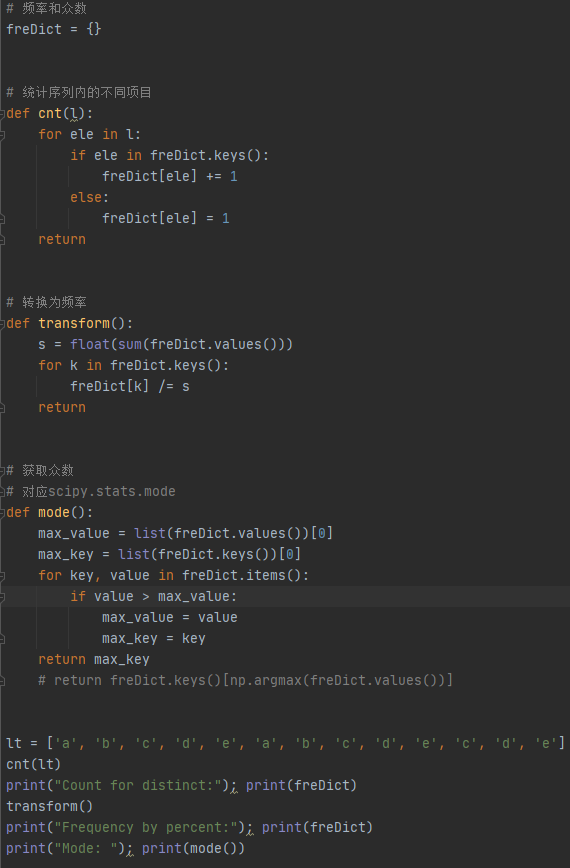
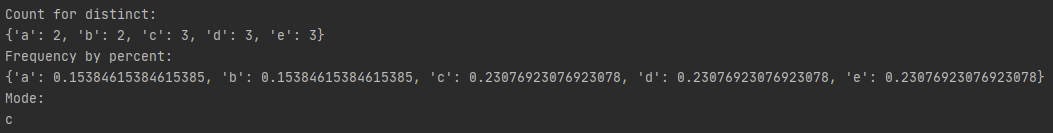
## 1 频率和众数

import numpy as np  
  
  
# 频率和众数  
freDict = {}  
  
  
# 统计序列内的不同项目  
def cnt(l):  
 for ele in l:  
 if ele in freDict.keys():  
 freDict[ele] += 1  
 else:  
 freDict[ele] = 1  
 return  
  
  
# 转换为频率  
def transform():  
 s = float(sum(freDict.values()))  
 for k in freDict.keys():  
 freDict[k] /= s  
 return  
  
  
# 获取众数  
# 对应scipy.stats.mode  
def mode():  
 max\_value = list(freDict.values())[0]  
 max\_key = list(freDict.keys())[0]  
 for key, value in freDict.items():  
 if value > max\_value:  
 max\_value = value  
 max\_key = key  
 return max\_key  
 # return freDict.keys()[np.argmax(freDict.values())]  
  
  
lt = ['a', 'b', 'c', 'd', 'e', 'a', 'b', 'c', 'd', 'e', 'c', 'd', 'e']  
cnt(lt)  
print("Count for distinct:"); print(freDict)  
transform()  
print("Frequency by percent:"); print(freDict)  
print("Mode: "); print(mode())

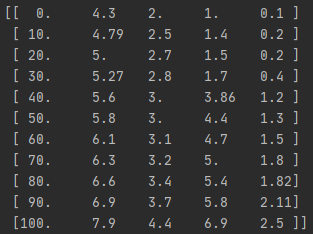




## 2 百分位数

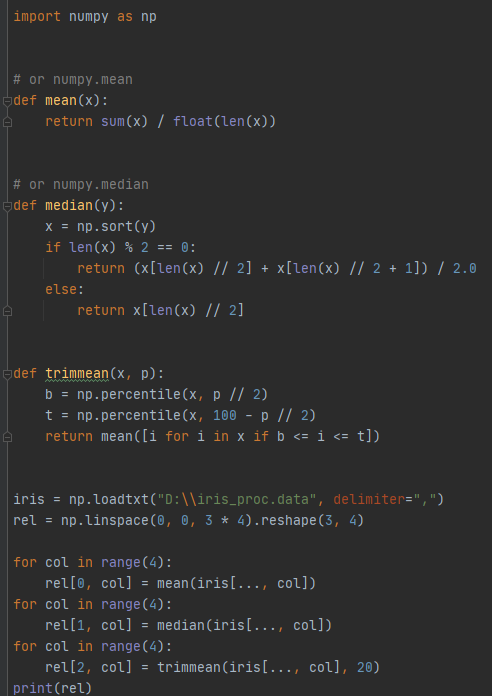
from math import \*  
import numpy as np  
  
  
# iris数据计算百分位数  
# 注意，没有处理p = 100的情况，同时默认数据是排序好的  
# 对应numpy.percentile  
def percentile(y, p):  
 x = np.sort(y)  
 if p <= 0:  
 return x[0]  
 if p >= 100:  
 return x[len(x) - 1]  
 pos = (len(x) - 1) \* p / 100.0  
 pos\_ceil = int(ceil(pos))  
 pos\_diff = pos - pos\_ceil  
 diff = x[pos\_ceil + 1] - x[pos\_ceil]  
 return x[pos\_ceil] + diff \* pos\_diff  
  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
  
rel = np.linspace(0, 0, 11 \* 5).reshape(11, 5)  
rel[..., 0] = range(0, 101, 10)  
for col in range(1, 5):  
 rel[..., col] = [percentile(iris[..., col - 1], p) for p in rel[..., 0]]  
print(rel)

