中山大学数据科学与计算机学院

移动信息工程专业-数据挖掘

本科生实验报告

（2018-2019 学年春季学期）

课程名称：数据挖掘

[redacted]			
[redacted]			

一、 实验题目

聚类

二、 实验内容

1. 算法原理

（1）K-means

该算法首先随机选择 k 个对象作为初始聚类的质心。对于其余的每一个对象，根据该对象与各聚类质心之间的距离，把它分配到与之最近的质心代表的聚类中。然后，重新计算每个聚类的新质心（该聚类中所有对象的均值）。重复上述过程，直到聚类中心收敛。

该算法的特点是：① k 值要事先确定，k 值的选定难以估计；② 需要不断进行样本分类调整，不断计算调整后的新的聚类质心，当数据量非常大时，算法的时间开销很大；③ 对离散点和 k 值的选取比较敏感；④ 只对线性可分数据有效。

（2）DBSCAN

DBSCAN 是一种基于高密度连通区域的、基于密度的聚类算法，能够将具有足够高度的区域划分成若干个不相交的类簇。该算法首先指定半径 Eps（划定样本点的领域范围）以及密度阈值 MinPts，然后找到没有搜索过的核心点（ 该点半径 Eps 中的样本个数>=MinPts ）标记为一个类，将该核心点领域内的点都划到该核心点的类中。接下来进行领域搜索，如果该领域内某样本点为核心点，则继续搜索该样本点的领域；如果该领域内某样本点为边界点，则不继续搜索。重复上述步骤，直到所有的样本点都已经被成功分类。

该算法的特点：① 可以对任意形状的稠密数据集进行聚类，可以适用于凸样本集，也可以适用于非凸样本集；③ 在聚类的同时可以找出异常，对数据集中的异常点不敏感；④ 聚类结果没有偏倚；⑤ 如果数据集的密度不均匀或聚类间距相差很大时，聚类质量较差；⑥ 不能很好反映高维数据。

（3）谱聚类-Ncut

该算法是基于图的算法，将样本看做无向图中的结点，将这些点用边连接起来。相似度

较高的两个点之间的边的权重较高，相似度低的两个点之间的边的权重较低。通过找到权重最小，又能平衡切出子图大小的边，对所有样本组成的图进行切图。然后不同的子图代表不同的聚类结果。

该算法的特点：① 只需要数据之间的相似度矩阵，因此对于处理稀疏数据的聚类很有效；② 在处理高维数据聚类时的复杂度比传统聚类算法好；③ 如果最终聚类的维度非常高，则谱聚类的运行速度和最后的聚类效果都不好； ④ 聚类效果依赖于相似矩阵，不同的相似矩阵得到的最终聚类效果可能很不同。

2. 伪代码

（1）K-means

```
随机选择 K 个点作为起始质心；
While(聚类中心收敛之前)：
    for 对数据集中的每一个点：
        计算点到每一个质心的距离；
        将点分配到距离其最近的质心代表的簇；
    for 对每一个簇：
        求出该簇点的均值，更新质心为该均值；
    当聚类中心收敛时，分类结束。
```

（2）DBSCAN

```
将数据集 D 中所有样点标记为未访问状态，且簇 id=0；
for 对数据集中的每一个点 p：
    if p 已经访问过或 p 的簇 id!=0:
        continue;
    else:
        标记 p 已经访问过；
        获取 p Eps 领域内的点集合，个数为 num(p)；
        if num(p) <= 0:
            标记 p 的簇 id=-1（噪声点）；
        else if num(p) >=MinPts:
            标记 p 为核心点，簇 id+=1，设置其领域内的所有点的簇 id=p 的簇 id；
            for  p Eps 领域中所有未处理的点 q：
                if num(q) >= MinPts:
                    将 q Eps 领域中的所有未处理的点的簇 id 设置为 p 的簇 id;
```

（3）Ncut

```
① 计算数据集中样本点之间的高斯距离，构建权重矩阵 W；
② 将 W 的每一列元素加起来得到个数，将它们放在对角线上，其余地方为 0，构建归一
```

化矩阵 D；

③ 计算拉普拉斯矩阵 L： L = D – W

④ 归一化拉普拉斯矩阵： L = D^（-0.5）* L * D^(0.5)

