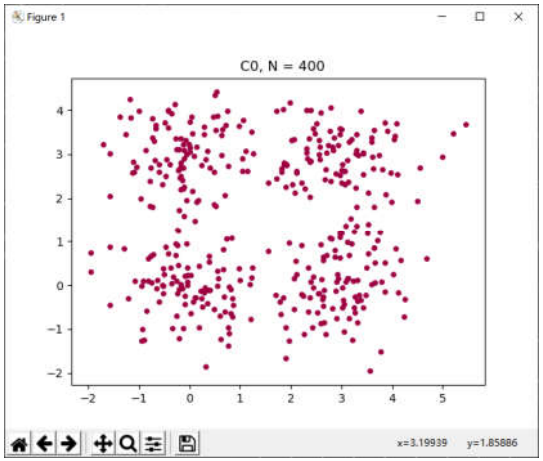

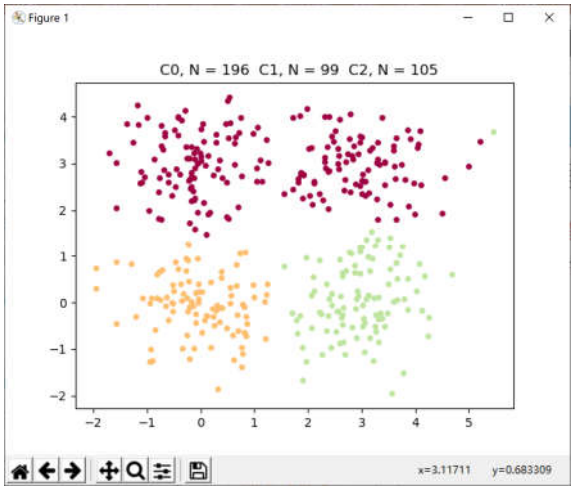

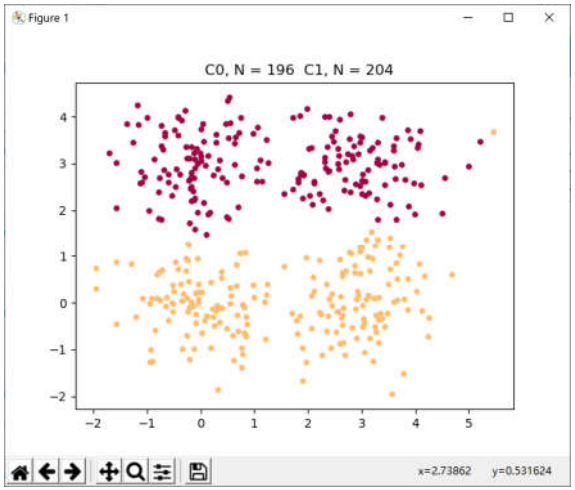

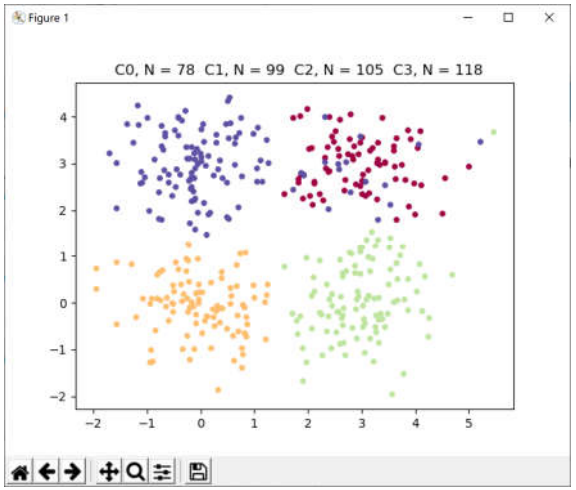

Python编程与人工智能实践

算法篇：

DIANA

（**DIvisive ANAlysis**

（分裂分析聚类））

于泓

鲁东大学

信息与电气工程学院

2022.6.30

DIANA

- DIANA是一种自上向下的分裂式聚类方法，首先将所有的数据当做一个簇，然后对这个簇进行分裂，分解成两个簇。在选取两个簇中分离度比较大的簇进行进一步的分解……通过不断的迭代直到簇的数目达到提前的预设值为止。