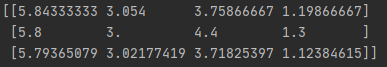




## 3 位置度量

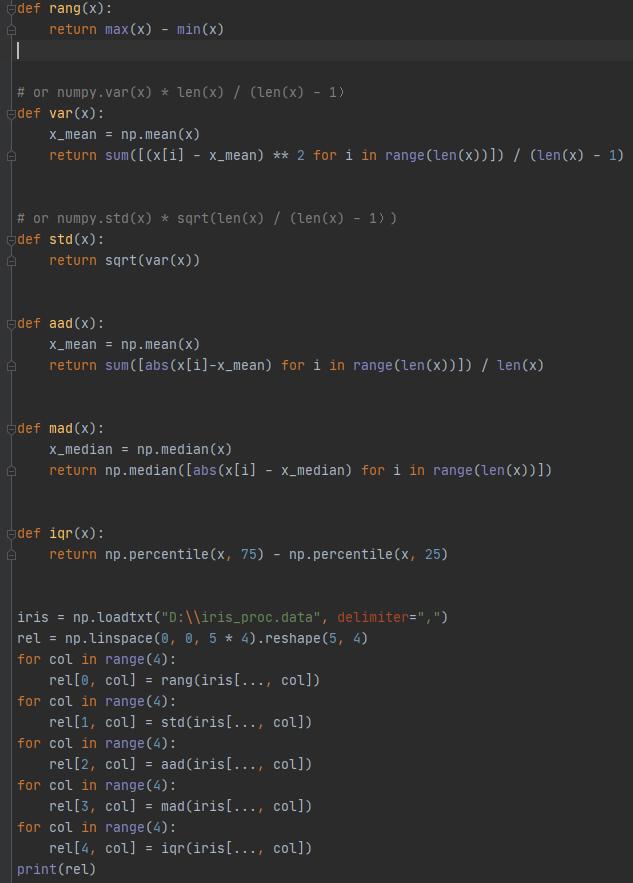
import numpy as np  
  
  
# or numpy.mean  
def mean(x):  
 return sum(x) / float(len(x))  
  
  
# or numpy.median  
def median(y):  
 x = np.sort(y)  
 if len(x) % 2 == 0:  
 return (x[len(x) // 2] + x[len(x) // 2 + 1]) / 2.0  
 else:  
 return x[len(x) // 2]  
  
  
def trimmean(x, p):  
 b = np.percentile(x, p // 2)  
 t = np.percentile(x, 100 - p // 2)  
 return mean([i for i in x if b <= i <= t])  
  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
rel = np.linspace(0, 0, 3 \* 4).reshape(3, 4)  
  
for col in range(4):  
 rel[0, col] = mean(iris[..., col])  
for col in range(4):  
 rel[1, col] = median(iris[..., col])  
for col in range(4):  
 rel[2, col] = trimmean(iris[..., col], 20)  
print(rel)

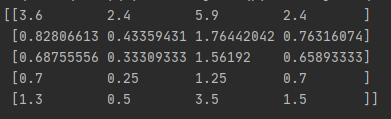




## 4分散程度度量

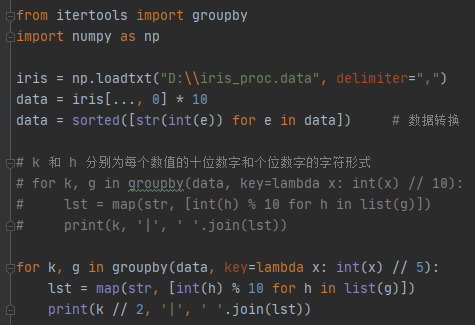
from math import \*  
import numpy as np  
  
  
def rang(x):  
 return max(x) - min(x)  
  
  
# or numpy.var(x) \* len(x) / (len(x) - 1）  
def var(x):  
 x\_mean = np.mean(x)  
 return sum([(x[i] - x\_mean) \*\* 2 for i in range(len(x))]) / (len(x) - 1)  
  
  
# or numpy.std(x) \* sqrt(len(x) / (len(x) - 1）)  
def std(x):  
 return sqrt(var(x))  
  
  
def aad(x):  
 x\_mean = np.mean(x)  
 return sum([abs(x[i]-x\_mean) for i in range(len(x))]) / len(x)  
  
  
def mad(x):  
 x\_median = np.median(x)  
 return np.median([abs(x[i] - x\_median) for i in range(len(x))])  
  
  
def iqr(x):  
 return np.percentile(x, 75) - np.percentile(x, 25)  
  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
rel = np.linspace(0, 0, 5 \* 4).reshape(5, 4)  
for col in range(4):  
 rel[0, col] = rang(iris[..., col])  
for col in range(4):  
 rel[1, col] = std(iris[..., col])  
for col in range(4):  
 rel[2, col] = aad(iris[..., col])  
for col in range(4):  
 rel[3, col] = mad(iris[..., col])  
for col in range(4):  
 rel[4, col] = iqr(iris[..., col])  
print(rel)



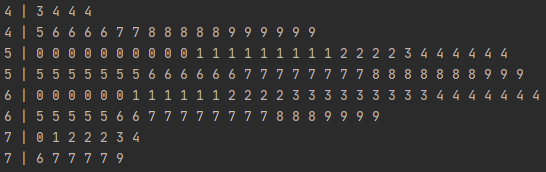


## 5 茎叶图

from itertools import groupby  
import numpy as np  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
data = iris[..., 0] \* 10  
data = sorted([str(int(e)) for e in data]) # 数据转换  
  
# k 和 h 分别为每个数值的十位数字和个位数字的字符形式  
# for k, g in groupby(data, key=lambda x: int(x) // 10):  
# lst = map(str, [int(h) % 10 for h in list(g)])  
# print(k, '|', ' '.join(lst))  
  
for k, g in groupby(data, key=lambda x: int(x) // 5):  
 lst = map(str, [int(h) % 10 for h in list(g)])  
 print(k // 2, '|', ' '.join(lst))



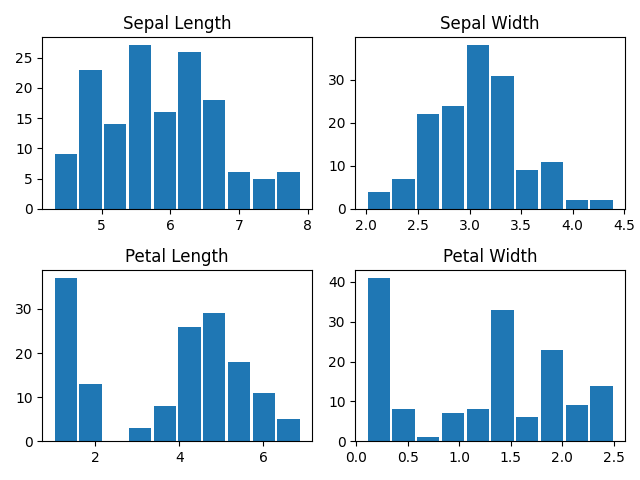


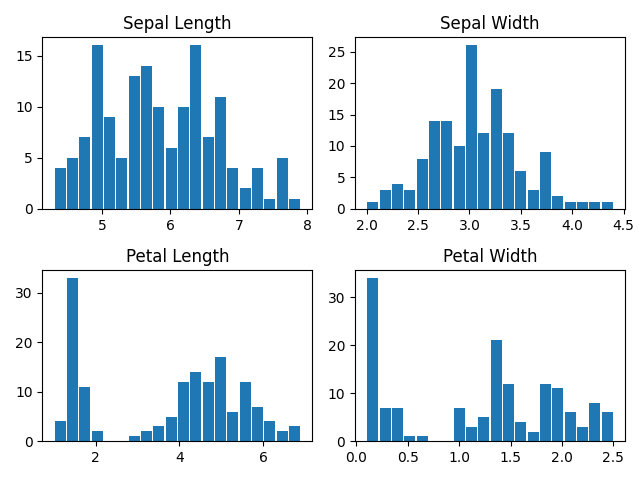


## 6 直方图

import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
len\_sepal = iris[..., 0]  
width\_sepal = iris[..., 1]  
len\_petal = iris[..., 2]  
width\_petal = iris[..., 3]  
  