⑤ 计算归一化后 L 矩阵的 K 个最小特征值以及对应到的特征向量（将 K 个特征向量竖着并排放在一起，形成一个特征矩阵 Q）；

⑥ 对特征矩阵 Q 作 Kmeans 聚类，得到一个向量 C（分别对应 W 中每一行所代表的样本点所属的类别，即聚类结果）。

PS：

③④ 可以得到 L = D^（-0.5）* L * D^(0.5) = I - D^（-0.5）* W * D^(0.5)

3. 关键代码截图（带注释）

（1）K-means

① 随机选择数据集中的 k 个点作为初始聚类的质心：

```
% 指定簇心初始位置：随机选择k个数
clusters_center = zeros(k, dimension);
for i=1:k
    clusters_center(i,:)=data(randi(num,1),:);
end
```

② 遍历所有样本点，计算该样本到每个质心的距离，设置该样本点的簇为距离最小的质心代表的簇：

```
%记录该样本到每个质心的距离
distance = zeros(k,1);
for j=1:k
    %计算该样本到每个质心的欧式距离
    distance(j,1) = norm(data(i,:)-clusters_center(j,:));
end
%找出最小的距离
[min_dis, row] = min(distance);
c(i,:) = row;
```

③ 更新质心，判断是否收敛：

```
for i=1:k
    total_dis = 0;
    total_num = 0;
    for j=1:num
        total_dis = total_dis + (c(j,:)==i)*data(j,:);
        total_num = total_num + (c(j,:)==i);
    end
    new_cluster = total_dis/total_num;
    %当更新的质心和原来的质心的距离小于PRECISION，则认定为收敛
    if(norm(clusters_center(i,:)-new_cluster) < PRECISION )
        converagence = 1;
    end
    %更新质心
    clusters_center(i,:) = new_cluster;
end
```

（2）DBSCAN

① 设置参数半径 Eps 和密度阈值 MinPts：

```
%定义参数Eps和MinPts
MinPts = 6;
Eps = 5;
```

② 计算数据集中点与点之间的距离：

```
%计算矩阵中点与点之间的欧式距离
all_dis = zeros(num, num);
for i=1:num
    for j=i:num
        all_dis(i,j) = norm(data(i,:)-data(j,:));
        all_dis(j,i) = all_dis(i,j);
    end
end
```

③ 遍历处理每个样本，如果这个点是没有处理过的，则取得该点到其他所有点的距离，并且找出半径 Eps 内的所有点。

```
%找到没处理过的点
if visited(i)==0
    %取得该点到其它所有点的距离
    dis = all_dis(i,:);
    %找到半径EPS内的所有点
    Eps_point = find(dis <= Eps);
```

④ 根据半径内的点的个数区分点的类型：

```
%根据点数区分点的类型
%噪声点
if length(Eps_point)==1
    types(i) = -1;
    class(i) = -1;
    visited(i) = 1;
end
%边界点
if length(Eps_point)>1 && length(Eps_point) < MinPts+1
    types(i) = 0;
    class(i) = 0;
end
```

如果是核心点，则让该核心点代表一个簇，将该核心点的领域中的点都划到该簇中：

```
%核心点
if length(Eps_point) >= MinPts+1
    visited(i) = 1;
    cluster_num = cluster_num+1;
    types(i) = 1;
    %将该核心点EPS范围内的点都划到同一个簇
    class(Eps_point) = cluster_num;
```

遍历该样本点领域中的未处理过的点，如果该点是核心点，则该点领域中的点也划到该样本点的簇中：

```
while ~isempty(Eps_point)
    %取Eps_point第一行序号的样本
    point = data(Eps_point(1),:);
    visited(Eps_point(1))=1;
    Eps_point(1)=[];
    dis = all_dis(point(1,1),:);
    eps_point = find(dis<=Eps);
    %处理非噪声点
    if length(eps_point) >1
        class(eps_point(:,1))=cluster_num;
        visited(eps_point(:,1))=1;
        if length(eps_point) >= MinPts+1
            types(point(1,1))=1;
        else
            types(point(1,1))=0;
        end

for j=1:length(eps_point)
    if visited(eps_point(j))==0
        Eps_point = [Eps_point eps_point(1)];
    end
end
```

