**实验5 Python科学计算实践**

**姓名： 徐超信学号：1935010102 班级：计科1902成绩：**

**1. 实验内容与目标**

本次实验旨在了解和掌握python科学计算工具，包括numpy和scipy的数值计算、pandas数据分析、matplotlib画图等，具备快速高效的数据处理和一定的可视化分析能力。需掌握的相关知识点主要包括：

1) numpy数组array的创建、使用，数组元素的切片，各种常见函数、通用函数（universal function）的运用，np.random模块组织和获取随机数据，以及numpy对文件数据的处理方法np.loadtxt()和np.savetxt()。了解微积分、线性代数、最优化等领域相关的scipy科学计算方法；

2) pandas的数据结构：Series和DataFrame，以及相关数据探索分析方法;

3) matplotlib画图：折线图、散点图、柱状图、直方图、饼图，以及多子图布局方式。（注：请自学matplotlib绘图库，还可以包括*seaborn*）

请完成下列实验练习题，报告书写要求同第1次实验作业。注意，请在本报告中将自己的姓名、学号、班级书写正确，文件名按以下格式写出具体的班级、学号、姓名，样例如下：

“软件1902-1912190001-张三-实验5\_Python科学计算实践.docx”

最后，请在规定时间内完成本次实验，并将报告的word文档(非pdf版本)以附件形式提交至学习通平台。

**2. 实验练习题**

**答题要求**：将正确运行的**源代码书写在题目下方**，并紧跟着贴上一份**源代码的截图**和运行结果的**截图**。答题格式同第1次实验作业。

## (1) 数组元素索引。利用numpy.random模块随机产生一个9行10列的二维数组，其元素要求服从标准正态分布。完成以下该数组的索引任务，打印输出相关结果：

(a) 使用索引方式获取第2行第5列、第6行第3列的元素。

(b) 使用切片方式获取第3行至第5行和第4列至第6列的数据。

(c) 使用切片与整数序列索引混合的方式，获取第3行至第5行且为第1列、第2列和第4列的数据。

(d) 使用布尔索引方法将数组中取值大于1的元素重新赋值为10.00，小于-1的元素重新赋值为-10.00，并打印输出这个新数组。

源代码：

import numpy as np

from numpy import random

a1=random.randn(9,10)

print(a1)

print(a1[2,5])

print(a1[6,3])

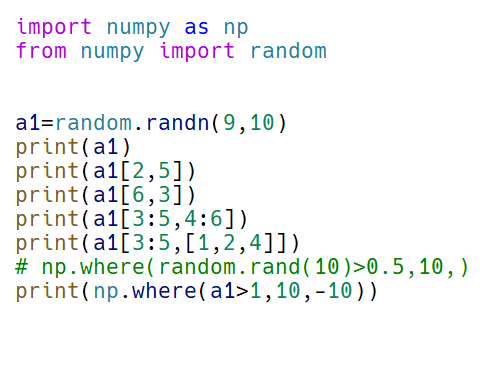
print(a1[3:5,4:6])

print(a1[3:5,[1,2,4]])

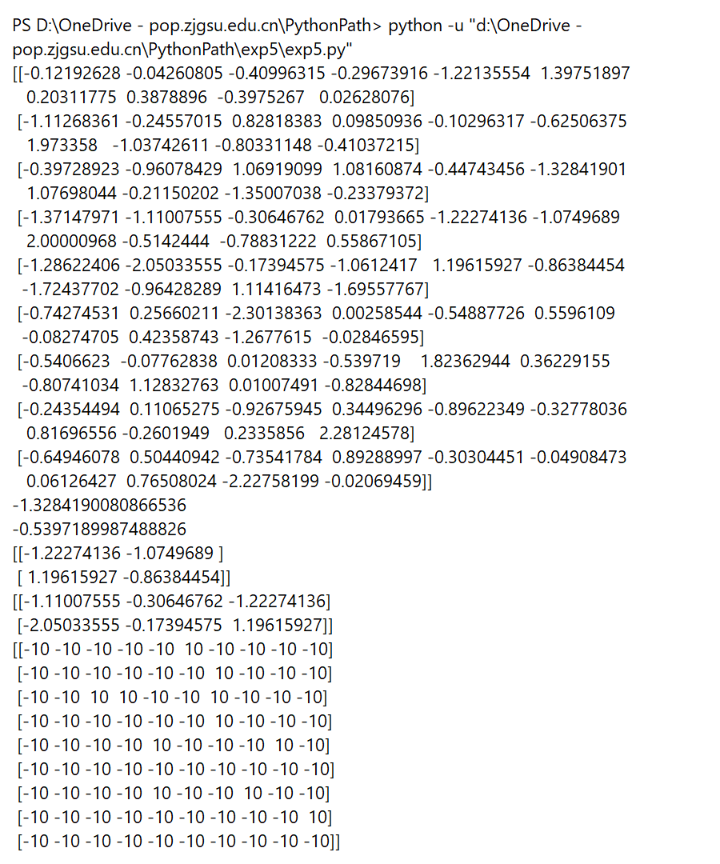
# np.where(random.rand(10)>0.5,10,)

