聚类算法

Machine Learning Engineer
机器学习工程师
讲师：Ivan

稀牛学院 網易 NETEASE

目 录

CONTENTS

01

无监督学习：聚类介绍

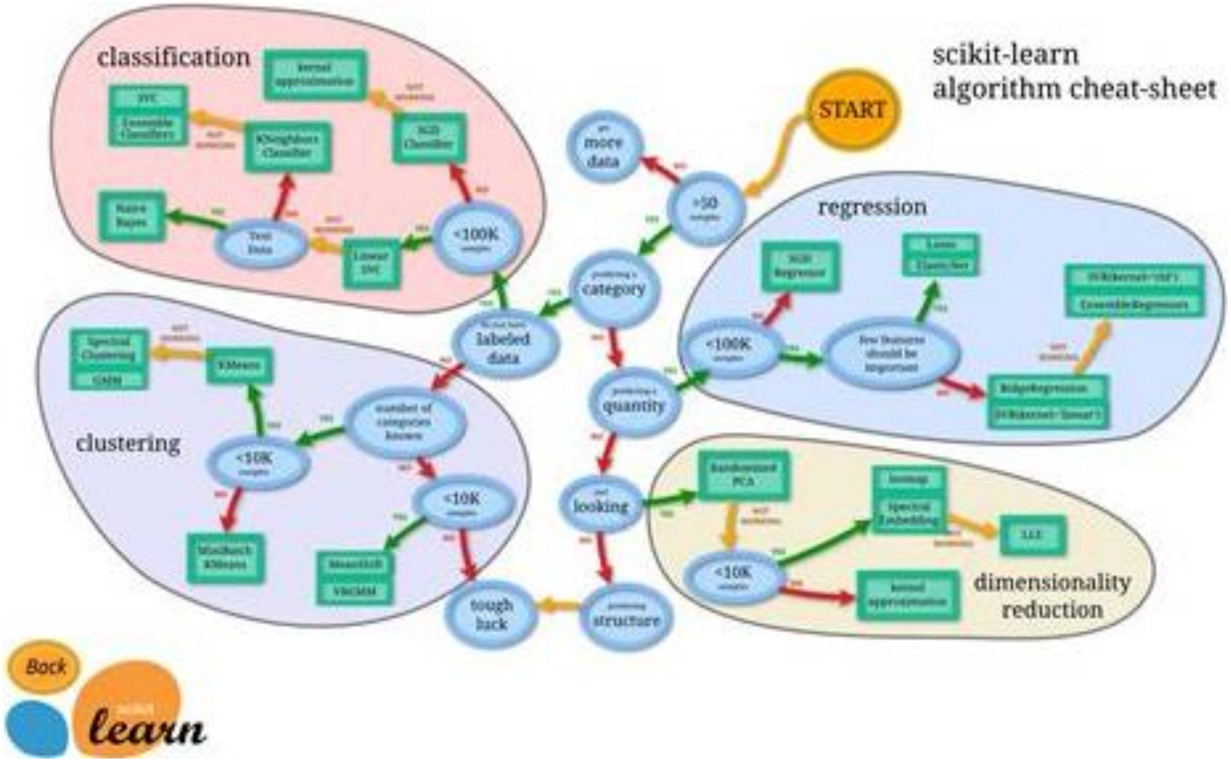
1.1 什么是聚类?

1.2 聚类算法的应用

1.1 什么是聚类?

什么是聚类(clustering)?