（3）Ncut

① 计算数据集中样本点之间的高斯距离，构建权重矩阵 W：

```
%构建权重矩阵——高斯距离
data = data/max(max(abs(data))); %归一化
sigma = 1;
dis_fir = pdist(data); %data行与行之间的距离
dis_zero = squareform(dis_fir); % 对角线化为0，ij代表第xi和xj的距离 ||xi-xj||
dis_double = dis_zero.*dis_zero; %||xi-xj||^2
top = -dis_double/(2*sigma*sigma);
res = spfun(@exp,top);
S = full(res);
W = S;
```

②构建归一化矩阵 D：

```
%计算归一化矩阵D
D = full(sparse(1:num,1:num,sum(W)))
```

③ 归一化的拉普拉斯矩阵 L ：

```
% L=D-W, 归一化L=D^(-1/2) * L * D^(-1/2) = I-D^(-1/2) * W * D^(-1/2)
L =eye(num)-(D^(-1/2)*W*D^(-1/2));
```

⑤ 计算归一化后 L 矩阵的 K 个最小特征值以及对应到的特征向量：

```
%找特征值特征向量T并排序，找前K个
K = 3;
%'sm' 绝对值最小特征值
[T,~] = eigs(L, K, 'SM');
```

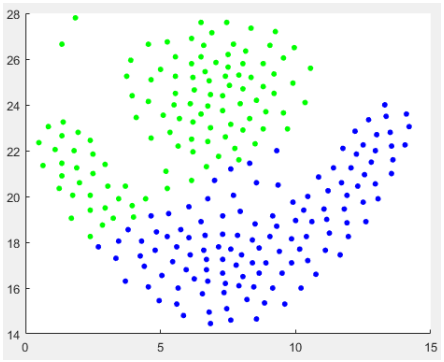
⑥ 对特征向量进行 Kmeans 聚类:

```
c = kmeans(T,K);
```