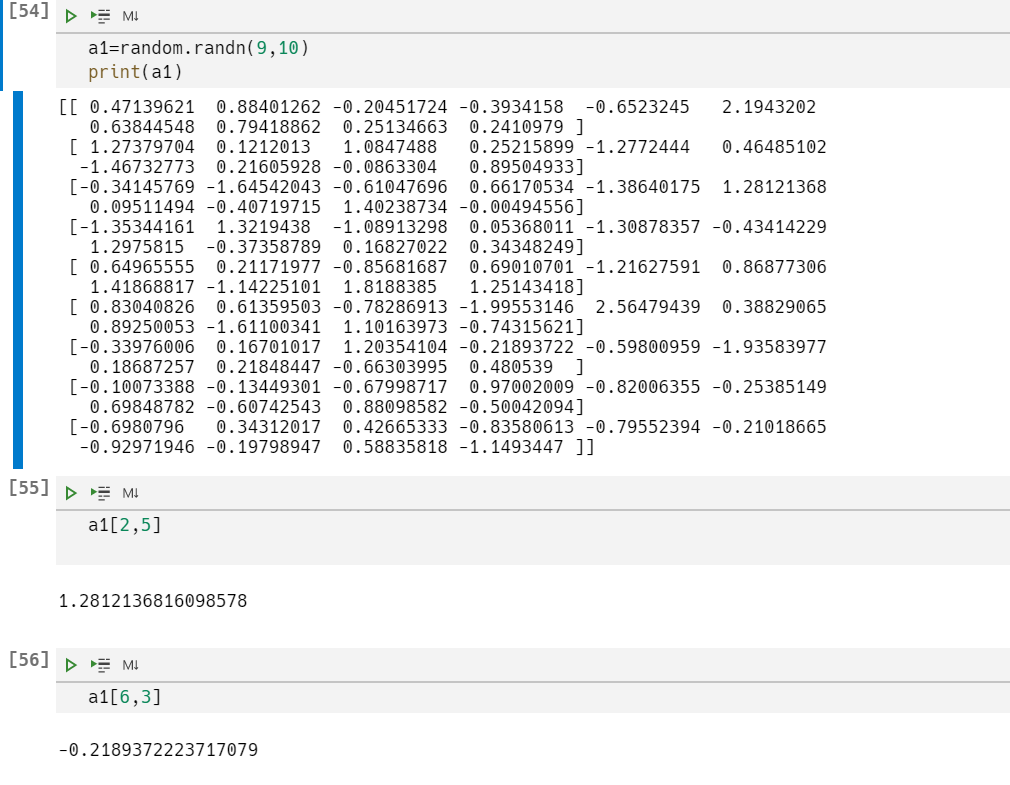
print(np.where(a1>1,10,-10))

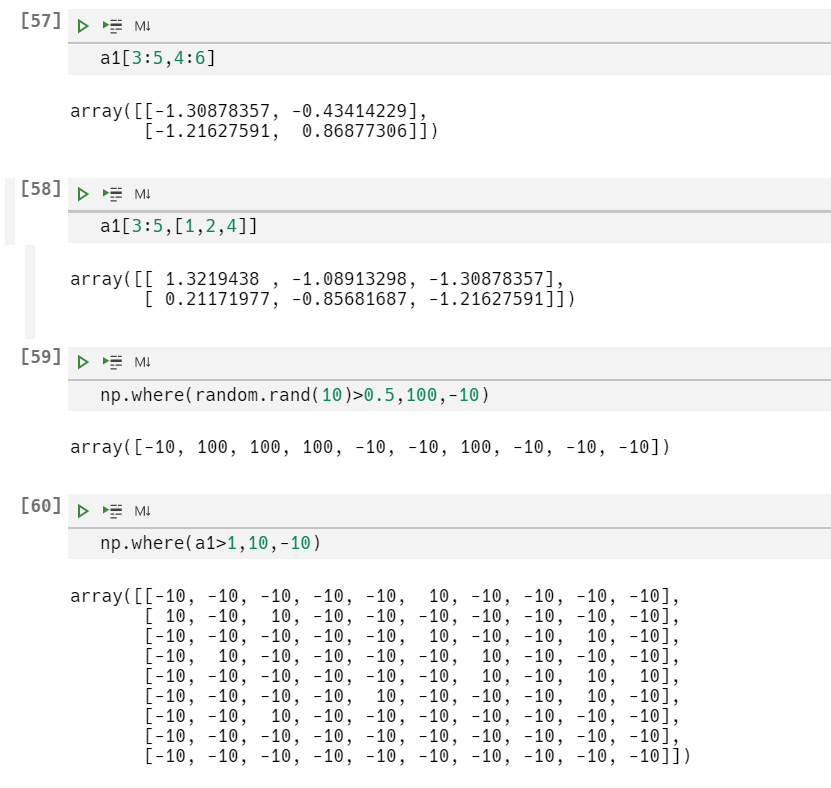
源代码截图：



运行结果截图：







## (2)利用numpy中相关模块，随机产生10000个服从 [-1,1]区间上均匀分布的实数，同时随机产生10000个服从标准正态分布的实数，由此获得两个一维数组，分别记为A和B。请计算下列式子的值：

(a) A+B

(b) A和B的乘积

(c) A/B

(d) exp(A)+exp(B)

