**一、实验目的及要求**

1. 掌握对多维数据统计、数据立方的理解

2. 掌握基本数据归一化、散点可视化、分布可视化方法

3. 按照既定格式书写实验报告

**二、实验设备与平台**

1. 实验设备：计算机；

2. 平台：Windows 10操作系统

**三、实验内容**

**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，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

**实验实施**：

（在此详述平台，技术栈，思路，处理逻辑等等）

平台：Windows10

技术栈：python

思路：直接读取实验一中处理后的数据，通过Jfreechart类包绘制要求的图表，

并根据题目要求计算得出各种矩阵。

处理逻辑：先读取，再绘图，后计算。

**实验一二三代码：**

import pandas as pd

import matplotlib.pyplot as plt

from 实验三四问 import cons2num

import numpy as np

from matplotlib.pyplot import MultipleLocator

df=pd.read\_csv('合并数据.csv',encoding='gbk')

# print(df.head())

C1\_arr=df['C1'].tolist()

Cons\_arr=df['Constitution'].apply(cons2num).tolist()

Cons\_xticks=np.arange(0,105,5)

# print(C1\_arr)

# print(Cons\_arr)

# print(Cons\_xticks)

fig=plt.figure(figsize=(20,10),num='C1与体能成绩散点图')

ax1=fig.add\_subplot(1,2,1)

ax2=fig.add\_subplot(1,2,2)

ax2\_major\_locator = MultipleLocator(5)

ax1.scatter(C1\_arr,Cons\_arr,color='red',s=30)

# ax2.set\_xlim([0,100])

max\_value = max(C1\_arr)

min\_value = min(C1\_arr)

num\_bin = int((max\_value-min\_value)//5)

ax2.set\_xlim([0,100])

ax2.xaxis.set\_major\_locator(ax2\_major\_locator)

ax2.hist(C1\_arr,bins=num\_bin)

# ax2.set\_xlim([0,100])

# ax2.xaxis.set\_major\_locator(ax2\_major\_locator)

plt.show()

df['Constitution']=df['Constitution'].apply(cons2num)

def z\_score(z\_score\_list,df):

for i in z\_score\_list:

mean\_value=df[i].mean()

s\_value=0

print(mean\_value)

for j in df[i]:

s\_value+=(j-mean\_value)\*\*2

print(s\_value)

s\_value=pow((s\_value/df[i].count()),0.5)

print(s\_value)

df[i]=df[i].apply(lambda x:(x-mean\_value)/s\_value)

print(df[i])

z\_score\_list=['C1','C2','C3','C4','C5','C6','C7','C8','C9','C10','Constitution']

z\_score(z\_score\_list,df)

# df.to\_csv('归一化的数据.csv')

**实验四五代码：**

import pandas as pd

import numpy as np

np.set\_printoptions(threshold=np.inf) #矩阵全部显示

ori\_df=pd.read\_csv('归一化的数据.csv')

for col in ori\_df.columns:

ori\_df[col]=ori\_df[col].apply(lambda x:ori\_df[col].mean() if x==0 else x)

ori\_df['C10']=ori\_df['Constitution']

zero\_df=ori\_df.iloc[:,7:17]

# zero\_df.to\_csv('归一化的成绩.csv')

print(zero\_df.shape[0])

def r\_matrix(df):

matrix\_r=np.empty(shape=(df.shape[0],df.shape[0]))

for i in range(df.shape[0]):

i\_mean=df.iloc[i,:].mean() #求所求行的平均数

for j in range(df.shape[0]):

j\_mean=df.iloc[j,:].mean()#求其余行平均数（包括所求行）

X=df.iloc[i,:].apply(lambda x:(x-i\_mean))

Y=df.iloc[j,:].apply(lambda x:(x-j\_mean))

COVX\_Y=(X\*Y).sum(axis=0)

VARX\_Y=pow((pow(X,2).sum()\*pow(Y,2).sum()),0.5)

matrix\_r[i][j]=round(COVX\_Y/VARX\_Y,3)

return matrix\_r

# for col in col\_list:

matrix\_r=r\_matrix(zero\_df)

# np.savetxt('matrix\_r.txt',matrix\_r) # 保存矩阵

print(matrix\_r)

# print(matrix\_r.shape)

def rel\_ID(matrix\_r):

matrix=np.empty(shape=(matrix\_r.shape[0],4))

matrix=matrix.astype(np.str)

matrix\_r\_index = np.argsort(-matrix\_r,axis=1)

for i in range(matrix\_r\_index.shape[0]):

for j in range(4):

index=matrix\_r\_index[i][j]

matrix[i][j]=ori\_df.loc[index,'ID']

return matrix

rel\_ID\_matrix=rel\_ID(matrix\_r)