plt.subplot(221)  
plt.title("Sepal Length")  
plt.hist(len\_sepal, bins=20, rwidth=0.9)  
plt.subplot(222)  
plt.title("Sepal Width")  
plt.hist(width\_sepal, bins=20, rwidth=0.9)  
plt.subplot(223)  
plt.title("Petal Length")  
plt.hist(len\_petal, bins=20, rwidth=0.9)  
plt.subplot(224)  
plt.title("Petal Width")  
plt.hist(width\_petal, bins=20, rwidth=0.9)  
plt.show()

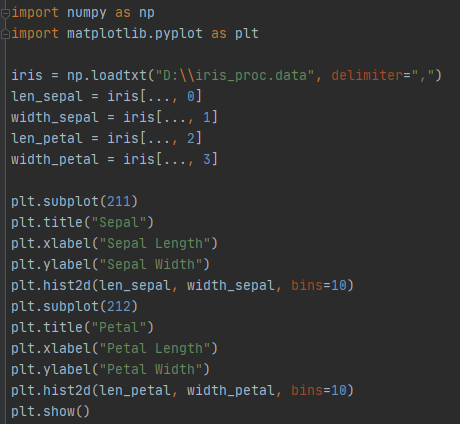


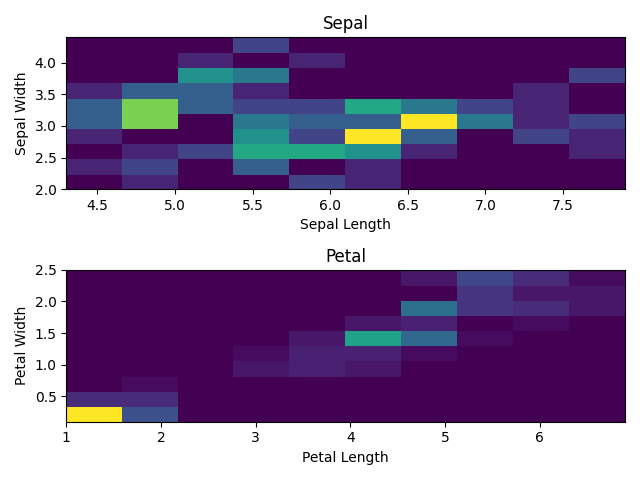




## 7 2D直方图

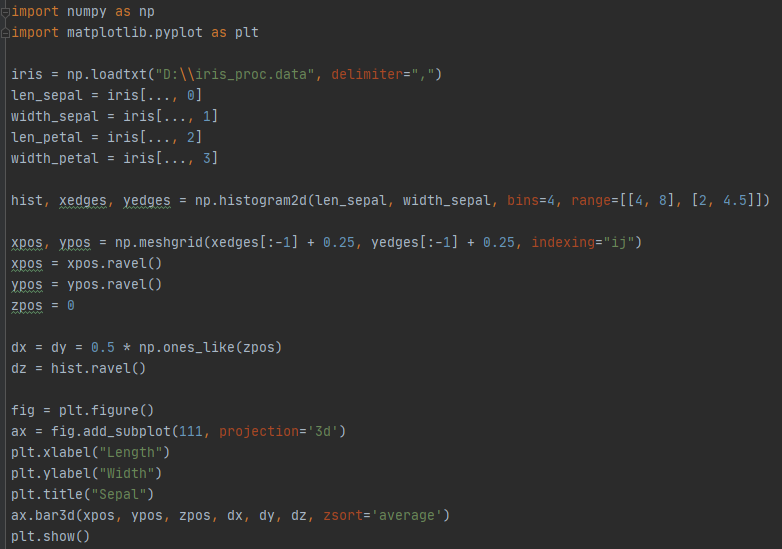
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
len\_sepal = iris[..., 0]  
width\_sepal = iris[..., 1]  
len\_petal = iris[..., 2]  
width\_petal = iris[..., 3]  
  
plt.subplot(211)  
plt.title("Sepal")  
plt.xlabel("Sepal Length")  
plt.ylabel("Sepal Width")  
plt.hist2d(len\_sepal, width\_sepal, bins=10)  
plt.subplot(212)  
plt.title("Petal")  
plt.xlabel("Petal Length")  
plt.ylabel("Petal Width")  
plt.hist2d(len\_petal, width\_petal, bins=10)  
plt.show()

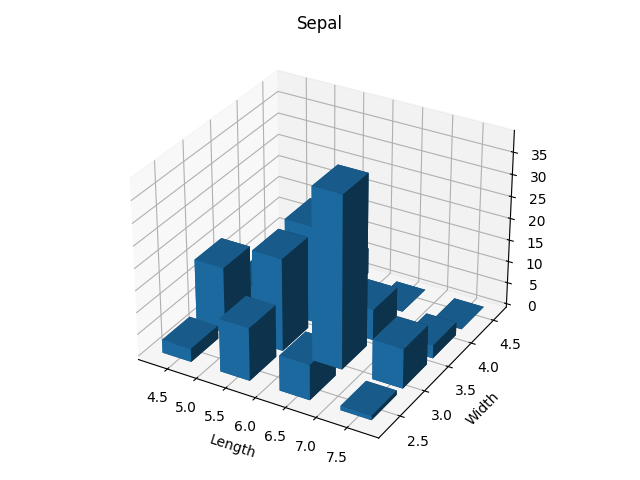




## 8 3D直方图

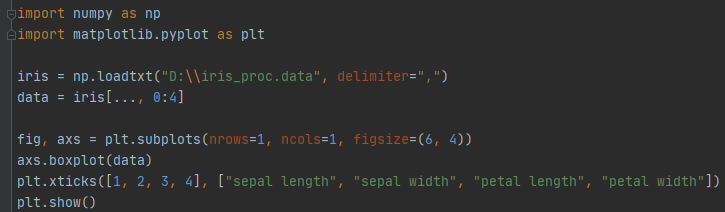
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
len\_sepal = iris[..., 0]  
width\_sepal = iris[..., 1]  
len\_petal = iris[..., 2]  
width\_petal = iris[..., 3]  
  
hist, xedges, yedges = np.histogram2d(len\_sepal, width\_sepal, bins=4, range=[[4, 8], [2, 4.5]])  
  
xpos, ypos = np.meshgrid(xedges[:-1] + 0.25, yedges[:-1] + 0.25, indexing="ij")  
xpos = xpos.ravel()  
ypos = ypos.ravel()  
zpos = 0  
  
dx = dy = 0.5 \* np.ones\_like(zpos)  
dz = hist.ravel()  
  
fig = plt.figure()  
ax = fig.add\_subplot(111, projection='3d')  
plt.xlabel("Length")  
plt.ylabel("Width")  
plt.title("Sepal")  
ax.bar3d(xpos, ypos, zpos, dx, dy, dz, zsort='average')  
plt.show()