	A	B	C	D	E	
A	0	1	2.8	3.6	4.2	2.9
B	1	0	2.2	3.2	3.6	2.5
C	2.8	2.2	0	1	1.4	1.9
D	3.6	3.2	1	0	1	2.2
E	4.2	3.6	1.4	1	0	2.6

[A] [B,C,D,E] (2.1)

AB=1 BC=2.2,BD=3.2,BE=3.6

[A,B] [C,D,E]

CA=2.8 CB=2.2 CD=1,CE=1.4

DA=3.6 DB=3.2 DC=1,DE=1

EA=4.2 EB=3.6 EC=1.4,ED=1

算法原理:

(1) 所有样本点作为一个簇

(2) 进行簇的分裂

split-cluster last-cluster

(2.1) 在样本点中选取分离度最大(距离其他点最远)的点,放入split-cluster

(2.2)计算last-cluster 中的点（P）与split-cluster中点的距离以及与last-cluster中其他点的距离。若P距离split-cluster 更近，则将P点移到split-cluster

(2.3) 迭代遍历（2.2）直到last-cluster中没有点能移动到split-cluster中。此时实现了一次分裂

(2.4) 计算分裂出的两个簇中相异度更大的簇跳转到步骤（2.1）进行进一步的分裂

	A	B
A	0	1
B	1	0

	C	D	E	
C	0	1	1.4	1.2
D	1	0	1	1
E	1.4	1	0	1.2

[C] [D,E]

CD=1 DE = 1.4

[C,D] [E]

代码：

计算差异度

找到差异度最大的点

```
# 对一个簇 进行分裂
# cluster 一个簇中的样本的编号
# dis_matrix 全部样本 两两之间的距离
def split_one_cluster(cluster,dis_matrix):
    # 在一个簇中找到一个离群点,这个点距离其他点的平均距离最远
    temp_dis_matrix = dis_matrix[cluster][:,cluster]
    max_dis_index = np.argmax(np.mean(temp_dis_matrix,axis=1))
    id_split = cluster[max_dis_index]

    # 将簇先分成两个簇，离群点一个簇，其他点一个簇
    split_cluster = [id_split]
    last_cluster = cluster.copy()
    last_cluster.pop(max_dis_index)
```

分成两个簇

```
while True:
    flag_split = False
    for i in range(len(last_cluster))[::-1]:
        # 遍历其他点簇 中的所有点
        p = last_cluster[i]
        # 计算点p 和 split 中点的距离
        dis_p_split = dis_matrix[p,split_cluster]
        # 计算点 p 和 last 中 其他点的距离
        point_left = last_cluster.copy()
        point_left.pop(i)
        dis_p_last = dis_matrix[p,point_left]

        # 如果点p 距离 split 更近
        if np.mean(dis_p_split) <= np.mean(dis_p_last):
            # 那么把点p 加入到 split 簇中
            split_cluster.append(p)
            last_cluster.pop(i)
            flag_split = True
            break

    # 如果遍历一轮没有找到新的分离点，则分裂结束
    if flag_split == False:
        break

    return split_cluster,last_cluster
```

