**实验三 《k-means聚类算法》**

**题目**

用C++实现k-means聚类算法，

1. 对实验二中的z-score归一化的成绩数据进行测试，观察聚类为2类，3类，4类，5类的结果，观察得出什么结论？
2. 由老师给出测试数据，进行测试，并画出可视化出散点图，类中心，类半径，并分析聚为几类合适。

样例数据(x,y)数据对：

|  |  |
| --- | --- |
| 3.45 | 7.08 |
| 1.76 | 7.24 |
| 4.29 | 9.55 |
| 3.35 | 6.65 |
| 3.17 | 6.41 |
| 3.68 | 5.99 |
| 2.11 | 4.08 |
| 2.58 | 7.10 |
| 3.45 | 7.88 |
| 6.17 | 5.40 |
| 4.20 | 6.46 |
| 5.87 | 3.87 |
| 5.47 | 2.21 |
| 5.97 | 3.62 |
| 6.24 | 3.06 |
| 6.89 | 2.41 |
| 5.38 | 2.32 |
| 5.13 | 2.73 |
| 7.26 | 4.19 |
| 6.32 | 3.62 |

找到聚类中心后，判断(2,6)是属于哪一类？

**注意**

除文件读取外，不能使用C++基础库以外的API和库函数。

实现：

# 使用随机样例初始化质心

def findPoints(dataSet, k):

numSamples, dim = dataSet.shape # 计算数据集的行列

centroids = np.zeros((k, dim)) # 产生k行，dim列零矩阵

for i in range(k):

index = int(np.random.uniform(0, numSamples)) # 给出一个服从均匀分布的在0~numSamples之间的整数

centroids[i, :] = dataSet[index, :] # 第index行作为簇心

return centroids

# k均值聚类

def kmeans(dataSet, k):

numSamples = dataSet.shape[0] # 计算数据集的行

clusterAssment = np.zeros((numSamples, 2)) # 随机生成一个106\*2矩阵

clusterChanged = True

# 1: 初始化质心，选取k个点作为种子点

centroids = findPoints(dataSet, k)

while clusterChanged:

clusterChanged = False

for i in range(numSamples):

minDist = 1000000.0 # 最小距离

minIndex = 0 # 最小距离对应的点群

# 2: 分别计算每个数据点到k个种子点的距离，离哪个种子点最近，就属于哪类

for j in range(k):

distance = euclDistance(centroids[j, :], dataSet[i, :]) # 计算每个数据到每个簇中心的欧式距离

if distance < minDist: # 如果距离小于当前最小距离

minDist = distance # 则最小距离更新

minIndex = j # 对应的点群也会更新

# 3：重新计算k个种子点的坐标

if clusterAssment[i, 0] != minIndex: # 如当前数据不属于该点群

clusterChanged = True # 聚类操作需要继续

clusterAssment[i, :] = minIndex, minDist \*\* 2

# 4：重复2、3步，直到种子点坐标不变或者循环次数完成

for j in range(k):

# 提取同一类别的向量

pointsInCluster = dataSet[np.nonzero(clusterAssment[:, 0] == j)]

centroids[j, :] = np.mean(pointsInCluster, axis=0) # 对每列求均值

return centroids, clusterAssment

# 画图

def showCluster(dataSet, k, centroids, clusterAssment):

numSamples = dataSet.shape[0] # numSample - 样例数量

mark = ['or', 'ob', 'og', 'ok', '^r', '+r', 'sr', 'dr', '<r', 'pr']

# 画数据集

for i in range(numSamples):

markIndex = int(clusterAssment[i, 0])

plt.plot(dataSet[i, 0], dataSet[i, 1], mark[markIndex])

plt.title(" The classification results of k-means cluster")

mark = ['Dr', 'Db', 'Dg', 'Dk', '^b', '+b', 'sb', 'db', '<b', 'pb']

# 画质心

for i in range(k):

plt.plot(centroids[i, 0], centroids[i, 1], mark[i], ms=12.0)

plt.show()

def exe3():

# 1.对实验二中的z-score归一化的成绩数据进行测试

# 源数据获取

df = pd.read\_table("Lee\_data.txt", header=None, sep='\t')

df = df.sample(frac=1) # 打乱数据，使之更具客观性

data = df.to\_numpy()