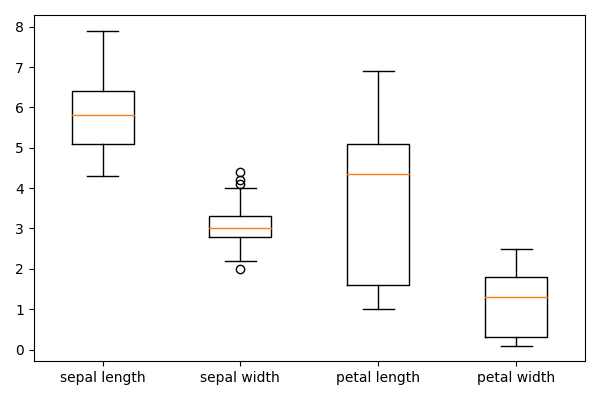




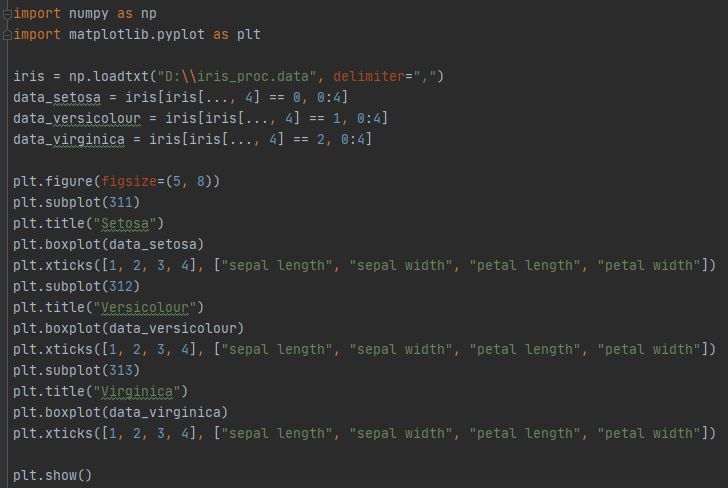
## 9 Box图

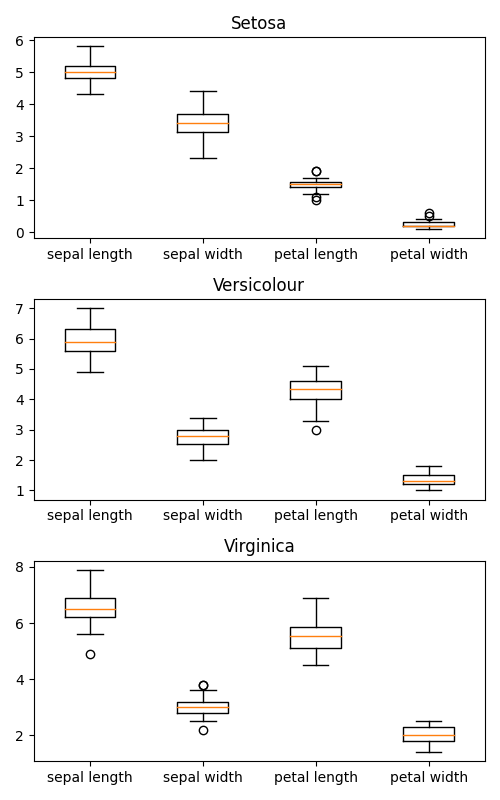
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
data = iris[..., 0:4]  
  
fig, axs = plt.subplots(nrows=1, ncols=1, figsize=(6, 4))  
axs.boxplot(data)  
plt.xticks([1, 2, 3, 4], ["sepal length", "sepal width", "petal length", "petal width"])  
plt.show()





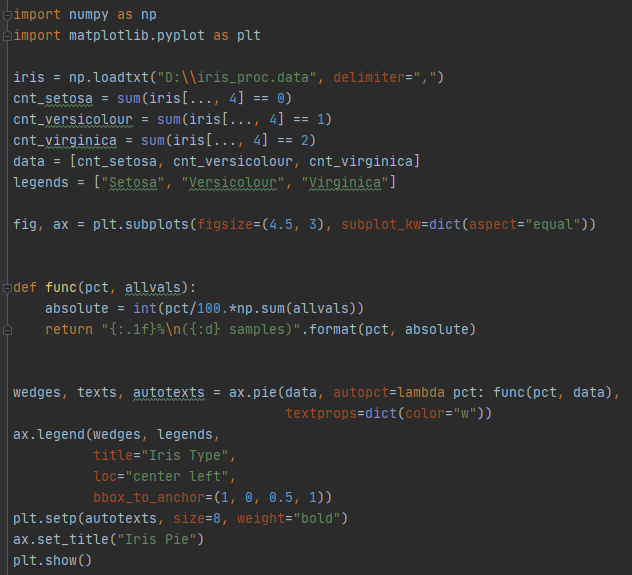
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
data\_setosa = iris[iris[..., 4] == 0, 0:4]  
data\_versicolour = iris[iris[..., 4] == 1, 0:4]  
data\_virginica = iris[iris[..., 4] == 2, 0:4]  
  
plt.figure(figsize=(5, 8))  
plt.subplot(311)  
plt.title("Setosa")  
plt.boxplot(data\_setosa)  
plt.xticks([1, 2, 3, 4], ["sepal length", "sepal width", "petal length", "petal width"])  
plt.subplot(312)  
plt.title("Versicolour")  
plt.boxplot(data\_versicolour)  
plt.xticks([1, 2, 3, 4], ["sepal length", "sepal width", "petal length", "petal width"])  
plt.subplot(313)  
plt.title("Virginica")  
plt.boxplot(data\_virginica)  
plt.xticks([1, 2, 3, 4], ["sepal length", "sepal width", "petal length", "petal width"])  
  
plt.show()

