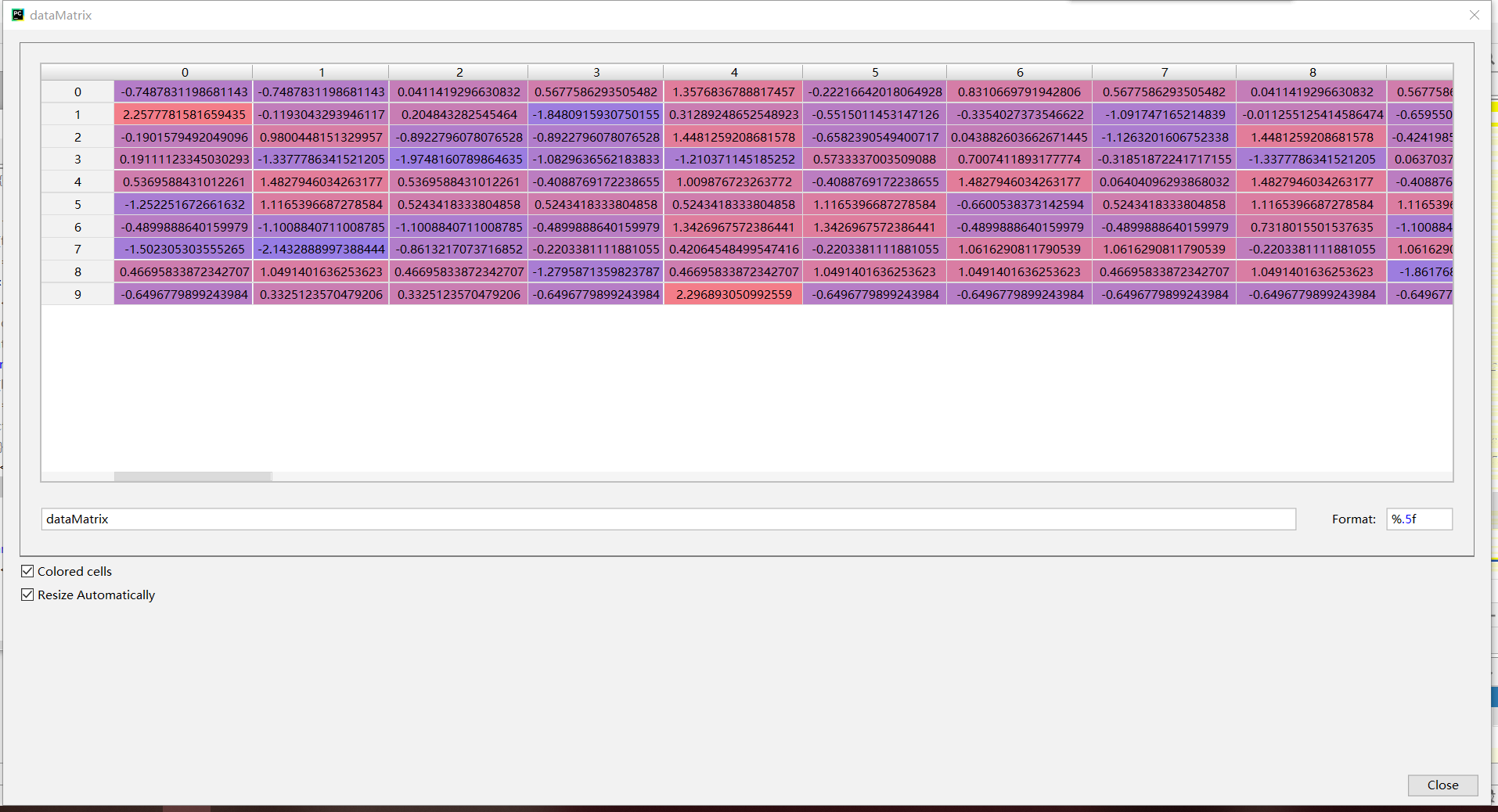
if \_\_name\_\_ == '\_\_main\_\_':