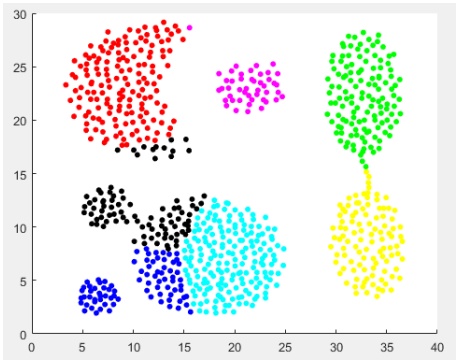
三、 实验结果及分析

（1）K-means

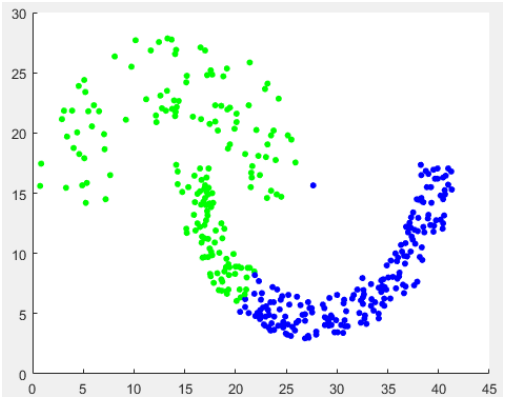
flame_cluster=2.txt



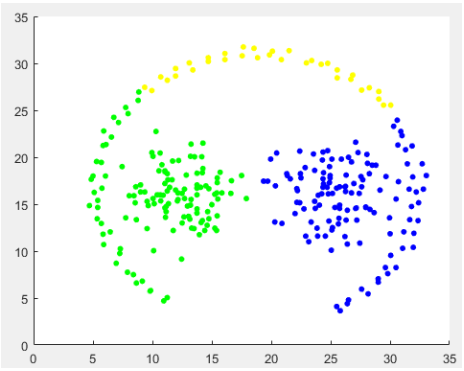
Aggregation_cluster=7.txt



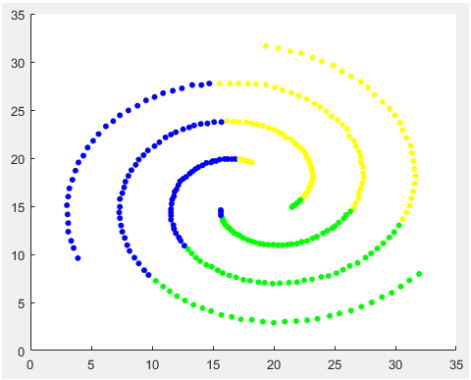
Jain_cluster=2.txt ,



Pathbased_cluster=3.txt



Spiral_cluster=3.txt



Mfeat

data_fou:
聚类准确率ACC为0.7295
标准化信息NMI为0.68284
聚类纯度PUR为0.736

data_fac:
聚类准确率ACC为0.6665
标准化信息NMI为0.63599
聚类纯度PUR为0.7555

data_kar:
聚类准确率ACC为0.721
标准化信息NMI为0.72342
聚类纯度PUR为0.8025

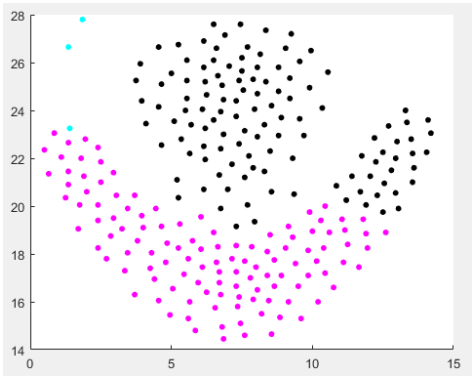
data_pix:
聚类准确率ACC为0.7115
标准化信息NMI为0.72132
聚类纯度PUR为0.7965

data_zer:
聚类准确率ACC为0.5355
标准化信息NMI为0.50077
聚类纯度PUR为0.611

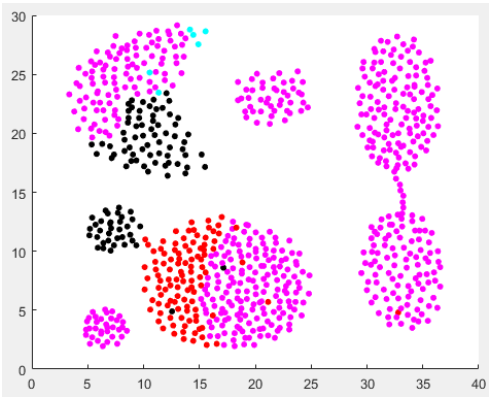
data_mor:
聚类准确率ACC为0.3935
标准化信息NMI为0.46738
聚类纯度PUR为0.527

