

目录

非监督聚类算法	1
1 k-means 算法	1
1.1 问题描述.....	1
1.2 算法描述.....	1
1.3 仿真结果.....	1
1.4 结果分析.....	3
2 分层聚类法.....	4
2.1 问题描述.....	4
2.2 算法描述.....	4
2.3 仿真结果.....	4
2.4 结果分析.....	6

非监督聚类算法

1 k-means 算法

1.1 问题描述

对给定数据进行 kmeans 聚类，令 $k=2,3,4$ 。画出聚类结果及每类的中心点，观察聚类结果。记录使用不同初始点时的聚类结果，收敛迭代次数及误差平方和。

1.2 算法描述

k-means 算法的算法流程如下：

- 1) （由人）适当选取 K 个初始类中心 $Z_1(1), Z_2(1), \dots, Z_K(1)$
- 2) 在第 k 次迭代中，样本 $\{X\}$ 按照如下方法分配到 K 个类中：如果 $\|X - Z_j(k)\| = \min \|X - Z_i(k)\|, (i=1, 2, \dots, K)$ ，则 $X \in S_j(k)$ ，其中 $S_j(k)$ 是以 $Z_j(k)$ 为中心的类的样本集
- 3) 令在第 2 步得到的类 $S_j(k)$ 的新的类中心为 $Z_j(k+1)$ ，取各类的误差平方和准则为

$$J_i = \sum_{X \in S_j(k)} \|X - Z_j(k+1)\|^2, j=1, 2, \dots, K$$

计算 $Z_j(k+1)$ 使 J_i 最小，即

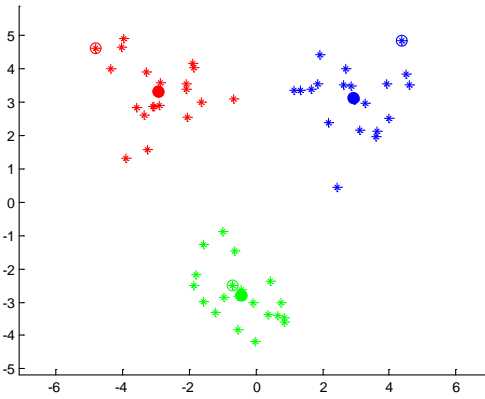
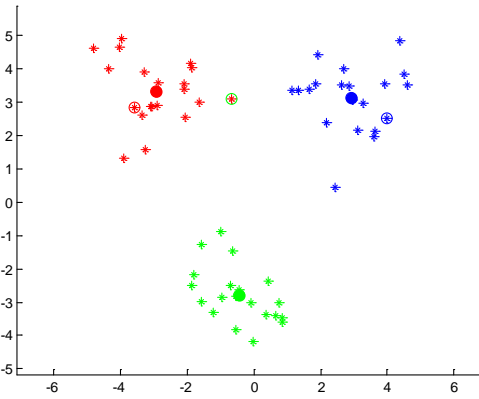
$$Z_j(k+1) = \frac{1}{N_j} \sum_{X \in S_j(k)} X, j=1, 2, \dots, K$$

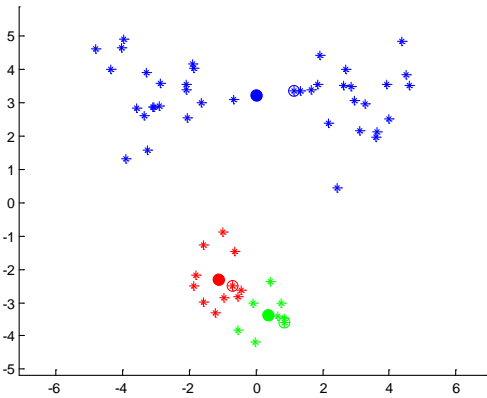
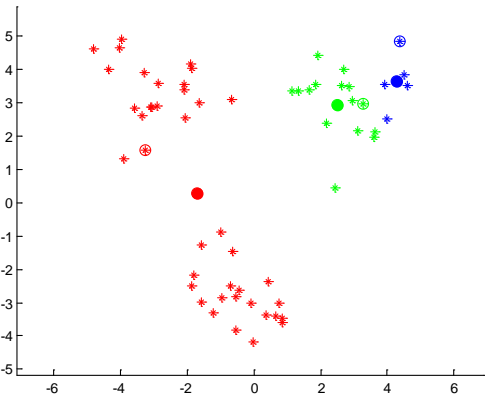
- 4) 对所有的 $j=1, 2, \dots, K$ ，如果都有 $Z_j(k+1) = Z_j(k)$ ，则算法结束，否则继续 2。