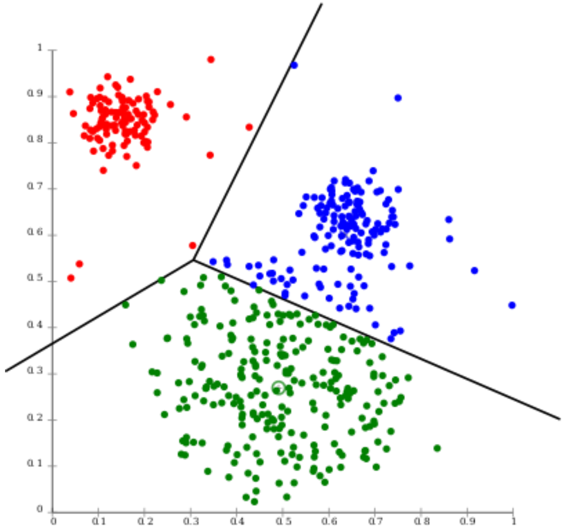
1. 无监督学习(不需要标签)
2. 按照相似性/结构性组织数据
3. 典型应用：
   - 数据压缩
   - 图像分割
   - 数据层次化组织
   - 数据预分类
   - ……

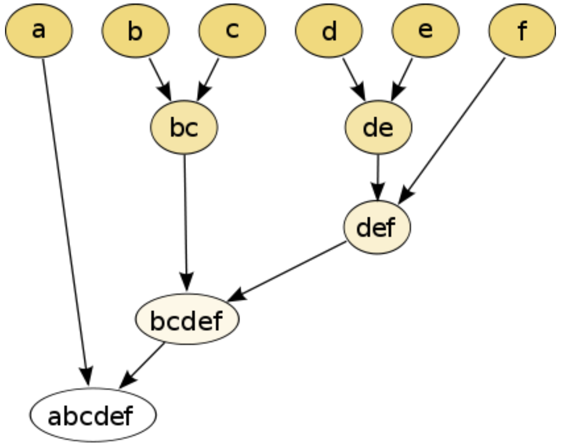


1.1 什么是聚类?

聚类算法大体上可以分成两类：

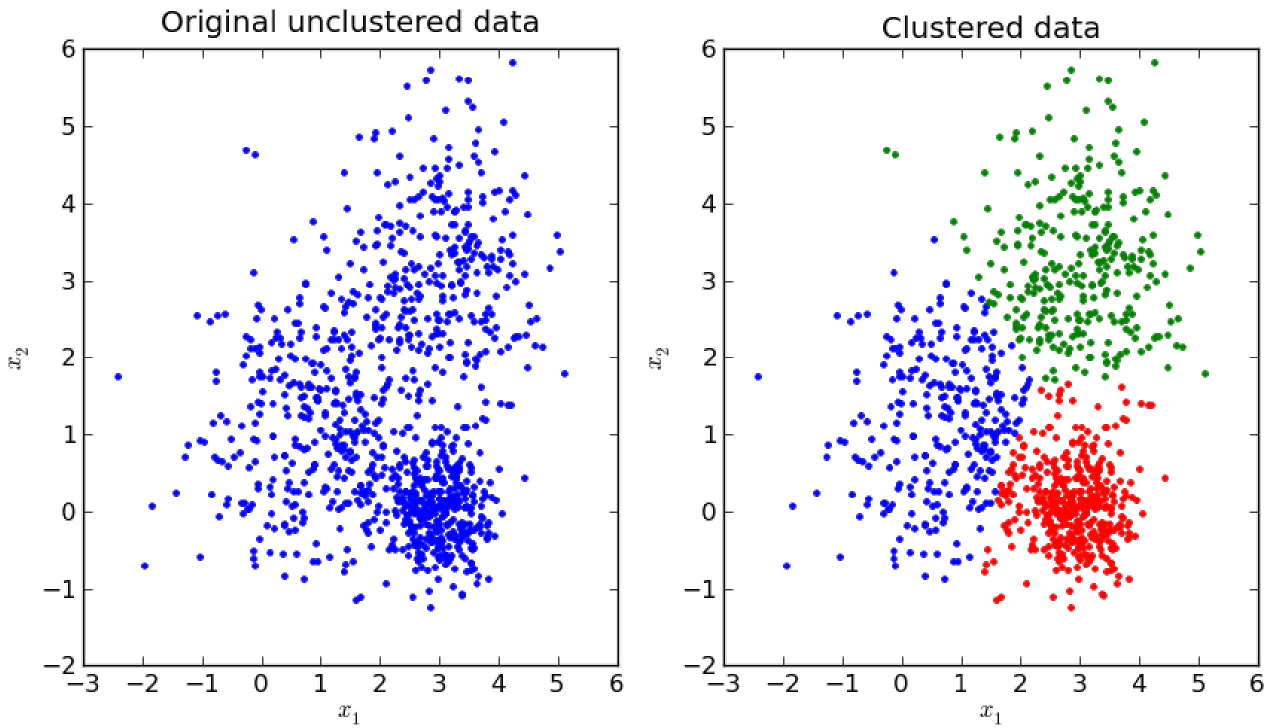
1. **Partitioning Clustering:**
    - K-means/K-medoids
    - Gaussian Mixture Model (高斯混合模型)
    - Spectral Clustering (谱聚类)
    - Centroid-based Clustering...
2. **Hierarchical Clustering:**
    - Single-linkage
    - Complete-linkage
    - Connectivity-based Clustering...