# 设置聚为几类 2、3、4、5

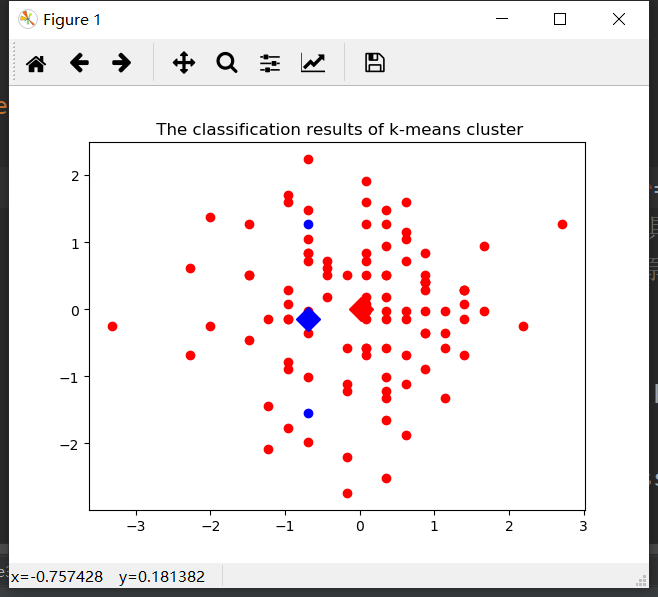
k = int(input("k="))

centroids, clusterAssment = kmeans(data, k)

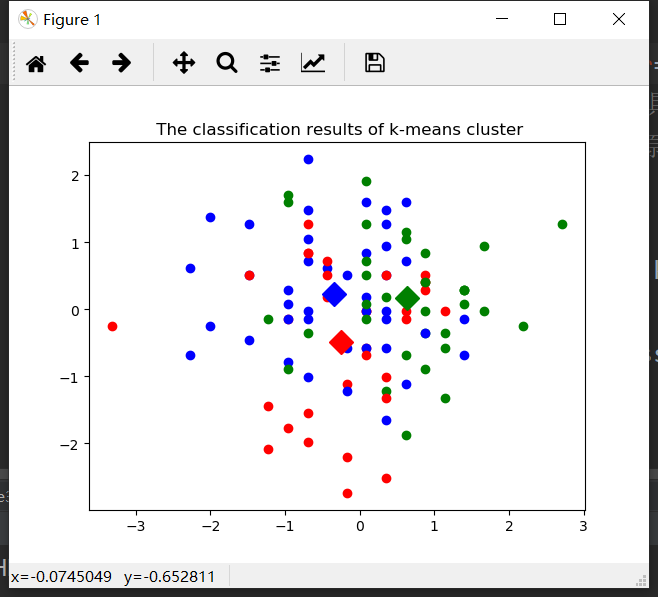
# 可视化

showCluster(data, k, centroids, clusterAssment)