1.3 仿真结果

当 $K=3$ 时，分别使用给定的初始点，得到的结果如表 1 所示。

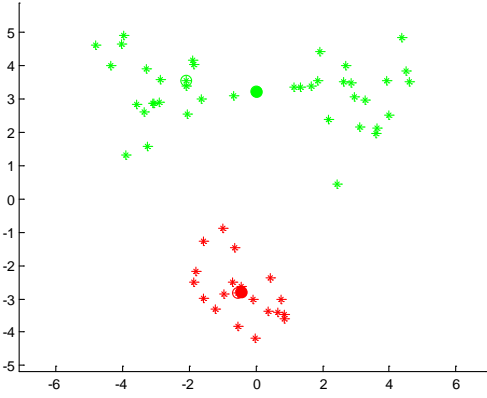
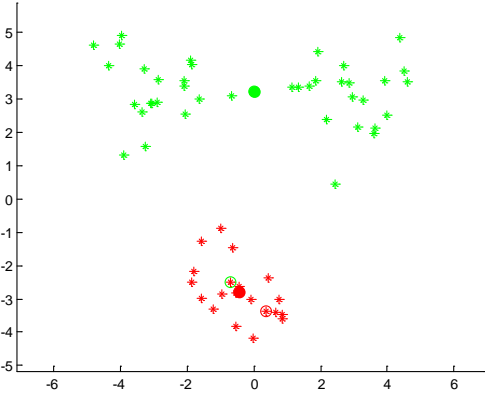
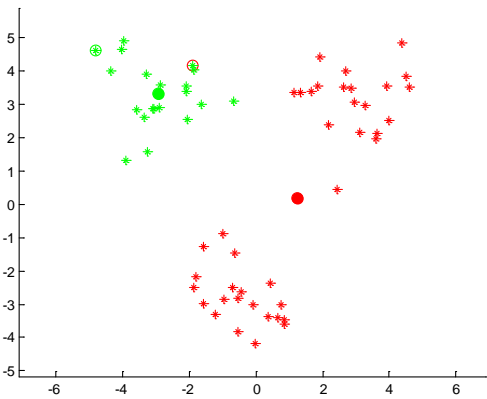
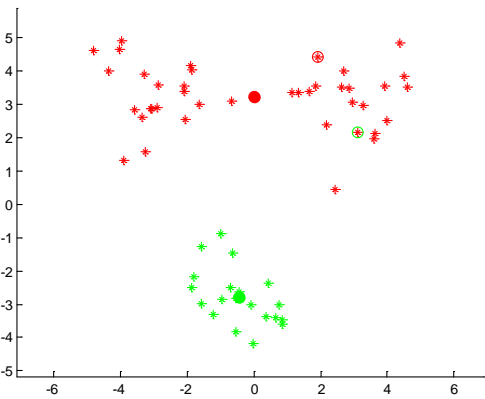
表 1 $K=3$ 时各组指定初始点对应的聚类结果

初始点组 1 [-4.822 4.607;-0.7188 -2.493;4.377 4.864]	初始点组 2 [-3.594 2.857;-0.6595 3.111;3.998 2.519]
<p>k = 3, given seeds 1</p> 	<p>k = 3, given seeds 2</p> 
迭代次数: 3	迭代次数: 3
误差平方和: 106.7495	误差平方和: 106.7495

初始点组 3 [-0.7188 -2.493;0.8458 -3.59;1.149 3.345]	初始点组 4 [-3.276 1.577;3.275 2.958;4.377 4.864]
	