（2）DBSCAN

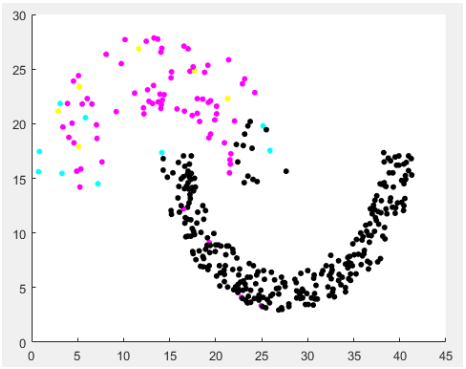
flame_cluster=2.txt



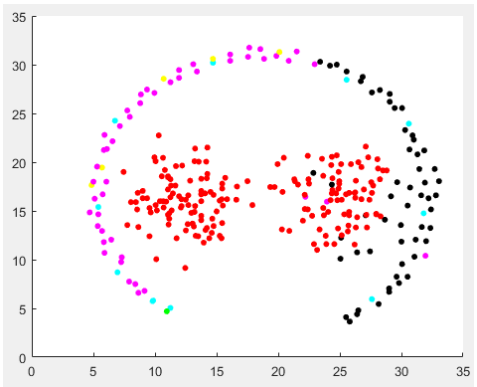
Aggregation_cluster=7.txt



Jain_cluster=2.txt ,

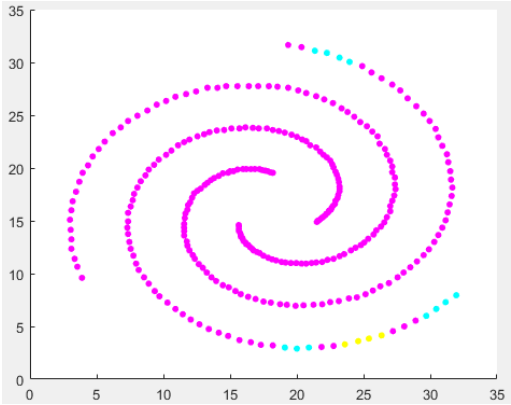


Pathbased_cluster=3.txt



Minpts=5 Eps=80

Spiral_cluster=3.txt



Mfeat

data_fou:
聚类准确率ACC为0.368
标准化信息NMI为0.63834
聚类纯度PUR为0.368

data_fac:
聚类准确率ACC为0.1
标准化信息NMI为0
聚类纯度PUR为1

data_kar:
聚类准确率ACC为0.363
标准化信息NMI为0.54584
聚类纯度PUR为0.3905

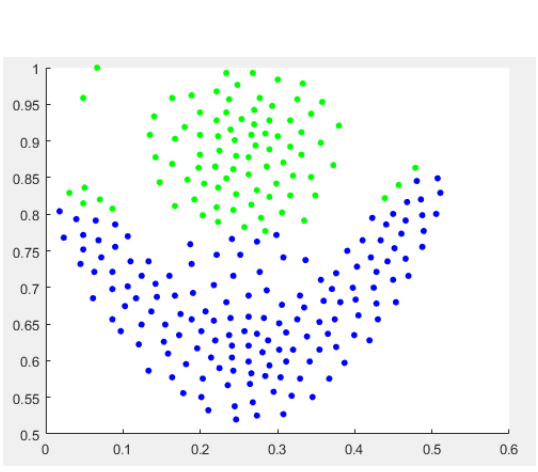
data_pix:
聚类准确率ACC为0.3915
标准化信息NMI为0.55335
聚类纯度PUR为0.396

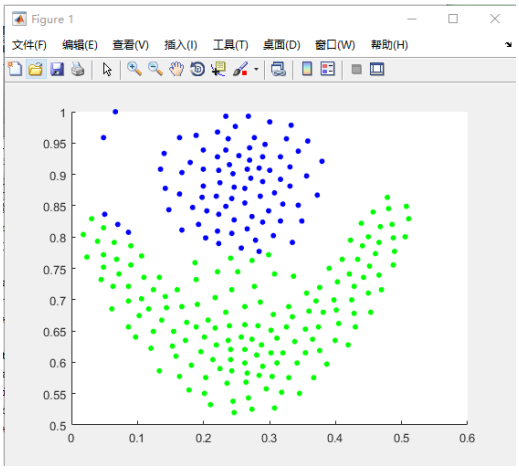
data_zer:
聚类准确率ACC为0.1
标准化信息NMI为0
聚类纯度PUR为1