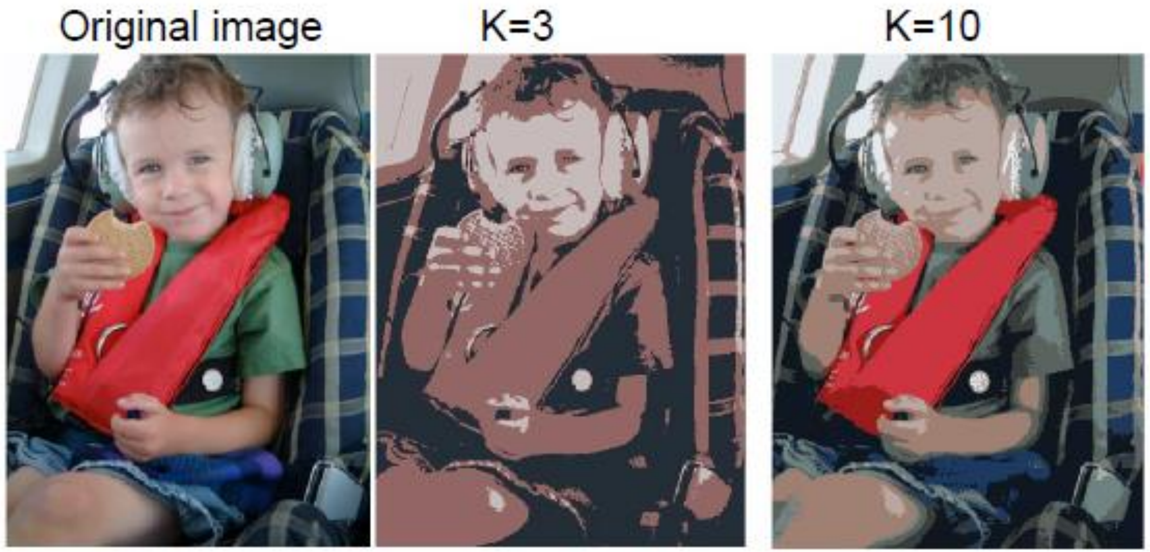
1.2 聚类算法的应用

数据压缩：



1. 对数据按照相似性进行组织：相似的数据在一个类中，距离较远的数据在不同类中
2. 每个类中只需要存储一个代表元

数据压缩：



1. 原图像包括240x180=43,200像素，每个像素包含{R, G, B}三个值，每个8bits
2. 原图像大小：43,200x8x3 = 1,036,800 bits
3. 聚类后数据压缩：直接存储K个类的代表元，每个8x3 = 24bits
4. 每个像素点存储类别分配：$\log_2 K$ bits
5. 压缩后图片需要 86,472 (K=3), 173,040 (K=10) bits，压缩率: 8.3%，16.7%