迭代次数: 3 误差平方和: 436.6295	迭代次数: 4 误差平方和: 527.3038

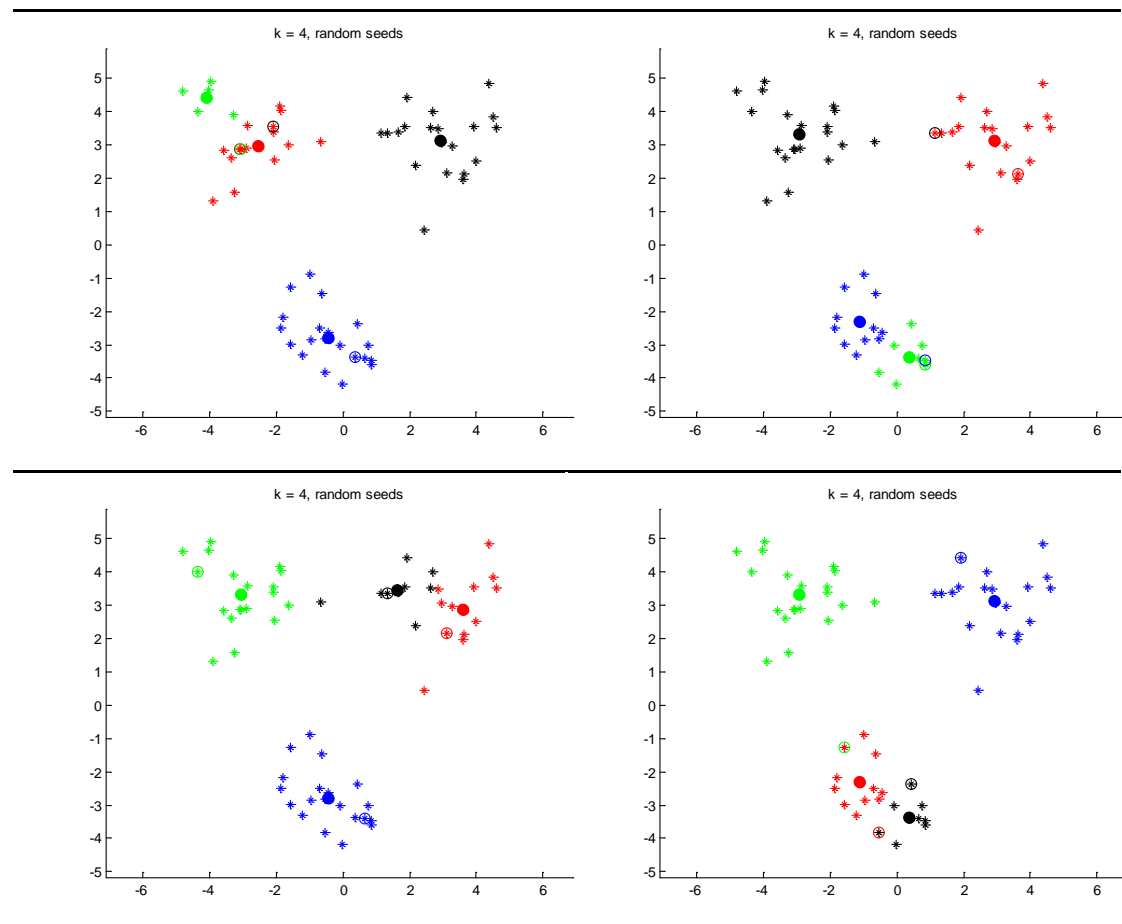
当 $K = 2$ 时，随机选取初始点，得到的结果如表 2 所示。

表 2 $K=2$ 时的聚类结果（随机初始点）

当 $K = 4$ 时，随机选取初始点，得到的结果如表 3 所示。

表 3 K=4 时的聚类结果（随机初始点）



1.4 结果分析

对上述结果分析如下。

- 1) 原始数据集，人为主观来看大约有 3 个聚类，但上面两个聚类的距离稍近，可以预料到，当 $K=3$ 时，如果选取的初始点比较接近上述两个类的中心位置，很可能会将上面两个聚成一类，而将下面的数据分成两类。初始点组 3 的结果证明了这一点。
- 2) 综合对比 $K=3$ 的四个结果可知，k-means 算法对初始值的选取比较敏感，不同的初始值往往导致不同的聚类结果。
- 3) $K=2$ 与 $K=4$ 也可以视为不同层次的聚类。通过表 2 和表 3 的结果可以也看出，k-means 算法对初始值的选取比较敏感。另一方面，不同的初始点选取有时可以得到相同的聚类结果。