(e) A和B的内积

(f)数组B全体元素的平均值，最大值，最小值。

源代码：

A=random.uniform(-1,1,10000)

# print(A)

B=random.normal(size=10000)

print(A+B)

print("A@B=",A\*B)

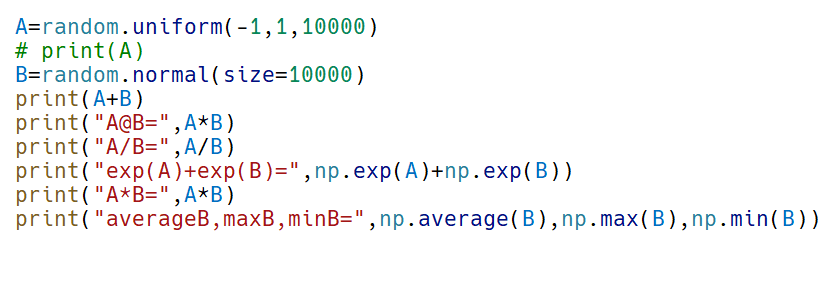
print("A/B=",A/B)

print("exp(A)+exp(B)=",np.exp(A)+np.exp(B))

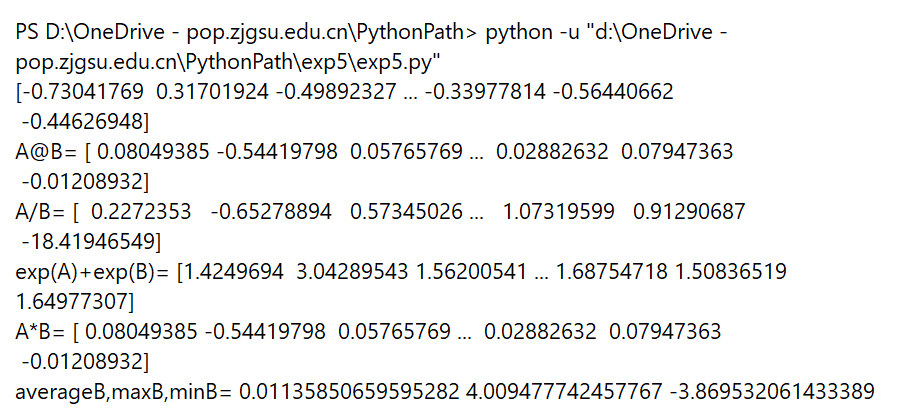
print("A\*B=",A\*B)

print("averageB,maxB,minB=",np.average(B),np.max(B),np.min(B))

源代码截图：



运行结果截图：



## (3) 利用numpy.random模块，服从区间[10., 20.]上的均匀分布，随机生成一个10行5列的矩阵(二维数组)。通过Python编程依序完成下列任务，打印输出原数组和产生的新数组：

(a)矩阵的每一行的元素都减去该行的平均值。

(b)利用numpy.savetxt()，将经过(a)操作变换后的新数组数据保存至文本文件”dat.csv”, 保持10行5列不变，并要求小数点后保留3位，两个数据之间用逗号隔开。

(c)利用numpy.loadtxt()从“dat.csv”读取数组数据，原地交换当前数组的第一行和第二行数据。

(d)要求按第2列数据对当前数组进行排序。例如，原数组为

[[1 7 9]

[7 8 1]

[8 4 2]]

若按第2列进行排序，结果如下：

[[8 4 2]

[1 7 9]

[7 8 1]]

源代码：

import numpy as np

from numpy import random

""" 3 """

a=random.uniform(10,20,(10,5))

print(a)

average\_array=np.mean(a,axis=1)

average\_matrix=average\_array.reshape(average\_array.shape[0],1)

np.savetxt("dat.csv",a,fmt="%.3f",delimiter=',')

a[[1,2],:]=np.loadtxt("dat.csv",delimiter=",")[[1,2],:]

index\_col=a[:,1]

indices\_col=np.argsort(index\_col)

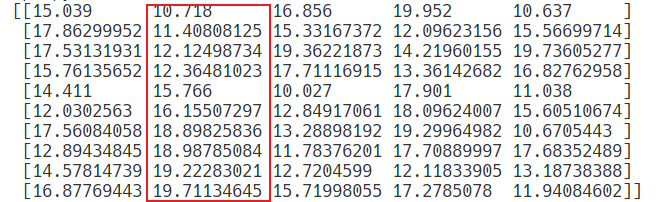
a=a[indices\_col,:]

print(a)

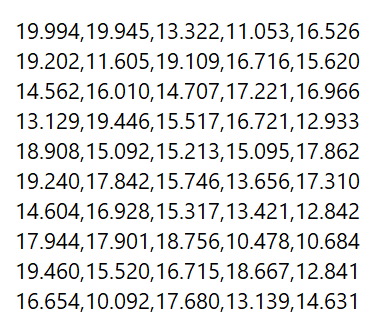
源代码截图：



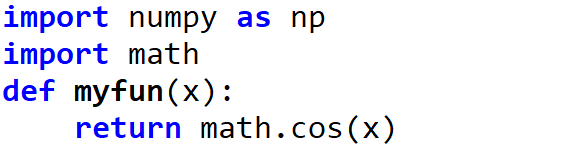
运行结果截图：



文件(dat.csv)内容截图：



## (4)假设已有下列代码：



显而易见，函数myfun()还不是一个ufunc函数，请定义一个新函数umyfun()，使得 myfun成为一个ufunc函数。进一步，分别记录umyfun()和np.cos() 在计算1~100000之间共计10万个正整数的余弦值时所需的时间，比较两者的计算速度。