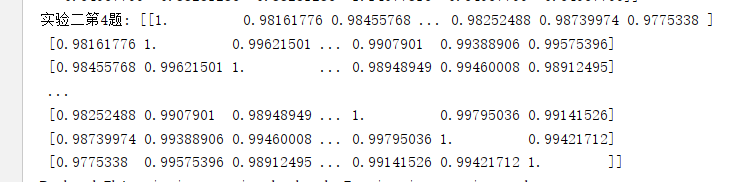
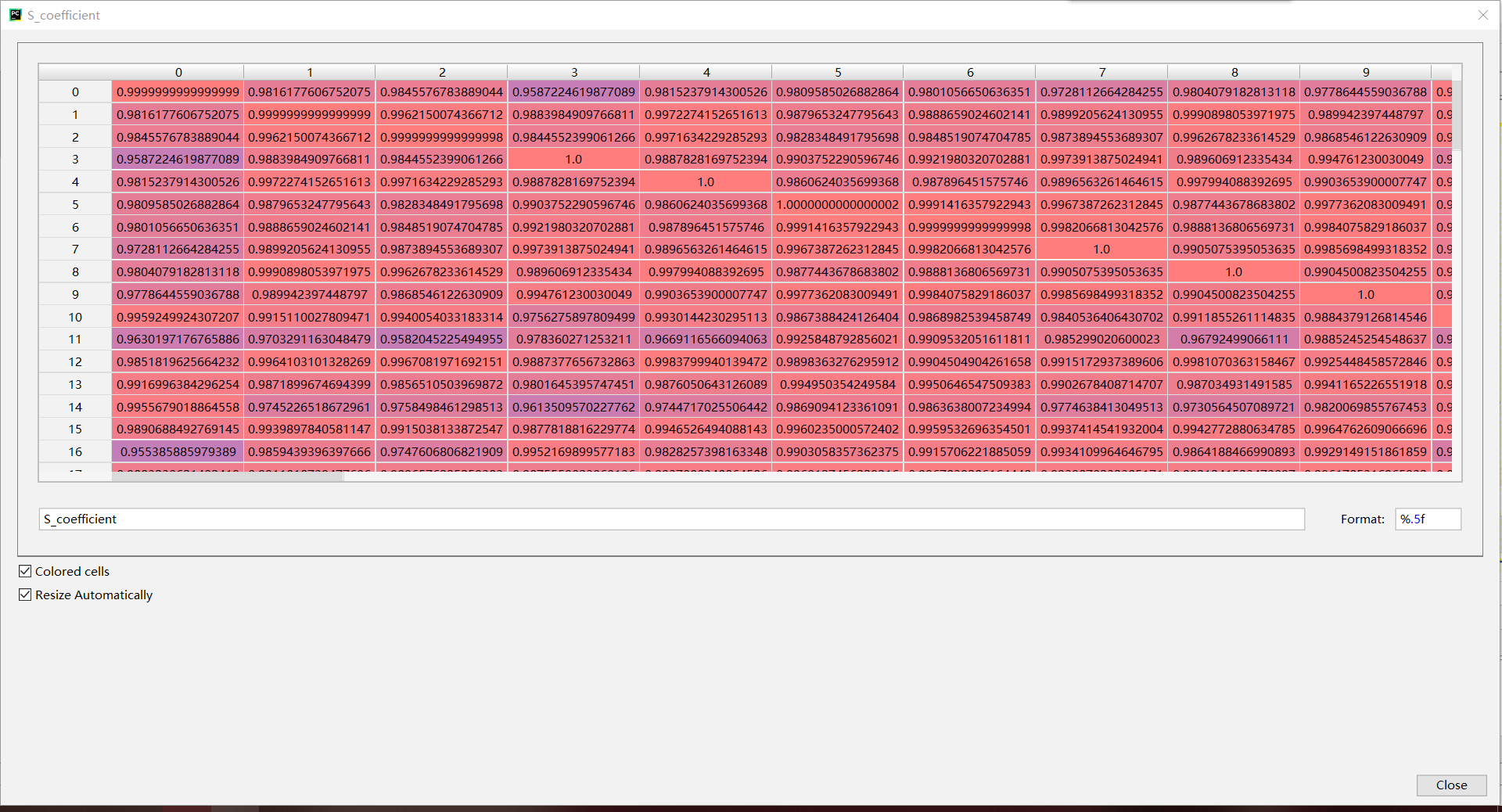
main()