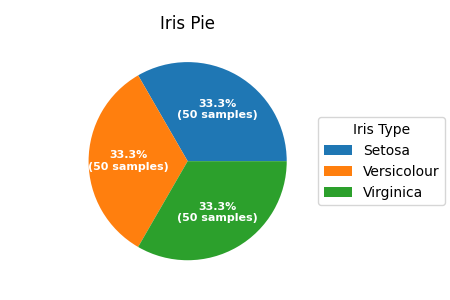




## 10 饼图

import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
cnt\_setosa = sum(iris[..., 4] == 0)  
cnt\_versicolour = sum(iris[..., 4] == 1)  
cnt\_virginica = sum(iris[..., 4] == 2)  
data = [cnt\_setosa, cnt\_versicolour, cnt\_virginica]  
legends = ["Setosa", "Versicolour", "Virginica"]  
  
fig, ax = plt.subplots(figsize=(4.5, 3), subplot\_kw=dict(aspect="equal"))  
  
  
def func(pct, allvals):  
 absolute = int(pct/100.\*np.sum(allvals))  
 return "{:.1f}%\n({:d} samples)".format(pct, absolute)  
  
  
wedges, texts, autotexts = ax.pie(data, autopct=lambda pct: func(pct, data),  
 textprops=dict(color="w"))  
ax.legend(wedges, legends,  
 title="Iris Type",  
 loc="center left",  
 bbox\_to\_anchor=(1, 0, 0.5, 1))  
plt.setp(autotexts, size=8, weight="bold")  
ax.set\_title("Iris Pie")  
plt.show()

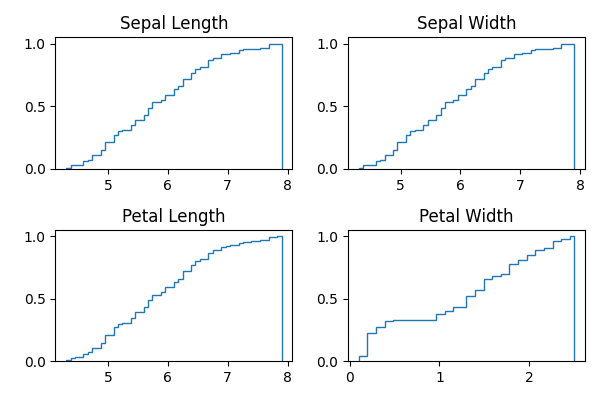




## 11 CDF图

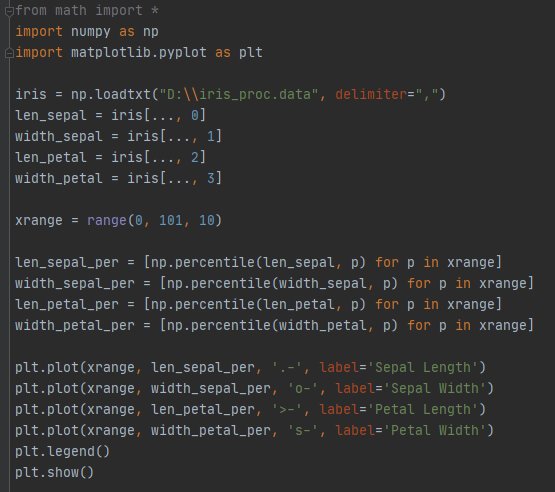
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
len\_sepal = iris[..., 0]  
width\_sepal = iris[..., 0]  
len\_petal = iris[..., 0]  
width\_petal = iris[..., 3]  
n\_bins = 50  
  
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(6, 4))  
axs[0, 0].set\_title("Sepal Length")  
axs[0, 0].hist(len\_sepal, n\_bins, density=True, histtype='step', cumulative=True)  
axs[0, 1].set\_title("Sepal Width")  
axs[0, 1].hist(width\_sepal, n\_bins, density=True, histtype='step', cumulative=True)  
axs[1, 0].set\_title("Petal Length")  
axs[1, 0].hist(len\_petal, n\_bins, density=True, histtype='step', cumulative=True)  
axs[1, 1].set\_title("Petal Width")  
axs[1, 1].hist(width\_petal, n\_bins, density=True, histtype='step', cumulative=True)  
plt.show()

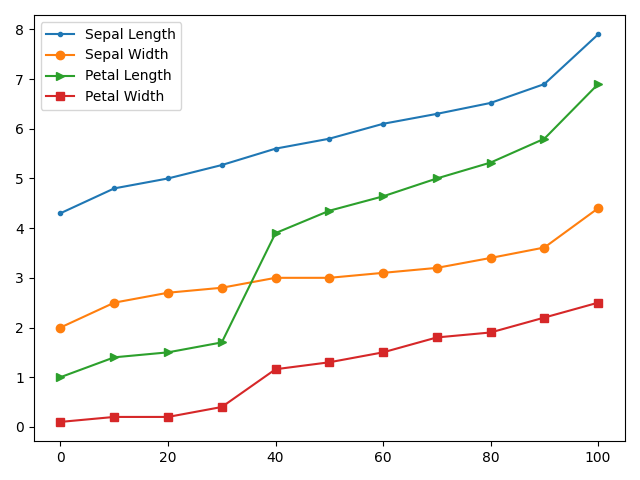




## 12 折线图

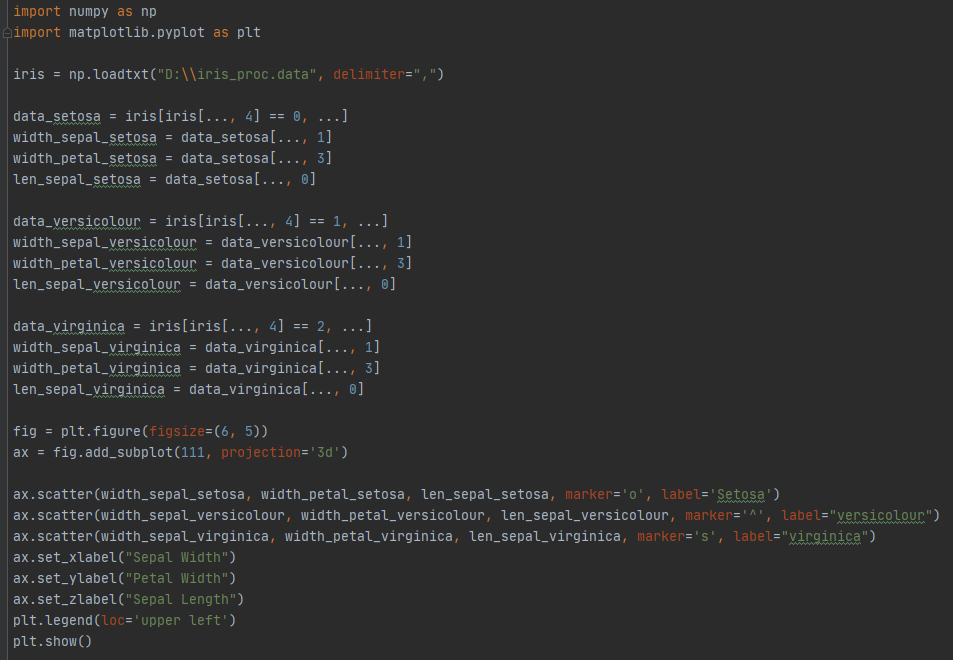
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
len\_sepal = iris[..., 0]  
width\_sepal = iris[..., 1]  
len\_petal = iris[..., 2]  
width\_petal = iris[..., 3]  
  