源代码：

import numpy as np

from numpy import random

import math

import time

""" 4 """

def myfun(x):

    return math.cos(x)

def umyfun():

    # def myfun(x):

    #     return math.cos(x)

    return np.frompyfunc(myfun, 1, 1)

# array = random.randint(1, 100000, size=100000)

array=np.arange(1,100000)

# print(umyfun()(np.array([math.pi,0])))

start\_time=time.time()

result = umyfun()(array)

print(result)

print("umyfun() used time:",time.time()-start\_time)

start\_time2=time.time()

result2=np.cos(array)

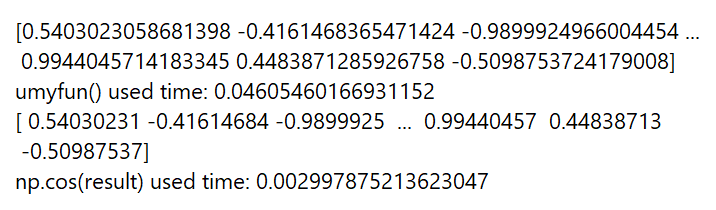
print(result2)

print("np.cos(result) used time:",time.time()-start\_time2)

源代码截图：



运行结果截图：



## (5)利用scipy.integrate模块计算函数在区间[1,2]上的定积分。

源代码：

import numpy as np

from numpy import random

import math

import time

import scipy

from scipy import integrate

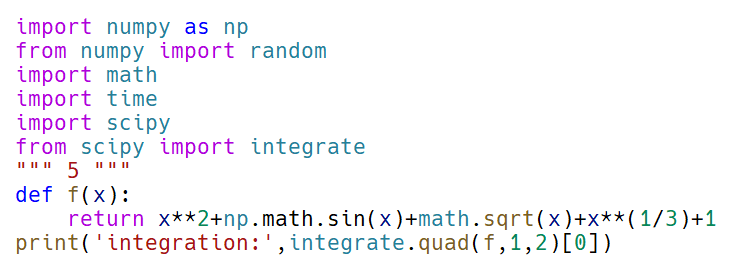
""" 5 """

def f(x):

    return x\*\*2+np.math.sin(x)+math.sqrt(x)+x\*\*(1/3)+1

print('integration:',integrate.quad(f,1,2)[0])

源代码截图：



运行结果截图：



## (6) 数据探索分析是指对数据进行缺失值、异常值查找以及统计分析，分析出数据的规律以及异常值。保存在文件“air\_data.csv”中的数据为某航空公司的大量会员档案信息及其乘坐航班记录。通过观察可以发现，原始数据中存在票价为空值，票价最小值为0、折扣率最小值为0、总飞行公里数大于0的记录。票价为空值的数据可能是客户不存在乘机记录造成，其它的数据可能是客户乘坐0折机票或者积分兑换产生的。

程序代码” data\_explore.py”旨在考察每个属性的观测值中空值个数、最大值和最小值。请读懂程序的每行代码，学习与领会pandas库中数据框(dataframe)的基本用法。

运行这个程序，并查看运行结果，然后小幅修改代码，完成填写下列表格的空白处。

**表1 数据探索分析结果表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 空值记录数 | 最大值 | 最小值 | 均值 | 标准差 |
| SUM\_YR\_1 | 3 | 239560 | 0 | 15553.4538919187 | 13165.5565269904 |
| SUM\_YR\_2 | 0 | 234188 | 0 | 17198.6145394465 | 13716.0046000281 |
| LAST\_TO\_END | 0 | 696 | 1 | 54.69855432 | 81.42268 |
| FLIGHT\_COUNT | 0 | 213 | 2 | 32.00611318 | 19.95965 |
| SEG\_KM\_SUM | 0 | 558440.1 | 18974.79 | 38751.78245 | 26052.37 |
| AVG\_DISCOUNT | 0 | 1.5 | 0.353298 | 0.797543454 | 0.191168 |

源代码：

# -\*- coding: utf-8 -\*-

# 对数据进行基本的探索

# 返回缺失值个数以及最大最小值

from re import T

from openpyxl import Workbook

from openpyxl.utils.dataframe import dataframe\_to\_rows

import pandas as pd

prefix = "./exp5/"

datafile = 'air\_data.csv'  # 航空原始数据,第一行为属性标签

resultfile = 'explore\_result.xls'  # 数据探索结果表

# 读取原始数据，指定UTF-8编码（需要用文本编辑器将数据装换为UTF-8编码）

data\_table = pd.read\_csv(prefix + datafile, encoding='utf-8')

# print(data\_table)

""" Returns

    DataFrame or TextParser

    A comma-separated values (csv) file is returned as two-dimensional data structure with labeled axes. """

df\_described = data\_table.describe(percentiles=[

    0.75], include='all')

# print(df\_described)

# 包括对数据的基本描述，percentiles参数是指定计算多少的分位数表（如1/4分位数、(1/2分位数)中位数等）；T是转置，转置后更方便查阅;include :要显示的数据类型对应的数据列

df\_described\_T = df\_described.T

print(df\_described\_T)

'''

    DataFrame.count

    Count number of non-NA/null observations.