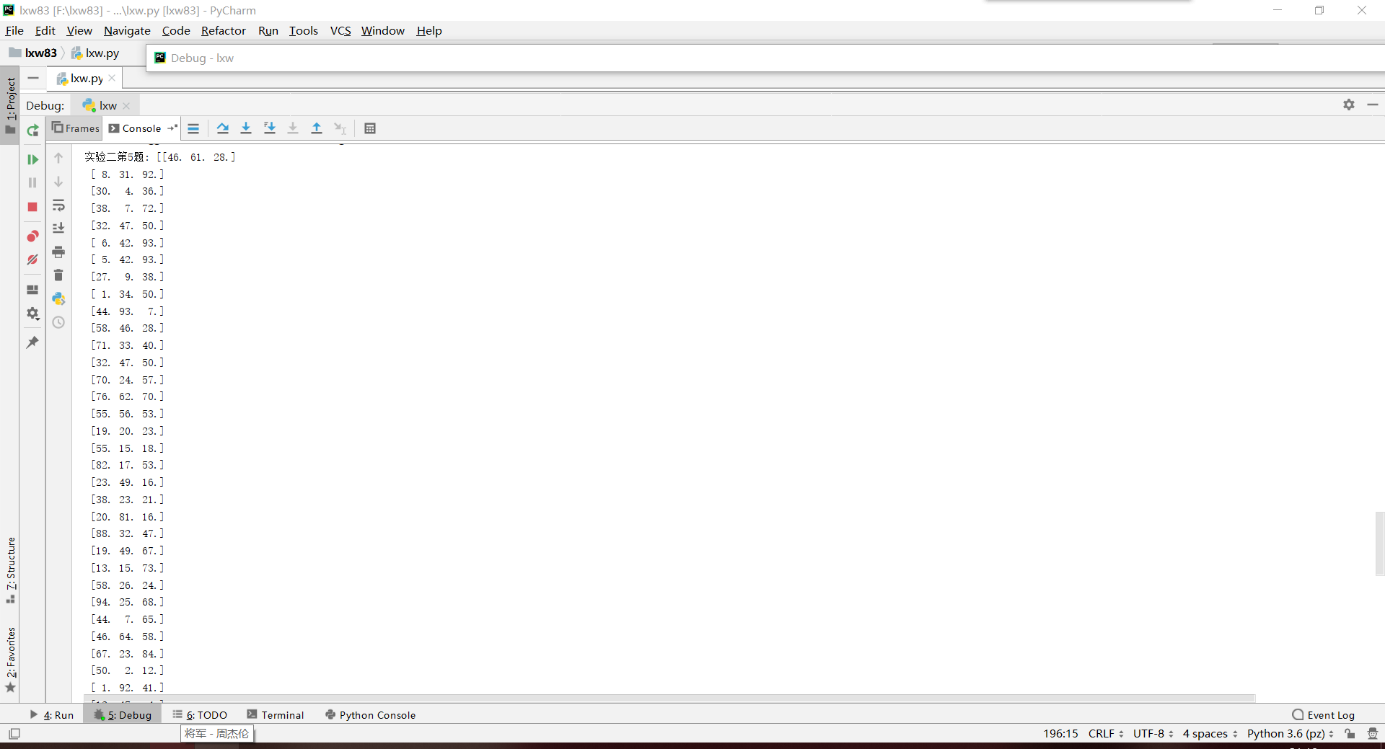
结果图如下：

第一题：以课程1成绩为x轴，体能成绩为y轴，画出散点图。

第二题：以5分为间隔，画出课程1的成绩直方图

第三题：对每门成绩进行z-score归一化，得到归一化的数据矩阵

第四题：计算出100x100的相关矩阵，并可视化出混淆矩阵。

第五题：根据相关矩阵，找到距离每个样本最近的三个样本，得到100x3的矩阵（每一行为对应三个样本的ID）输出到txt文件中，以\t,\n间隔

（结果保存在result2.txt中）