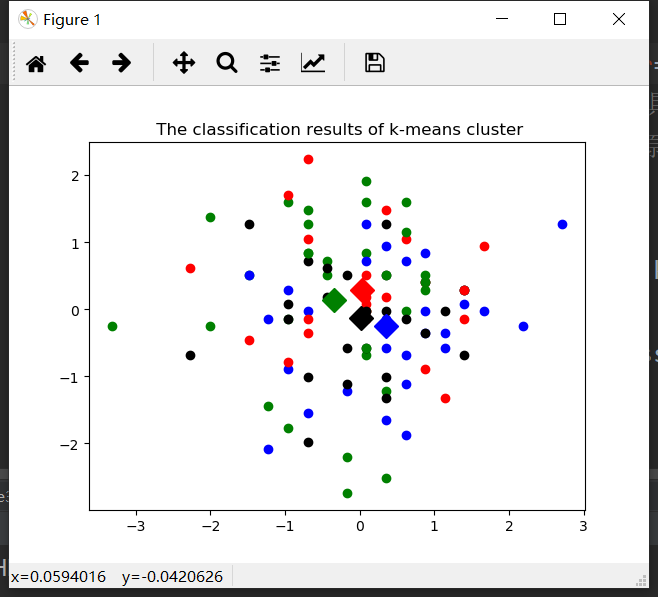
聚为2类：



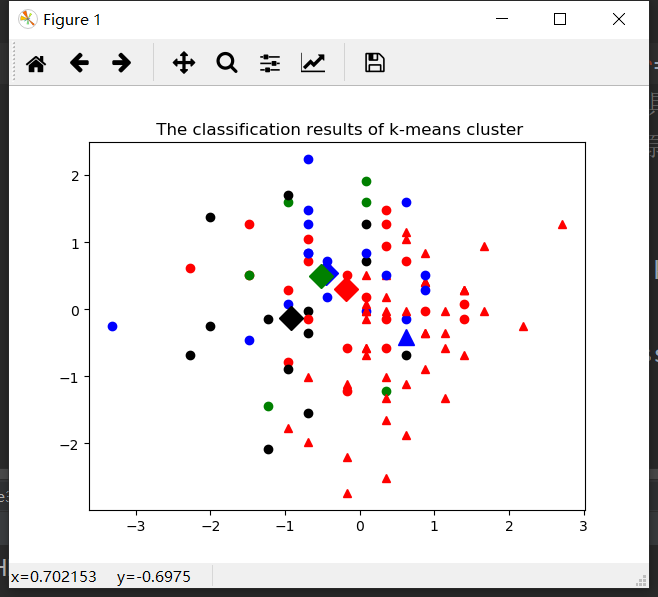
聚为3类：



聚为4类：



聚为5类：



从对数据分类进行可视化的结果来看，我们可以得出这样的结论：对学生进行分类（2类、3类、4类、5类）效果不好，说明学生的相似度低，难以划分为不同类别。

# 2、由老师给出测试数据，进行测试，并画出可视化出散点图，类中心，类半径，并分析聚为几类合适

# 源数据获取

df = pd.read\_table("teacher\_data.txt", header=None, sep='\t')

df.columns = ['x', 'y']

# 画散点图

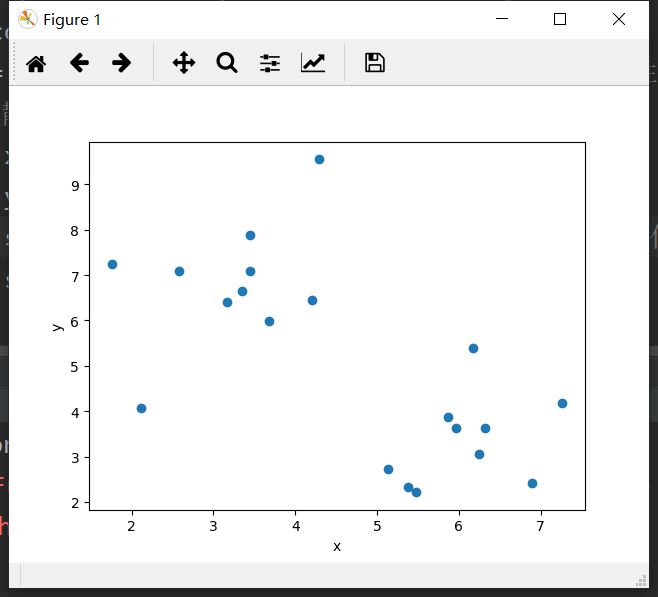
plt.xlabel("x") # 定义x坐标

plt.ylabel("y") # 定义y坐标

plt.scatter(df['x'].values, df['y'].values) # 传入数据

plt.show()

散点图：



从散点图可以看出数据明显分为两类

# 分析聚为2类较为合适

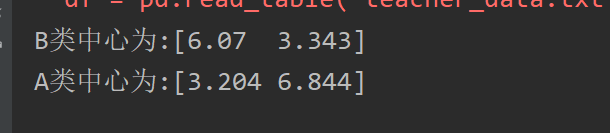
data = df.to\_numpy()

k = 2

centroids, clusterAssment = kmeans(data, k)

print("B类中心为:%s" % centroids[0])

print("A类中心为:%s" % centroids[1])



# 可视化

showCluster(data, k, centroids, clusterAssment)

# 判断（2，6）属于哪一类

print("请输入要判断的坐标：")

x = int(input("x="))

y = int(input("y="))

m1 = (centroids[0][0] + centroids[1][0]) / 2

m2 = (centroids[0][1] + centroids[1][1]) / 2

if x < m1 and y > m2:

print("属于A类")

if x > m1 and y < m2:

print("属于B类")