    DataFrame.max

    Maximum of the values in the object.

    DataFrame.min

    Minimum of the values in the object.

    DataFrame.mean

    Mean of the values.

    DataFrame.std

    Standard deviation of the observations.

    DataFrame.select\_dtypes

    Subset of a DataFrame including/excluding columns based on their dtype. '''

# print("len(data\_table)")

# print(len(data\_table))

# print("df\_described['count']")

# print(df\_described\_T["count"])

# print(len(data\_table)-df\_described\_T['count'])

# describe()函数自动计算非空值数，空值数需自己动手计算;df\_described['null']将为df\_described增加一列null列

df\_described\_T['null'] = len(data\_table)-df\_described\_T['count']

df\_described\_T['standard deviation'] = data\_table.std()

print(df\_described\_T)

''' get the sepecified colums :(use a list contains column names) '''

# df\_described\_3 = df\_described[['null', 'max', 'min']]

df\_described\_5 = df\_described\_T[['null', 'max', 'min','mean', 'std']]

# print(df\_described\_T)

# df\_described\_3.columns = [u'空值数', u'最大值', u'最小值']  # 表头重命名

df\_described\_5.columns = [u'空值数', u'最大值', u'最小值', u'均值',u'标准差']

'''这里只选取部分探索结果。

describe()函数自动计算的字df = pd.DataFrame({'categorical': pd.Categorical(['d','e','f']),

                   'numeric': [1, 2, 3],

                   'object': ['a', 'b', 'c']

                  })段有count（非空值数）、unique（唯一值数）、top（频数最高者）、freq（最高频数）、mean（平均值）、std（标准差）、min（最小值）、50%（中位数）、max（最大值）'''

# explore\_table.to\_excel(prefix + resultfile)  # 导出结果

wb = Workbook()

ws = wb.active

# write the entries in the dataframe to the excel table

# for r in dataframe\_to\_rows(df\_described\_3, index=True, header=True):

#     ws.append(r)

for r in dataframe\_to\_rows(df\_described\_5, index=True, header=True):

    ws.append(r)

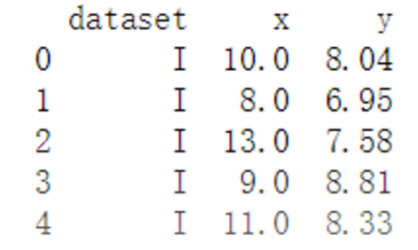
wb.save(prefix+resultfile)

# print(list(range(100)))

源代码截图：



## （7）利用pandas创建如下的数据框(DataFrame)，完成以下问题：



（a）将index为2和4的dataset值都改为II，并打印出修改后的数据框.

（b）分别计算dataset为I和II时，x和y的平均值。

（c）找出满足x>9.5的样本，并计算y值的平均值。

源代码：

import numpy as np

from numpy import random

from numpy.lib.function\_base import average

import pandas as pd

import math

import time

import scipy

from scipy import integrate

import matplotlib.pyplot as plt

''' 7 '''

''' df = pd.DataFrame([[1, 2], [4, 5], [7, 8]],

     index=['cobra', 'viper', 'sidewinder'],

     columns=['max\_speed', 'shield']) '''

data\_dict={'dataset':['I','I','I','I','I'],'x':[10,8,13,9,11],'y':[8.04,6.95,7.58,8.81,8.33]}

df=pd.DataFrame(data=data\_dict)

''' I found that if use numpy.array to make the dataFrame,the mean() method work uncorrectly!'''

# data\_lists=[['I','I','I','I','I'],[10,8,13,9,11],[8.04,6.95,7.58,8.81,8.33]]#the common python lists will be priority by line make the parameter for the constractor with specified columns; the numpy.array instance may be is more flexible:https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html?highlight=dataframe#pandas.DataFrame

# df=pd.DataFrame(data=np.array(data\_lists).T,columns=['dataset','x','y'])

# print(df)

df.loc[[2,4],'dataset']="II"

# tmp=df[[2,4],['dataset']]

print(df)

''' averages: '''

bool\_serial\_I=df['dataset']=='I'

# print(df['dataset']=='I')

items\_I=df.loc[bool\_serial\_I]

# print(tmp1.loc[:,['x','y']])

# tmp1=tmp1.loc[:,['x','y']]

# print(tmp1)

print(items\_I)

# print(type(items\_I))

# print(items\_I.mean(1))

average\_I=items\_I.mean(1)

# print(type(average\_I.values))#<class 'numpy.ndarray'>

# print(average\_I.values)

average\_I\_df=average\_I.to\_frame()

average\_I\_df.columns=["averages of x+y"]

print(average\_I\_df)

# print(type(average\_I))#<class 'pandas.core.series.Series'>

#note:the mean() there return a Serial rather than a dataFrame!,so the result just has the index tag but no columns!