xrange = range(0, 101, 10)  
  
len\_sepal\_per = [np.percentile(len\_sepal, p) for p in xrange]  
width\_sepal\_per = [np.percentile(width\_sepal, p) for p in xrange]  
len\_petal\_per = [np.percentile(len\_petal, p) for p in xrange]  
width\_petal\_per = [np.percentile(width\_petal, p) for p in xrange]  
  
plt.plot(xrange, len\_sepal\_per, '.-', label='Sepal Length')  
plt.plot(xrange, width\_sepal\_per, 'o-', label='Sepal Width')  
plt.plot(xrange, len\_petal\_per, '>-', label='Petal Length')  
plt.plot(xrange, width\_petal\_per, 's-', label='Petal Width')  
plt.legend()  
plt.show()

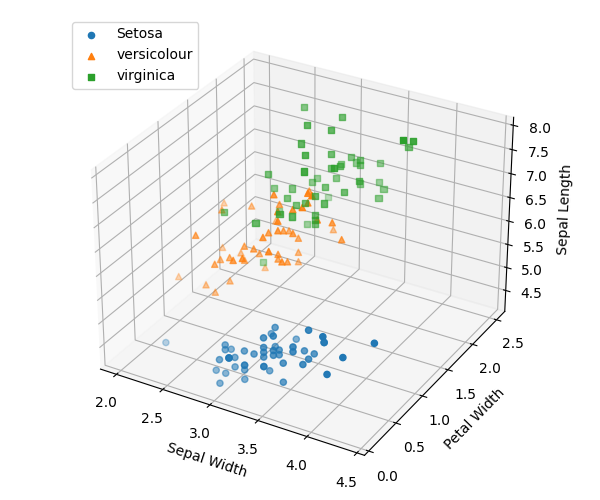




## 13 3D散点图

import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
  
data\_setosa = iris[iris[..., 4] == 0, ...]  
width\_sepal\_setosa = data\_setosa[..., 1]  
width\_petal\_setosa = data\_setosa[..., 3]  
len\_sepal\_setosa = data\_setosa[..., 0]  
  
data\_versicolour = iris[iris[..., 4] == 1, ...]  
width\_sepal\_versicolour = data\_versicolour[..., 1]  
width\_petal\_versicolour = data\_versicolour[..., 3]  
len\_sepal\_versicolour = data\_versicolour[..., 0]  
  
data\_virginica = iris[iris[..., 4] == 2, ...]  
width\_sepal\_virginica = data\_virginica[..., 1]  
width\_petal\_virginica = data\_virginica[..., 3]  
len\_sepal\_virginica = data\_virginica[..., 0]  
  
fig = plt.figure(figsize=(6, 5))  
ax = fig.add\_subplot(111, projection='3d')  
  
ax.scatter(width\_sepal\_setosa, width\_petal\_setosa, len\_sepal\_setosa, marker='o', label='Setosa')  
ax.scatter(width\_sepal\_versicolour, width\_petal\_versicolour, len\_sepal\_versicolour, marker='^', label="versicolour")  
ax.scatter(width\_sepal\_virginica, width\_petal\_virginica, len\_sepal\_virginica, marker='s', label="virginica")  
ax.set\_xlabel("Sepal Width")  
ax.set\_ylabel("Petal Width")  
ax.set\_zlabel("Sepal Length")  
plt.legend(loc='upper left')  
plt.show()

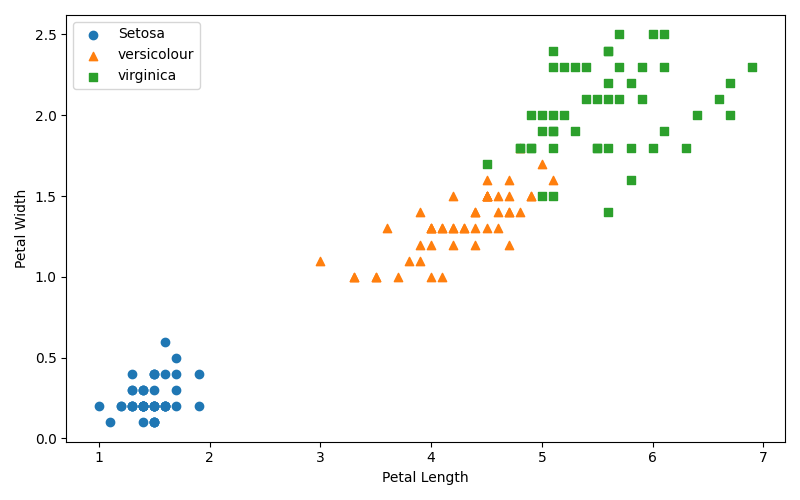




## 14 2D散点图

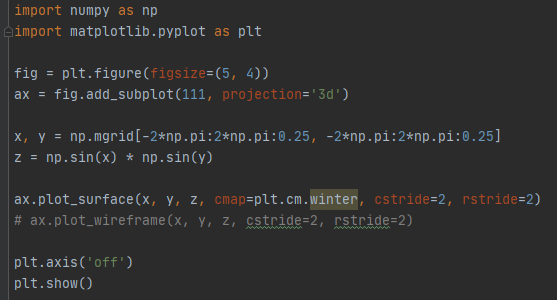
import numpy as np  
import matplotlib.pyplot as plt  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
  
data\_setosa = iris[iris[..., 4] == 0, ...]  
len\_petal\_setosa = data\_setosa[..., 2]  
width\_petal\_setosa = data\_setosa[..., 3]  
  
data\_versicolour = iris[iris[..., 4] == 1, ...]  
len\_petal\_versicolour = data\_versicolour[..., 2]  
width\_petal\_versicolour = data\_versicolour[..., 3]  
  
data\_virginica = iris[iris[..., 4] == 2, ...]  
len\_pepal\_virginica = data\_virginica[..., 2]  
width\_petal\_virginica = data\_virginica[..., 3]  
  
  
fig = plt.figure(figsize=(8, 5))  
ax = fig.add\_subplot(111)  
  
ax.scatter(len\_petal\_setosa, width\_petal\_setosa, marker='o', label='Setosa')  
ax.scatter(len\_petal\_versicolour, width\_petal\_versicolour, marker='^', label="versicolour")  
ax.scatter(len\_pepal\_virginica, width\_petal\_virginica, marker='s', label="virginica")  
ax.set\_xlabel("Petal Length")  
ax.set\_ylabel("Petal Width")  
plt.legend(loc='upper left')  
plt.show()