data_mor:
聚类准确率ACC为0.42
标准化信息NMI为0.55625
聚类纯度PUR为0.42

（3）Ncut

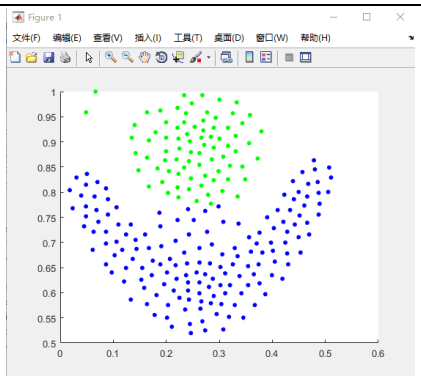
flame_cluster=2.txt

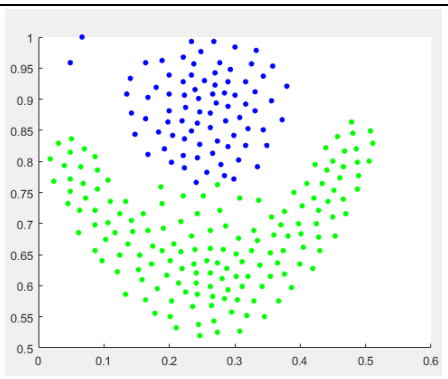




Sigma=0.1

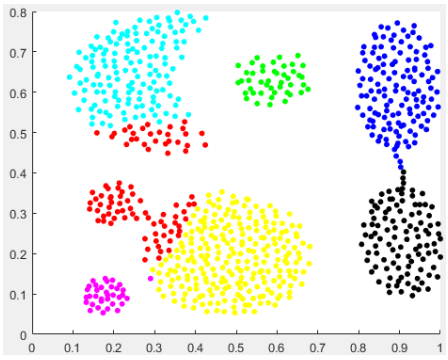
sigma=0.09

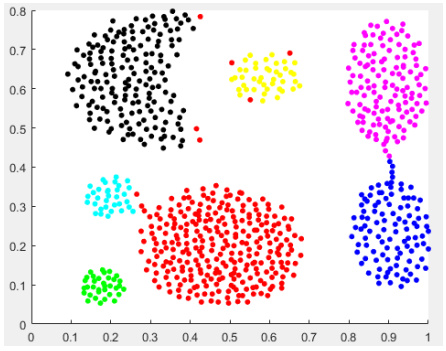




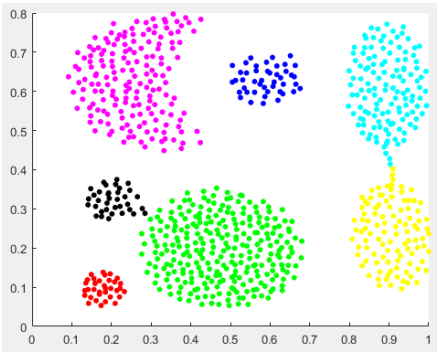
Sigma = 0.08　　sigma =0.07

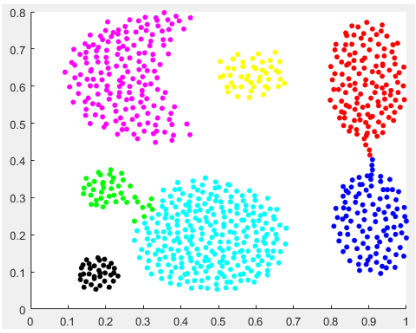
Aggregation_cluster=7.txt





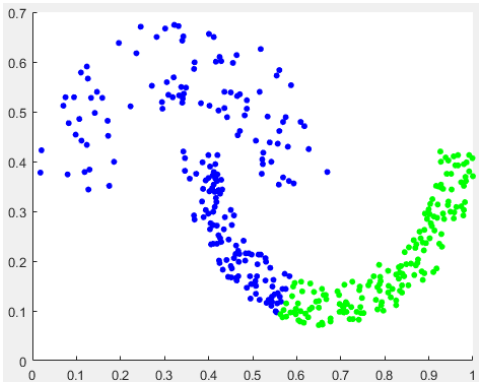
Sigma =0.1　　sigma = 0.01

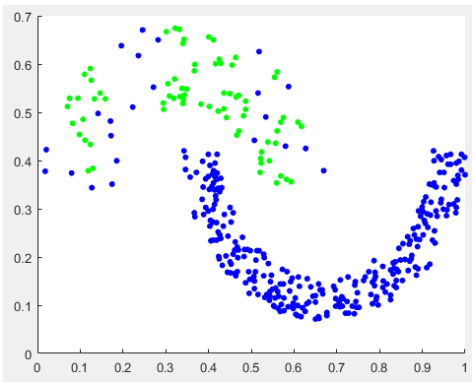




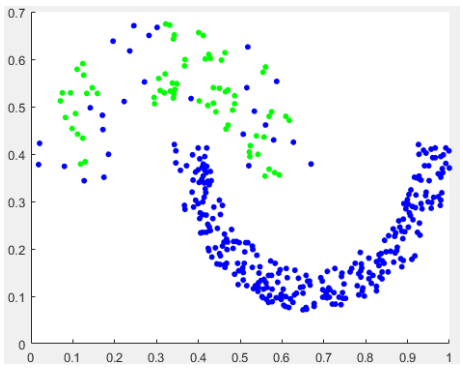
Sigma = 0.05　　Sigma = 0.08

Jain_cluster=2.txt ,

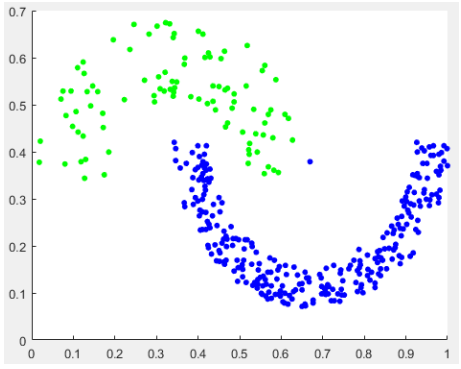




Sigma=0.1　　sigma=0.01

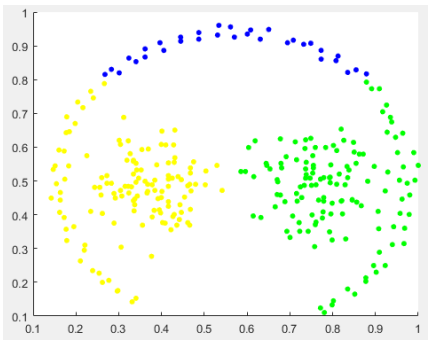


sigma = 0.09

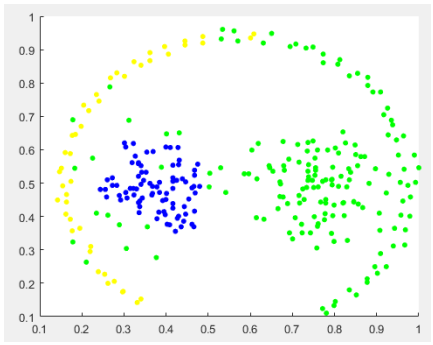