P与split簇的距离

P与last簇中其他点的距离

从若干簇中，找到差异性最大的簇

```python
# 从一组簇中 找到分离度最大的一个簇 进行分裂
def get_max_separation_cluster(clusters,dis_matrix):
    dgree_separation = []
    for cluster in clusters:
        temp_dis_matrix = dis_matrix[cluster][:,cluster]
        dgree_separation.append(np.max(np.mean(temp_dis_matrix,axis=1)))
    return np.argmax(dgree_separation)
```

```python
if __name__ == "__main__":
    # 读取数据
    a = np.random.multivariate_normal([3,3], [[.5,0],[0,.5]], 100)
    b = np.random.multivariate_normal([0,0], [[0.5,0],[0,0.5]], 100)
    c = np.random.multivariate_normal([3,0], [[0.5,0],[0,0.5]], 100)
    d = np.random.multivariate_normal([0,3], [[0.5,0],[0,0.5]], 100)
    data = np.r_[a,b,c,d]


    # data= data_generate()
    print(data.shape)
    N,D= np.shape(data)

    # 计算数据点两两之间的距离
    tile_x = np.tile(np.expand_dims(data,1),[1,N,1]) # N, N,D
    tile_y = np.tile(np.expand_dims(data,0),[N,1,1]) # N, N,D
    dis_matrix = np.linalg.norm((tile_x-tile_y),axis=-1)

    # 从初始化时所有的样本点在一个类中
    clusters = [[i for i in range(N)]]
    K = 4
```

```
while True:
    # 找到区分度最大的一个簇
    index_sel = get_max_separation_cluster(clusters,dis_matrix)
    #对这个簇进行分裂
    c_1,c_2 = split_one_cluster(clusters[index_sel],dis_matrix)

    # 删除分裂前的一个簇 添加分裂后的两个簇
    clusters.pop(index_sel)
    clusters.append(c_1)
    clusters.append(c_2)

    # 显示结果
    draw(data,clusters,str_title="")
    plt.show()
    if len(clusters)>=K:
        break

# 显示聚类结果
cluster_labels = np.zeros(N)
for i in range(len(clusters)):
    cluster_labels[clusters[i]] = i
print(cluster_labels)

draw(data,clusters,str_title="")
plt.show()
```

```
def draw(datas,clusters,str_title=""):

    N_cluster = len(clusters)
    plt.cla()
    colors = [plt.cm.Spectral(each) for each in np.linspace(0, 1,N_cluster)]

    for i,cluster in enumerate(clusters):
        datas_draw = datas[cluster,:]
        datas_draw = datas_draw[:,:2]
        plt.scatter(datas_draw[:,0],datas_draw[:,1],s=5,color=colors[i])

    plt.title(str_title)

    plt.show()
```