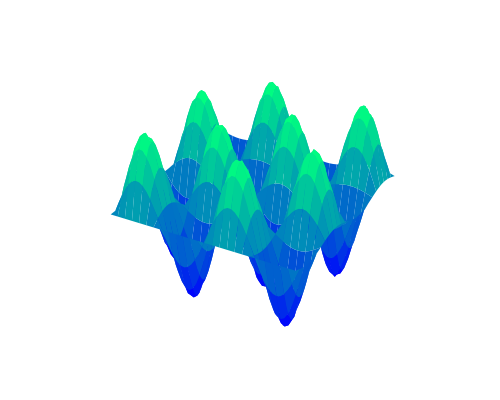




## 15 3D曲面图

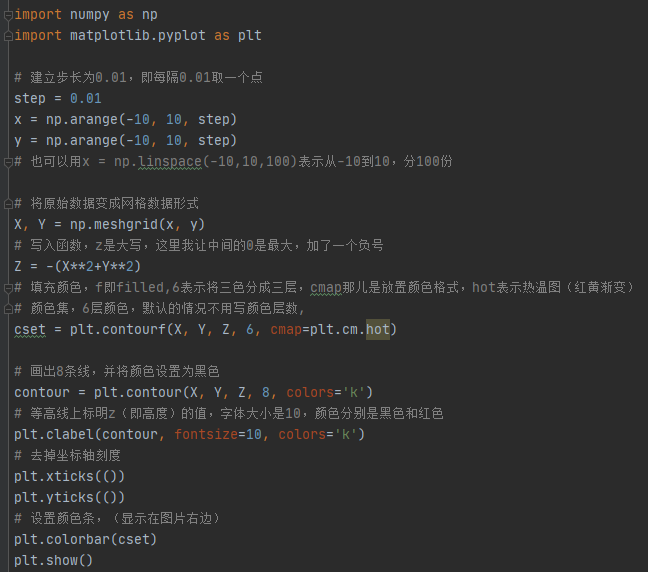
import numpy as np  
import matplotlib.pyplot as plt  
  
fig = plt.figure(figsize=(5, 4))  
ax = fig.add\_subplot(111, projection='3d')  
  
x, y = np.mgrid[-2\*np.pi:2\*np.pi:0.25, -2\*np.pi:2\*np.pi:0.25]  
z = np.sin(x) \* np.sin(y)  
  
ax.plot\_surface(x, y, z, cmap=plt.cm.winter, cstride=2, rstride=2)  
# ax.plot\_wireframe(x, y, z, cstride=2, rstride=2)  
  
plt.axis('off')  
plt.show()

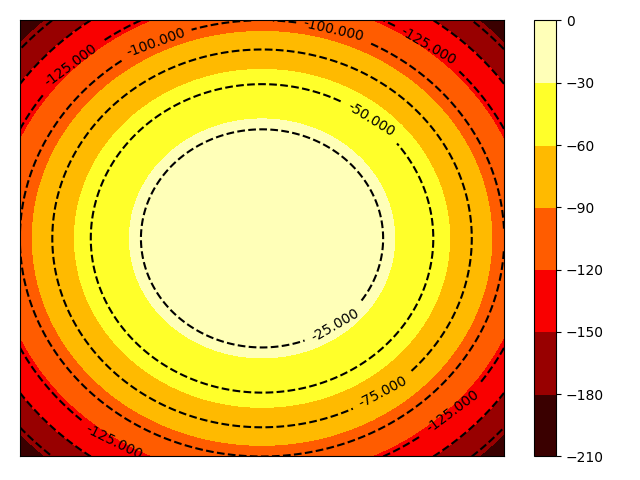




## 16 等高线图

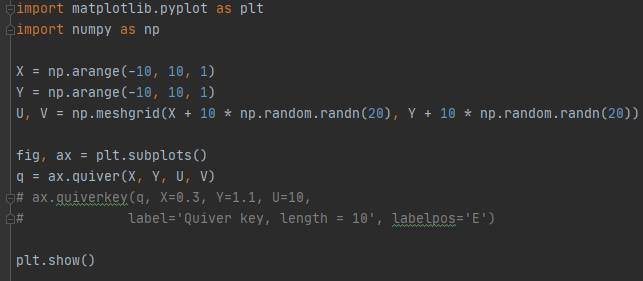
import numpy as np  
import matplotlib.pyplot as plt  
  
# 建立步长为0.01，即每隔0.01取一个点  
step = 0.01  
x = np.arange(-10, 10, step)  
y = np.arange(-10, 10, step)  
# 也可以用x = np.linspace(-10,10,100)表示从-10到10，分100份  
  
# 将原始数据变成网格数据形式  
X, Y = np.meshgrid(x, y)  
# 写入函数，z是大写，这里我让中间的0是最大，加了一个负号  
Z = -(X\*\*2+Y\*\*2)  
# 填充颜色，f即filled,6表示将三色分成三层，cmap那儿是放置颜色格式，hot表示热温图（红黄渐变）  
# 颜色集，6层颜色，默认的情况不用写颜色层数,  
cset = plt.contourf(X, Y, Z, 6, cmap=plt.cm.hot)  
  
# 画出8条线，并将颜色设置为黑色  
contour = plt.contour(X, Y, Z, 8, colors='k')  
# 等高线上标明z（即高度）的值，字体大小是10，颜色分别是黑色和红色  
plt.clabel(contour, fontsize=10, colors='k')  
# 去掉坐标轴刻度  
plt.xticks(())  
plt.yticks(())  
# 设置颜色条，（显示在图片右边）  
plt.colorbar(cset)  
plt.show()