对 k-means 的优缺点分析：

- 1) k-means 算法思路比较简单，空间需求适中，只需要存放数据点和类中心，时间上基本与数据点个数线性相关。
- 2) k-means 受初始点影响较大，如果数据中有离群点，k-means 效果会受到很大影响。此外，k-means 比较适合球形簇，对于复杂形状的簇识别能力有限。
- 3) 可以将层次聚类技术和 k-means 结合，或者在选取初始点时考虑数据点所在区域的密度，这样可以较好地避免 k-means 易受初始点或离群点影响。

2 分层聚类法

2.1 问题描述

对于给定的可用高斯分布近似的两个样本集：

- 1) 利用最小错误概率分类判则进行分类
- 2) 利用层次聚类法进行聚类

2.2 算法描述

最小错误概率分类算法已在上次作业中介绍，本次作业不再赘述。

层次聚类法的基本思想描述如表 4 所示。

表 4 层次聚类算法描述

基本层次聚类算法
<ol style="list-style-type: none"> 1. 如果需要，计算临近度（或距离）矩阵 2. Repeat 3. 合并最接近的两个簇 4. 更新临近度（或距离）矩阵，以反映新的簇与原来的簇之间的临近性 5. Until 只剩下 1 个（或指定个数）簇

本次实验中，计算的是簇之间的距离矩阵，且两个簇之间的距离定义如式(1)所示。

$$d_{\max}(S_i, S_j) = \max_{\substack{X_i \in S_i \\ X_j \in S_j}} \|X_i - X_j\| \quad (1)$$

同时，为了减少算法计算量，仿真时首先计算了所有数据点之间的距离矩阵，避免之后的重复计算。

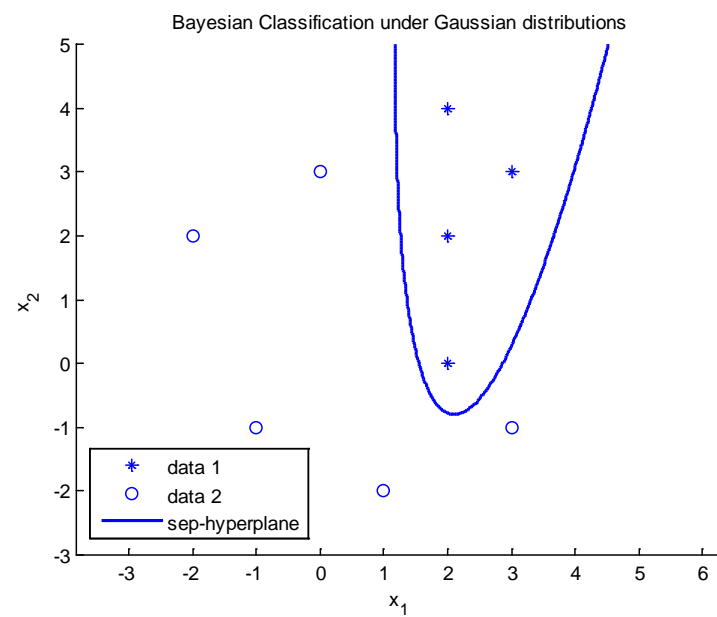
2.3 仿真结果

最小错误概率分类的结果如表 5 和图 1 所示。

表 5 识别函数与识别界面

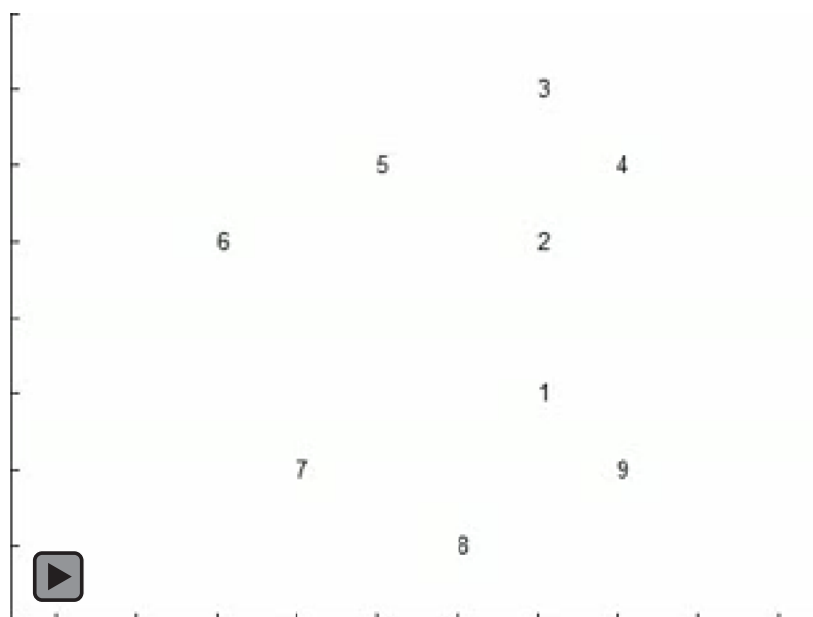
$d1(x1, x2) = (-2.916667)*x1^2 + (-0.250000)*x2^2 + (0.500000)*x1*x2 + (12.000000)*x1 + (-0.000000)*x2 + (-13.702733)$
$d2(x1, x2) = (-0.222749)*x1^2 + (-0.175355)*x2^2 + (-0.194313)*x1*x2 + (0.127962)*x1 + (0.109005)*x2 + (-1.783335)$
$(-2.693918)*x1^2 + (-0.074645)*x2^2 + (0.694313)*x1*x2 + (11.872038)*x1 + (-0.109005)*x2 + (-11.919398) = 0$