# print("average of x+y for I:",average\_I[:])

# items\_I.columns=["index","averages"]

# print(items\_I.columns)

# print(items\_I.index)

bool\_serial\_lt=df["x"]>9.5

tmp2=df[bool\_serial\_lt]

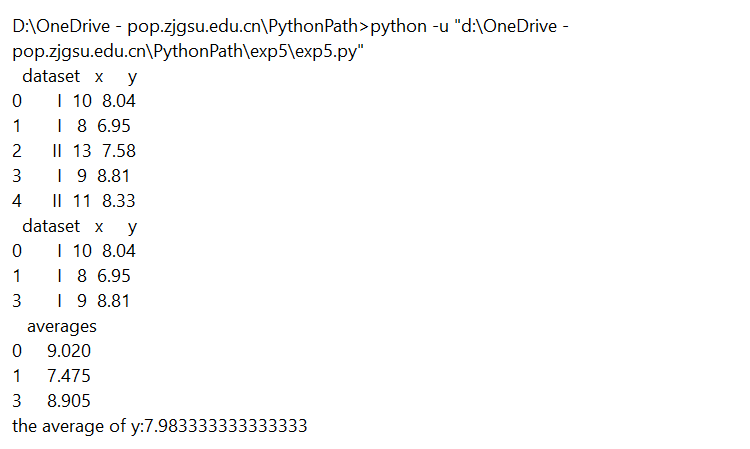
print("the average of y:",end="")

print(tmp2['y'].mean())

源代码截图：



运行结果截图：



## (8) 产生10000个随机数，服从均值为-1，标准差为1的正态分布N(-1,1)，利用matplotlib画出其直方图(100个bin)，即这10000个数的实际分布图。可参考源文件 “histgram\_plot.py”中的代码。

源代码：

import numpy as np

import matplotlib.pyplot as plt

''' to make the chart well,you must set well the graph with the matched axes range

as for the other information,such as the title, the labels of x,y axes;and the formula expression with the appropriate

location(axis value)

then is the color of the chart '''

mu,sigma=-1,1

array\_x=mu+sigma\*np.random.randn(10000)

# array\_x=np.random.normal(-1,1,10000)

n,bins,patches=plt.hist(array\_x,bins=100,density=1)

# print(bins)

# set the title and x,y lables of the chart:

plt.title('Histogram of normal attributions')

plt.xlabel('x')

plt.ylabel('y')

"""the text() fuction

    it support the latex syntax to express formula:r"$string#"

    DESCRIPTION

    Add text to the axes ['æksiːz].

    Add text in string s to axis at location x, y, data coordinates.

    PARAMETERS

    x, y : scalars

    data coordinates

    s : string

    text"""

# Add text in string s to axis at location x, y, data coordinates.

    #it must matched properly with the chart x,y axes range to display,or you couldn't see it in the chart!

plt.text(0,0, r'$\mu=-1,\ \sigma=1$')

""" Convenience method to get or set axis properties. """

#if you not specify the axis ,then the chart axes will auto adjust to display properly!

# plt.axis([-10,10,0,1])

''' set if you want to display the grid: '''

plt.grid(True)