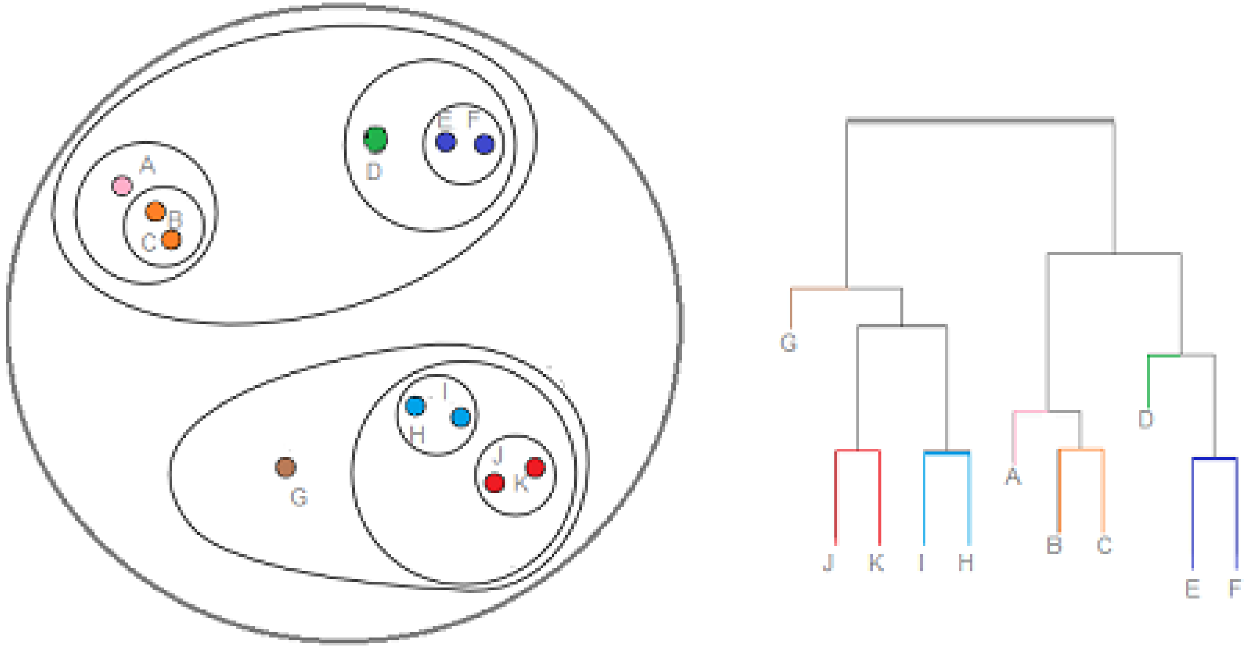
聚类算法的应用

图像分割：

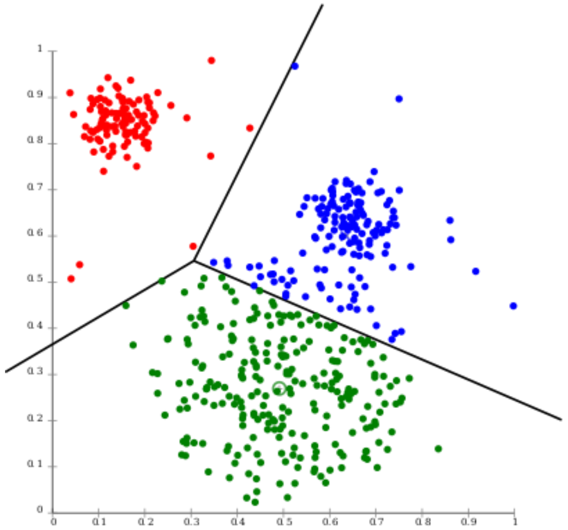
1. 将一幅图像分割成不同区域，理想情况下每一个区域对应图像中一个物体
2. 图像中每一个像素点是一个3维向量，对应{R, G, B}像素值
3. 给定聚类中类别个数K，算法用K个不同的颜色来表示原来的图像，每个像素点用K个颜色中一个表示