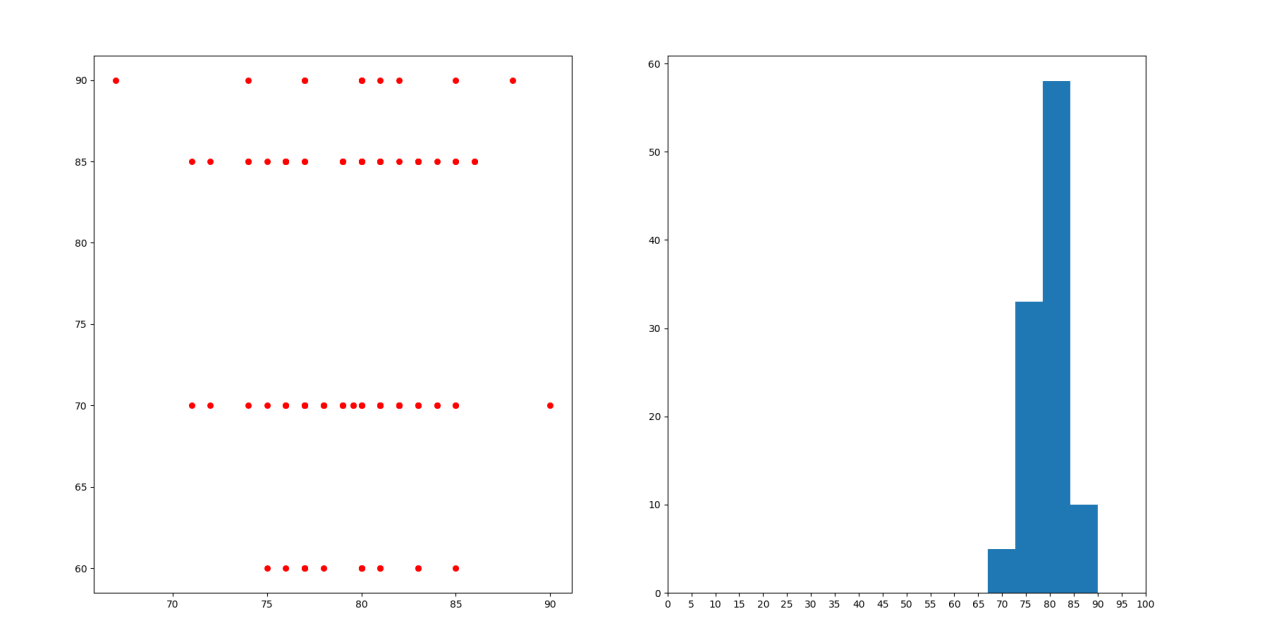
np.savetxt('rel\_ID\_matrix.txt',rel\_ID\_matrix,fmt='%s',delimiter='\t')

print(rel\_ID\_matrix)

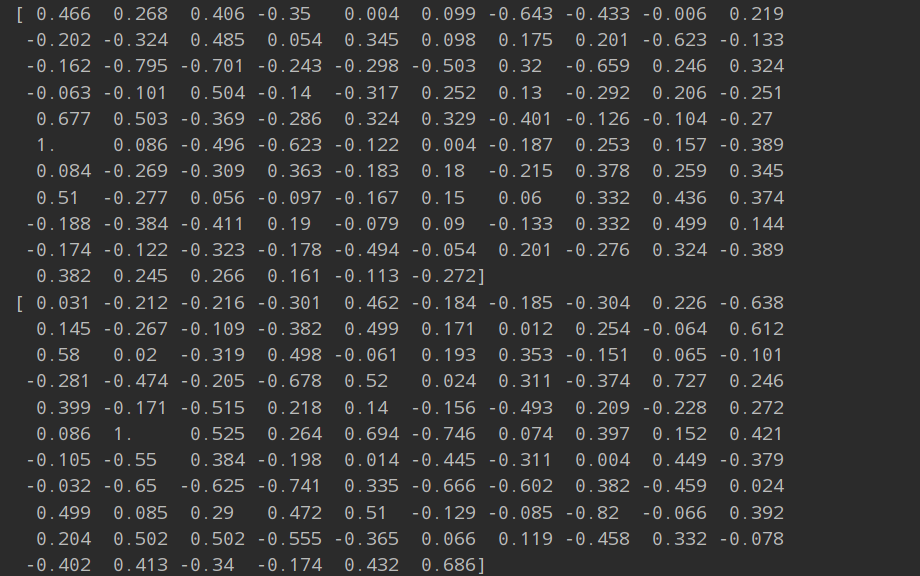
**运行结果**：

（运行结果、现象的说明与截图）

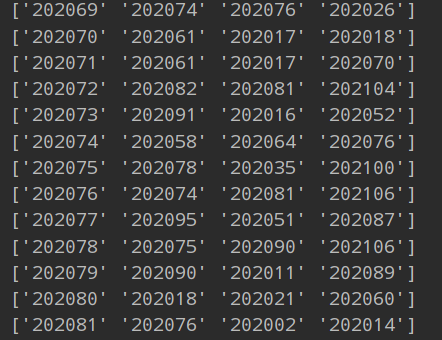
1、散点图与直方图



2、相关矩阵



3、距离每个样品最近的三个样品，第一列为检测



**实验总结**：

（遇到的问题，有什么感想和收获等等）

通过使用Jfreechart绘图，对这个包有了一定的了解，并且掌握了对各种矩阵的运算与对数据的处理。