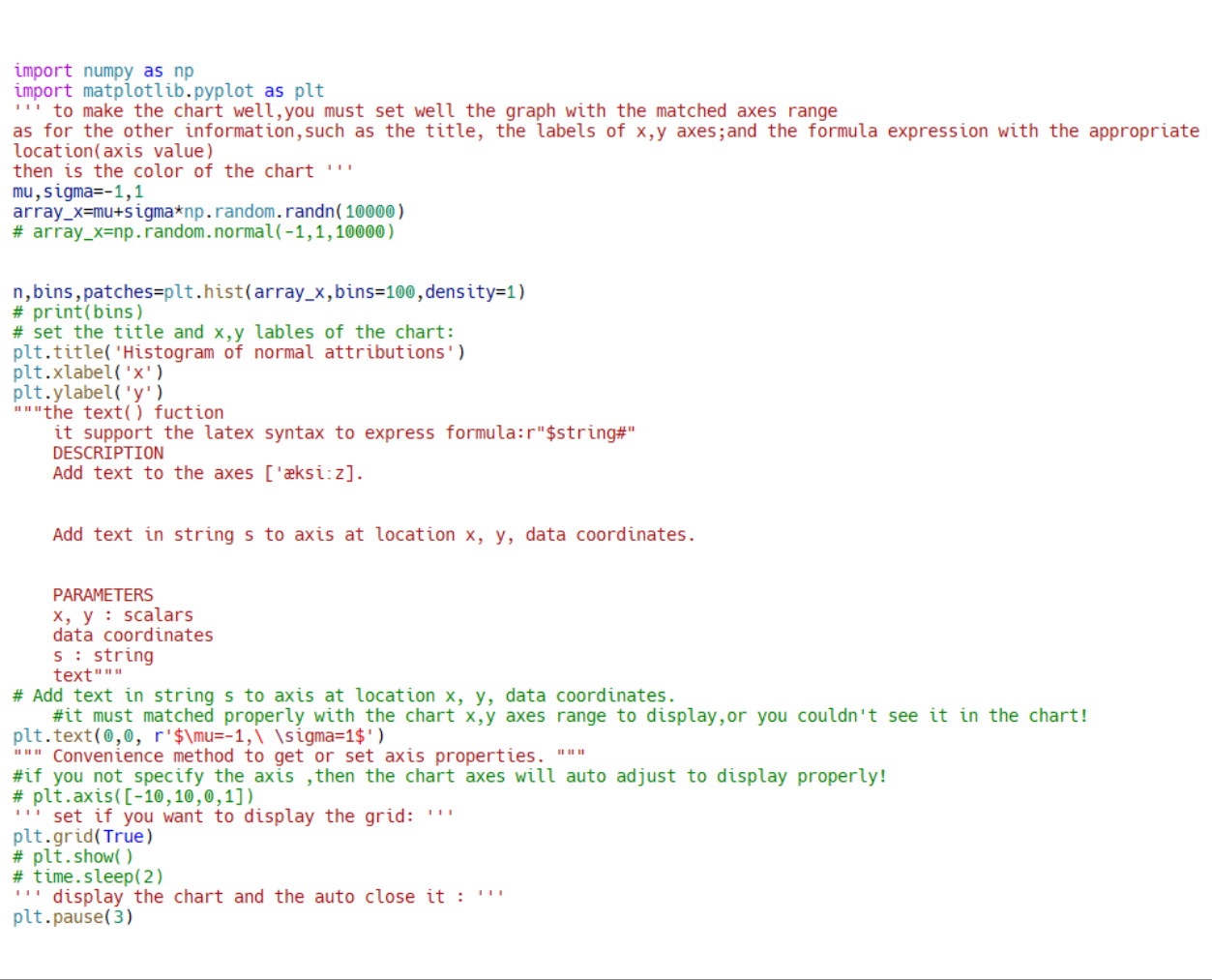
# plt.show()

# time.sleep(2)

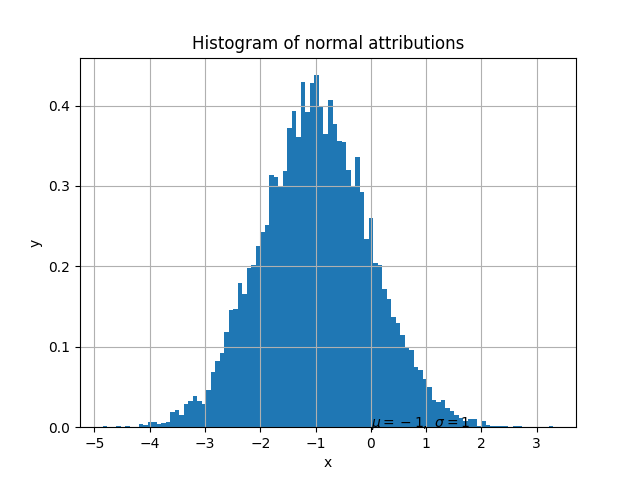
''' display the chart and the auto close it : '''

plt.pause(3)

源代码截图：



运行结果截图：



## (9) 画一张图，包含5幅子图，

## 布局为：第一行2幅，左边为柱状图（bar plot），右边为散点图(scatter plot)；

## 第二行2幅，左边为饼图（pie plot），右边为折线图（plot），第三行1幅子图，为直方图（histogram plot）。

## 相关数据请自行随机创建或从网上采集。

## 每幅子图要有相应的标题，表示什么类型的图。

## 可参考的源文件包括：“histgram\_plot.py”、“pie\_plot.py”和“multiple\_subplots.py”。

源代码：

# -\*- coding: utf-8 -\*-

"""

Created on Thu Nov 03 16:57:26 2016

@author: gsdx

"""

import matplotlib.pyplot as plt

import numpy as np

def f(t):

    return np.exp(-t) \* np.cos(2 \* np.pi \* t)

t1 = np.arange(0, 5, 0.1)

t2 = np.arange(0, 5, 0.02)

def bar\_plot():

    plt.subplot(321)

    k = 10

    x = np.arange(k)

    y = np.random.rand(k)

    plt.bar(x, y) # 画出 x 和 y 的柱状图

    # 增加数值

    for x, y in zip(x, y):

        plt.text(x, y , '%.2f' % y, ha='center', va='bottom')

def scatter\_plot():

    plt.subplot(322)

    # plt.plot(t2, np.cos(2 \* np.pi \* t2), 'r--')

    k = 500

    x = np.random.rand(k)

    y = np.random.rand(k)

    size = np.random.rand(k) \* 50 # 生成每个点的大小

    colour = np.arctan2(y, x) # 生成每个点的颜色大小

    plt.scatter(x, y, s=size, c=colour)

    plt.colorbar() # 添加颜色栏

# plt.figure(12)

def pie\_plot():

    plt.subplot(323)