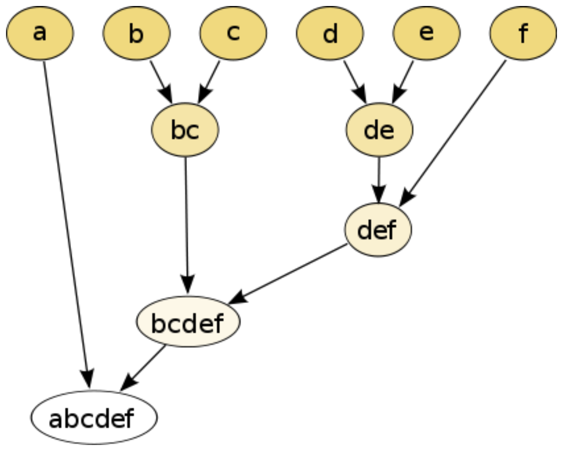
1.2 聚类算法的应用

数据层次化组织：



- 对数据按照不同的粒度进行聚类划分
- Cluster具有嵌套结构：nested clusters
- 应用：文档/新闻聚类，商品聚类，……
- 输出：二叉树，每个内部节点代表一个Cluster，每个叶节点代表一个数据





要点总结

1.1 聚类算法的目的

1.2 Partitioning/Hierarchical 聚类

1.3 聚类算法的应用

02 K-means/K-medoids 算法

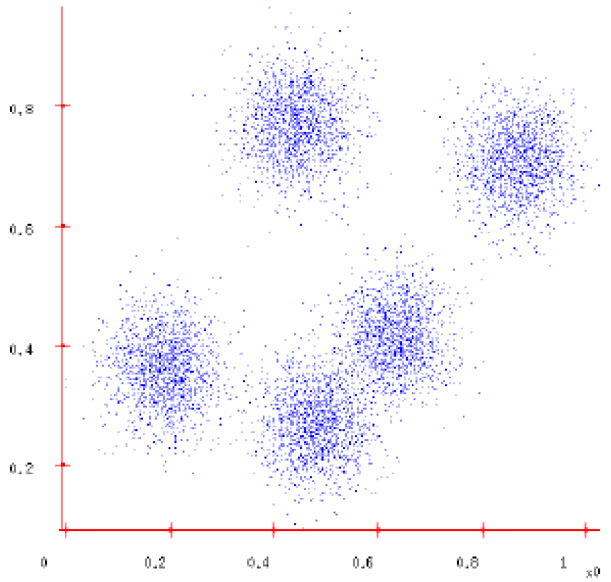
2.1 K-means 算法

2.2 K-medoids 算法

2.1 K-means 算法

目标：将n个数据点分成k类

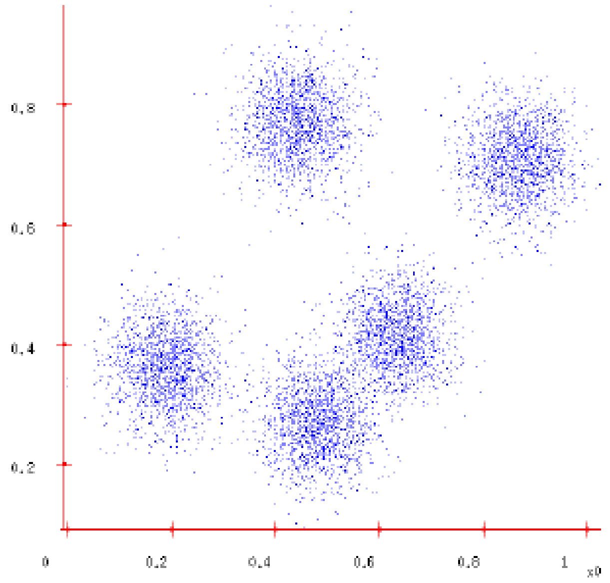
- 如何定义这个问题?
- 如何选择k?