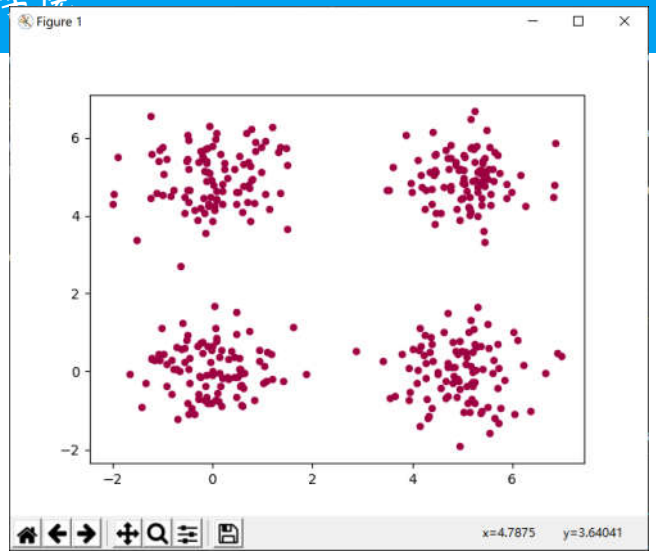
Figure 1
x=4.7875 y=3.64041

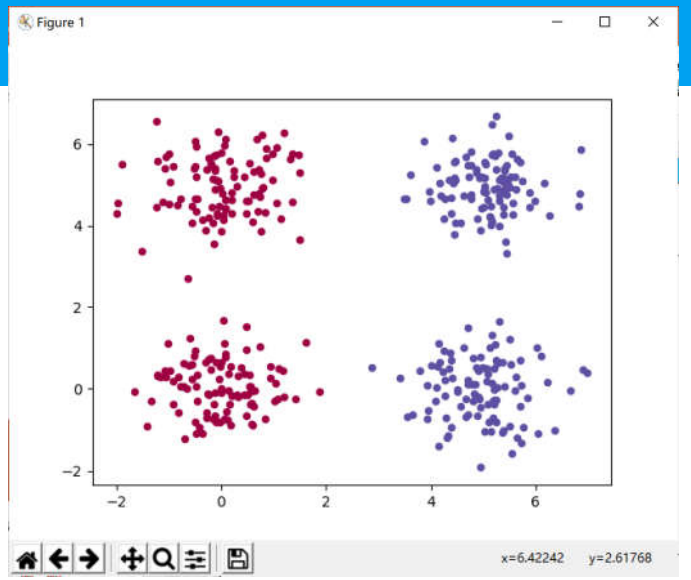
Figure 1
x=6.42242 y=2.61768

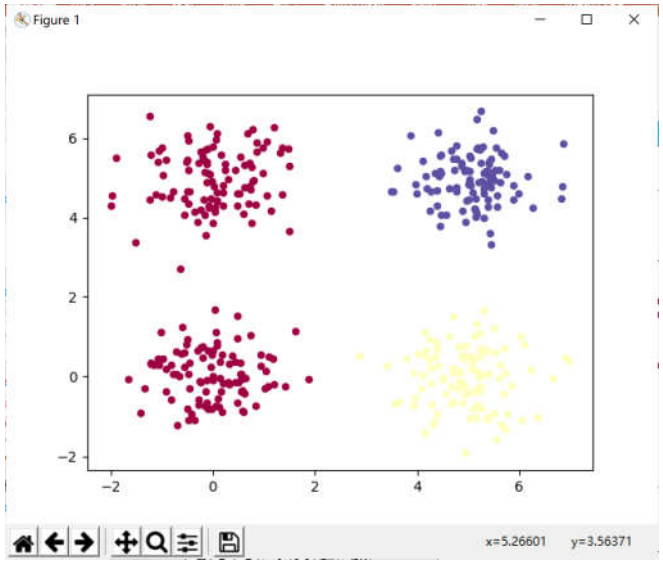
Figure 1
x=5.26601 y=3.56371

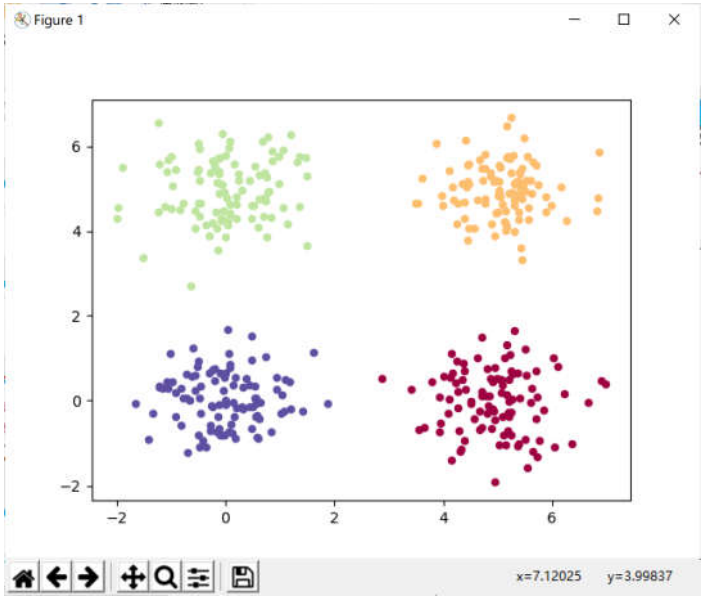
Figure 1
x=7.12025 y=3.99837