    # plt.plot(t1, f(t1), 'bo', t2, f(t2), 'r--')

    labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'

    sizes = [15, 30, 45, 10]

    colors = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral']

    explode = (0, 0.1, 0, 0) #only "explode" the 2nd slice (i.e. 'Hogs')

    plt.pie(sizes, explode=explode, labels=labels, colors=colors,

            autopct='%1.1f%%', shadow=True, startangle=90)

    # Set aspect ratio to be equal so that pie is drawn as a circle.

    plt.axis('equal')

# plt.pause(5)

def line\_plot():

    plt.subplot(324)

    plt.plot([1, 2, 3, 4], [1, 4, 9, 16])

def histgram\_plot():

    plt.subplot(313)

    mu,sigma=-1,1

    array\_x=mu+sigma\*np.random.randn(10000)

    # array\_x=np.random.normal(-1,1,10000)

    plt.hist(array\_x,bins=100,density=1)

    # print(bins)

    # set the title and x,y lables of the chart:

    plt.title('Histogram of normal attributions')

    plt.xlabel('x')

    plt.ylabel('y')

    plt.text(0,0, r'$\mu=-1,\ \sigma=1$')

    """ Convenience method to get or set axis properties. """

    #if you not specify the axis ,then the chart axes will auto adjust to display properly!

    # plt.axis([-10,10,0,1])

    ''' set if you want to display the grid: '''

    plt.grid(True)

bar\_plot()

scatter\_plot()

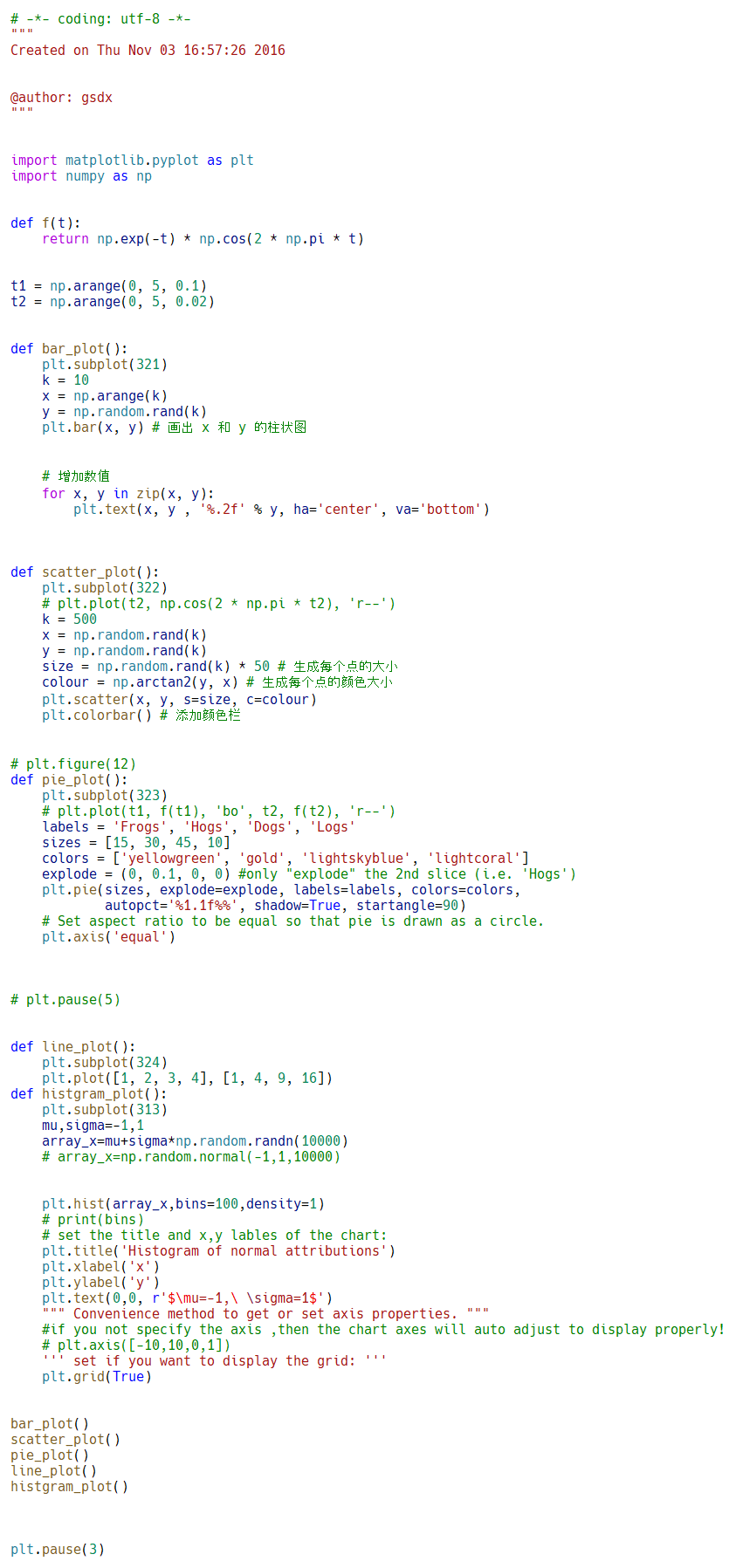
pie\_plot()

line\_plot()

histgram\_plot()

plt.pause(3)

源代码截图：



运行结果截图：