sigma=0.05

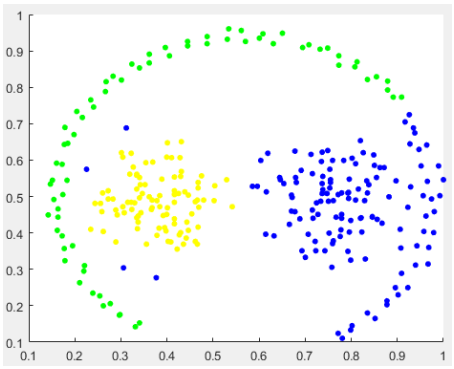
Pathbased_cluster=3.txt



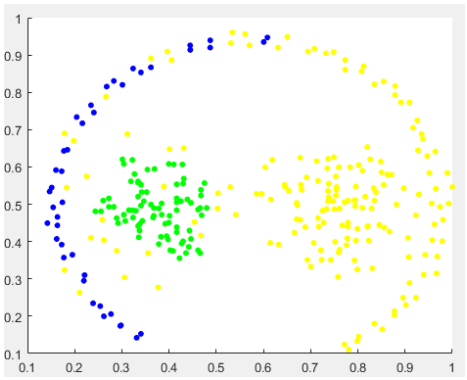
Sigma=0.1



Sigma=0.01

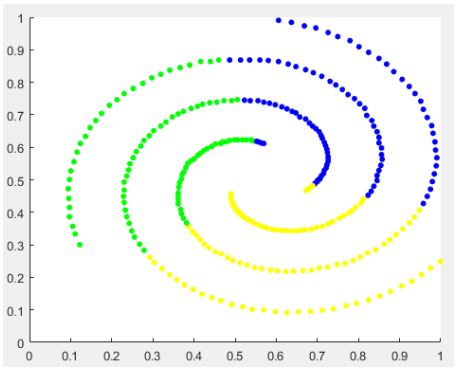


Sigma=0.2

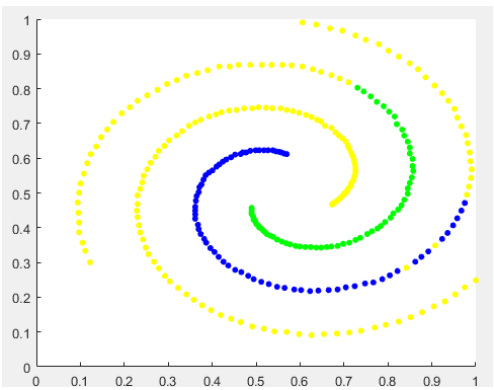


sigma =0.009555

Spiral_cluster=3.txt



Sigma = 0.1



Sigma = 0.1

Mfeat

data_fou:

聚类准确率ACC为0.7055
标准化信息NMI为0.64132
聚类纯度PUR为0.7095

Sigma = 1.5

data_fac:

聚类准确率ACC为0.6065
标准化信息NMI为0.614
聚类纯度PUR为0.6935

Sigma = 1.5

data_kar:

聚类准确率ACC为0.6695
标准化信息NMI为0.64933
聚类纯度PUR为0.7145

sigma = 1

data_zer:

聚类准确率ACC为0.541
标准化信息NMI为0.47492
聚类纯度PUR为0.5655

sigma=1.5

data_pix:

聚类准确率ACC为0.674
标准化信息NMI为0.64082
聚类纯度PUR为0.7835

sigma=1.5

聚类准确率ACC为0.686
标准化信息NMI为0.68474
聚类纯度PUR为0.7605

sigma=2

data_mor:

聚类准确率ACC为0.3905
标准化信息NMI为0.45777
聚类纯度PUR为0.6355

sigma=0.01