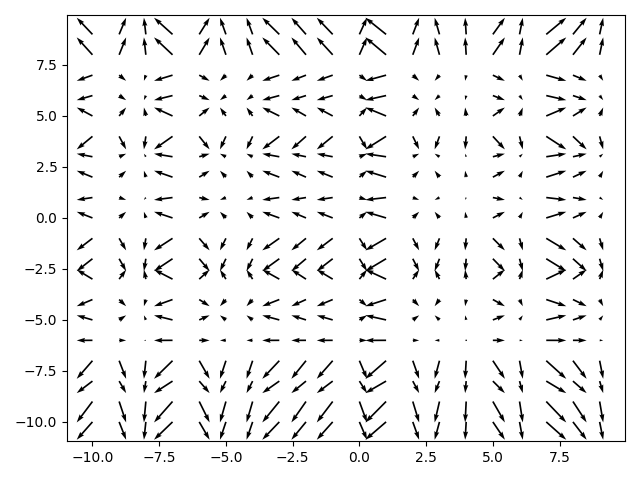




## 17 矢量图

import matplotlib.pyplot as plt  
import numpy as np  
  
X = np.arange(-10, 10, 1)  
Y = np.arange(-10, 10, 1)  
U, V = np.meshgrid(X + 10 \* np.random.randn(20), Y + 10 \* np.random.randn(20))  
  
fig, ax = plt.subplots()  
q = ax.quiver(X, Y, U, V)  
# ax.quiverkey(q, X=0.3, Y=1.1, U=10,  
# label='Quiver key, length = 10', labelpos='E')  
  
plt.show()

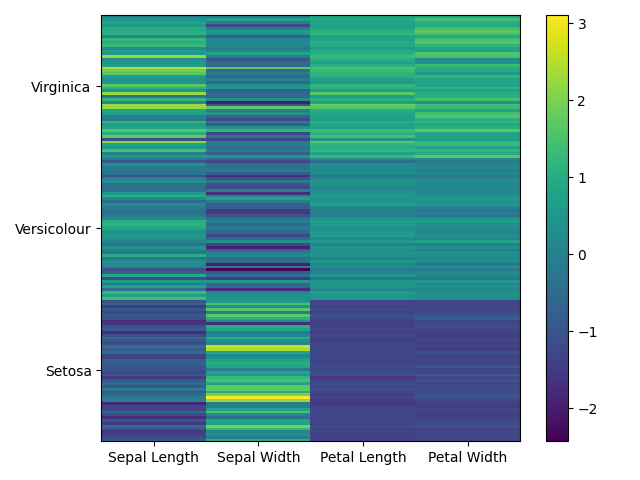




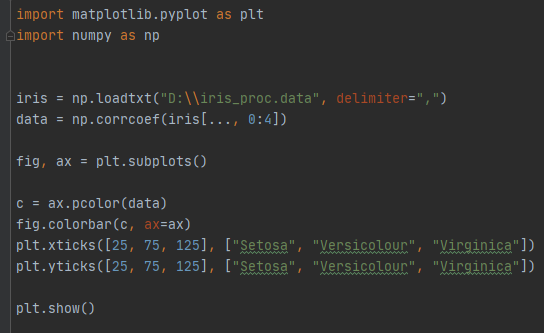
## 18 像素图

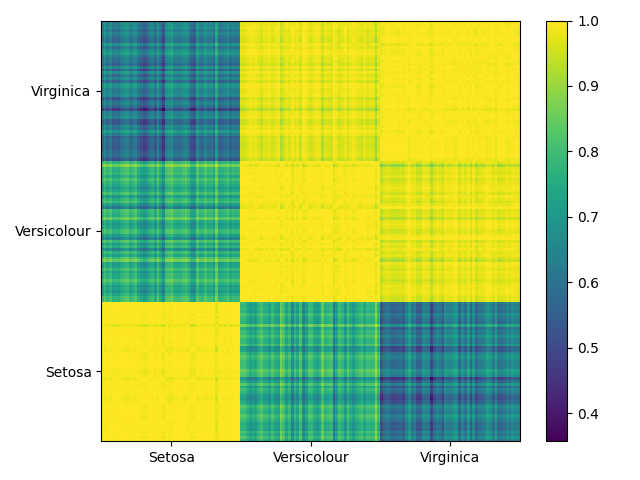
from math import \*  
import matplotlib.pyplot as plt  
import numpy as np  
  
  
# 序列标准化  
def normal(x):  
 x\_mean = np.mean(x)  
 x\_var = sum([(x[i] - x\_mean)\*\*2 for i in range(len(x))]) / (len(x) - 1)  
 x\_std = sqrt(x\_var)  
 return [(x[i] - x\_mean)/x\_std for i in range(len(x))]  
  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
# data = iris[..., 0:4]  
data = np.empty\_like(iris[..., 0:4])  
data[..., 0] = normal(iris[..., 0])  
data[..., 1] = normal(iris[..., 1])  
data[..., 2] = normal(iris[..., 2])  
data[..., 3] = normal(iris[..., 3])  
  
fig, ax = plt.subplots()  
  
c = ax.pcolor(data)  
fig.colorbar(c, ax=ax)  
plt.xticks([0.5, 1.5, 2.5, 3.5], ["Sepal Length", "Sepal Width", "Petal Length", "Petal Width"])  
plt.yticks([25, 75, 125], ["Setosa", "Versicolour", "Virginica"])  
  
plt.show()





import matplotlib.pyplot as plt  
import numpy as np  
  
  
iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
data = np.corrcoef(iris[..., 0:4])  
  
fig, ax = plt.subplots()  
  
c = ax.pcolor(data)  
fig.colorbar(c, ax=ax)  
plt.xticks([25, 75, 125], ["Setosa", "Versicolour", "Virginica"])  
plt.yticks([25, 75, 125], ["Setosa", "Versicolour", "Virginica"])  
  
plt.show()





## 19 属性平行坐标图

import matplotlib.pyplot as plt  
import numpy as np  
  
old\_iris = np.loadtxt("D:\\iris\_proc.data", delimiter=",")  
# 调整，第1列和第2列交换  
iris = old\_iris[..., [1, 0, 2, 3, 4]]  
  
data\_setosa = iris[iris[..., 4] == 0, 0:4]  
data\_versicolour = iris[iris[..., 4] == 1, 0:4]  
data\_virginica = iris[iris[..., 4] == 2, 0:4]  
  
x = np.arange(4)  
  
plt.figure(figsize=(6, 4))  
fig, ax = plt.subplots()  
  
for i in range(len(data\_setosa)):  
 ax.plot(x, data\_setosa[i], '-r', label=["", "Setosa"][i == 0])  
for i in range(len(data\_versicolour)):  
 ax.plot(x, data\_versicolour[i], '--g', label=["", "Versicolour"][i == 0])  
for i in range(len(data\_virginica)):  
 ax.plot(x, data\_virginica[i], ':b', label=["", "Virginica"][i == 0])  
  
ax.set\_xticks([0, 1, 2, 3])  
ax.set\_xticklabels(['Sepal Width', 'Sepal Length', "Petal Length", "Petal Width"])  
  
ax.legend(loc="upper right")  
plt.show()