2.1 K-means 算法

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \ldots, n \subseteq R^d$
- 给定：$k \in N$



直观理解：

- 寻找k个聚类中心，使得数据到聚类中心的距离最小
- 将每个数据点分配到距离最近的聚类中心

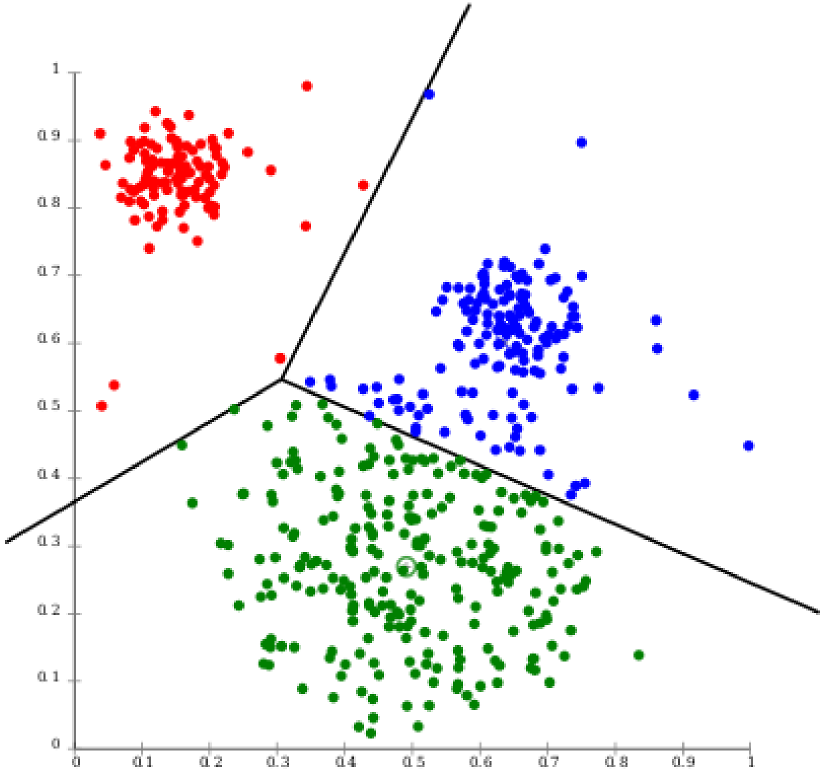
$$\min_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$

2.1 K-means 算法

目标：将n个数据点分成k类

$$\min_{\mu_1,\ldots,\mu_k} \quad \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$

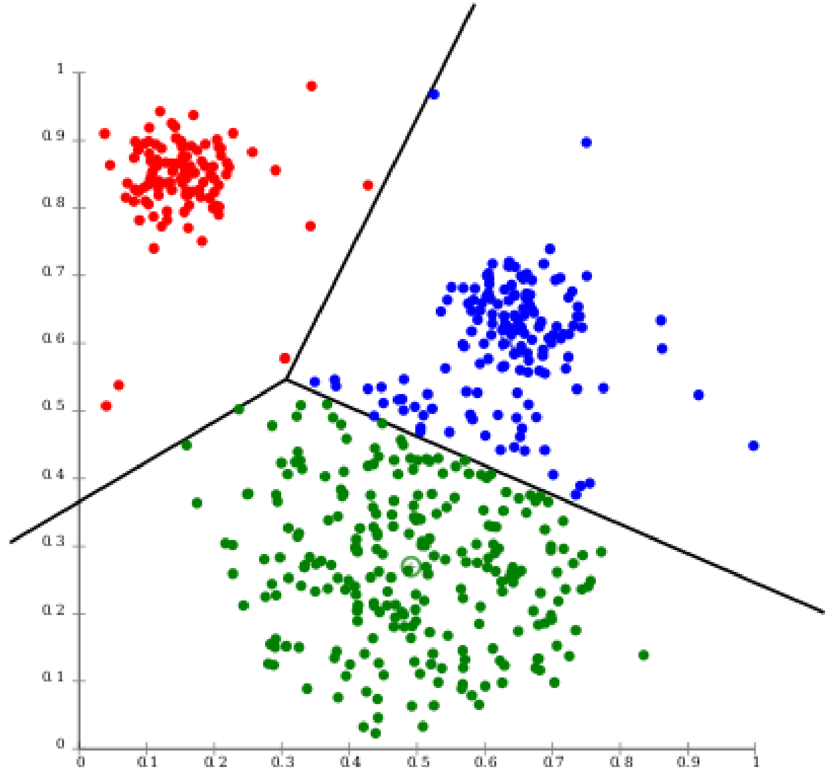
- 如何对距离数据与中心之间的距离进行度量?
- 如何最优地选择中心点?
- 聚类结果如何?