图 1 分类结果



层次聚类的结果如图 2 所示（也可通过附件中的 gif 查看）。

图 2 层次聚类的结果



各个数据点合并的过程如下：

4: 4 --> 2

9: 8 --> 1

3: 3 --> 2

6: 4 --> 3

8: 5 --> 1

7: 4 --> 1

5: 3 --> 2

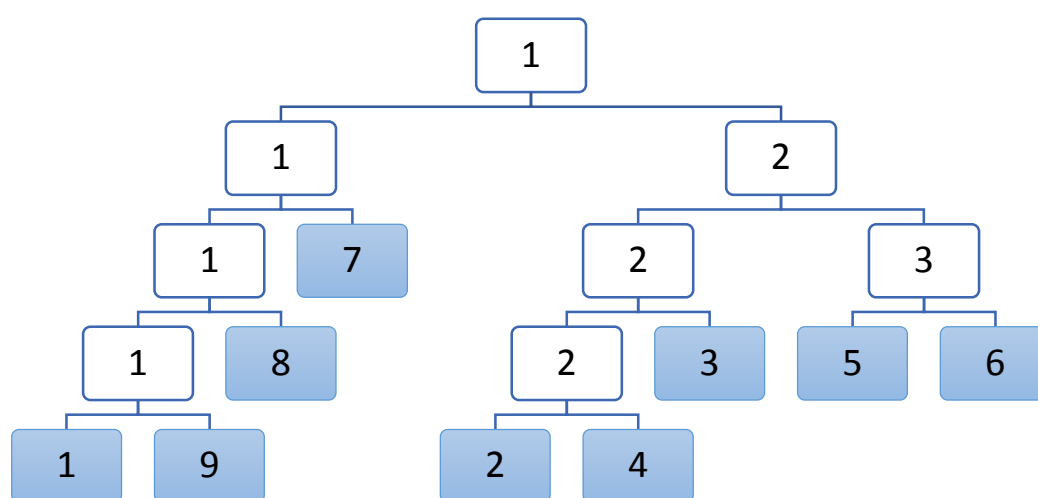
6: 3 --> 2
 2: 2 --> 1
 3: 2 --> 1
 4: 2 --> 1
 5: 2 --> 1
 6: 2 --> 1

2.4 结果分析

最小错误概率分类的结果不再详细分析，可以看出，在分类时，该算法考虑了数据点的分布特征。

对于层次聚类，根据仿真结果，可以得到如所图 3 示的层次聚类图。

图 3 层次聚类结果



对层次聚类算法的分析：

- 1) 层次聚类算法能够很好的识别不同粒度上的簇的聚集，能够有效避免人工指定簇的数目造成的影响。
- 2) 合并簇的操作中，由于使用了所有点的相似/相异信息，合并算法趋向于做出局部最优的决策。随着合并的进行，已经做出的合并决策无法撤回，因此，这种方法阻碍了局部最优标准变为全局最优。
- 3) 层次聚类的算法运算量是明显高于 k-means 的，可以将 k-means 与层次聚类相结合，如在层次聚类时，先使用 k-means 进行部分聚类，这样可以有效减少运算量，也能一定程度上缓解噪声的影响。