2.1 K-means 算法

目标：将n个数据点分成k类

$$\min_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$



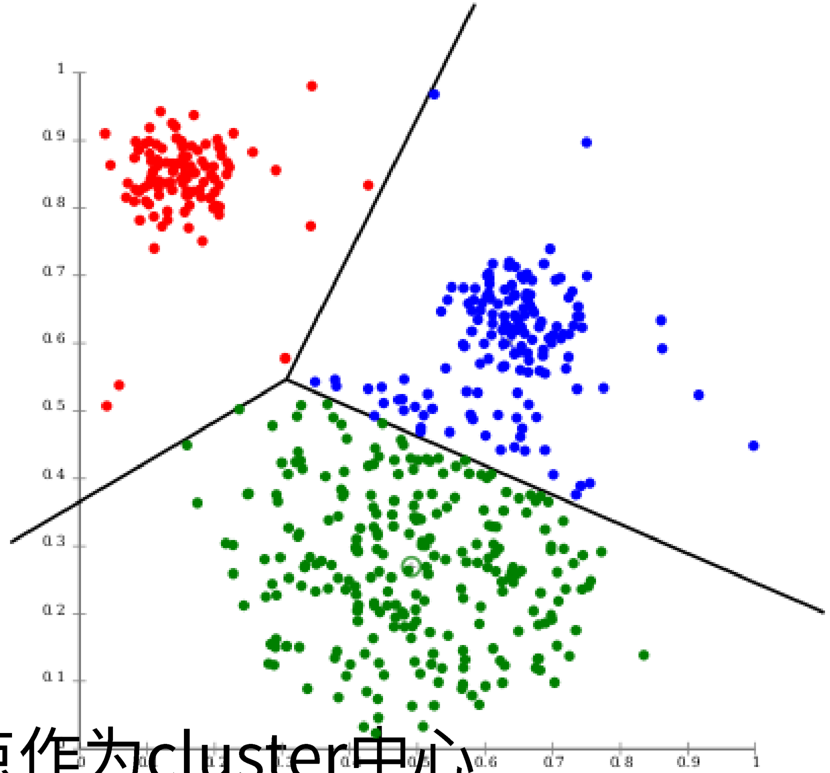
- 如何对距离数据与中心之间的距离进行度量?
  - 使用L2距离的平方
- 如何最优地选择中心点?
  - 对于d > 1, 全局最优是NP-hard问题
- 聚类结果如何?
  - 对于空间的Voronoi分割：将空间分割成多个多边形，每个多边形对应一个cluster中心 μ

2.1 K-means 算法

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \ldots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$



如何寻找一个局部最优解?

直观理解：

- 初始化：对每个cluster，任意选择空间中一个点作为cluster中心
- 迭代直至收敛：
  - ➢ 分配步骤：将每一个数据点分配至距离最近的中心
  - ➢ 重拟合步骤：根据新的分配重新计算聚类中心

2.1 K-means 算法

目标：将n个数据点分成k类

$$\min_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$

- 初始化：对每个cluster，任意选择空间中一个点作为cluster中心
- 迭代直至收敛：
  - 分配步骤：将每一个数据点分配至距离最近的中心
  - 重拟合步骤：根据新的分配重新计算聚类中心

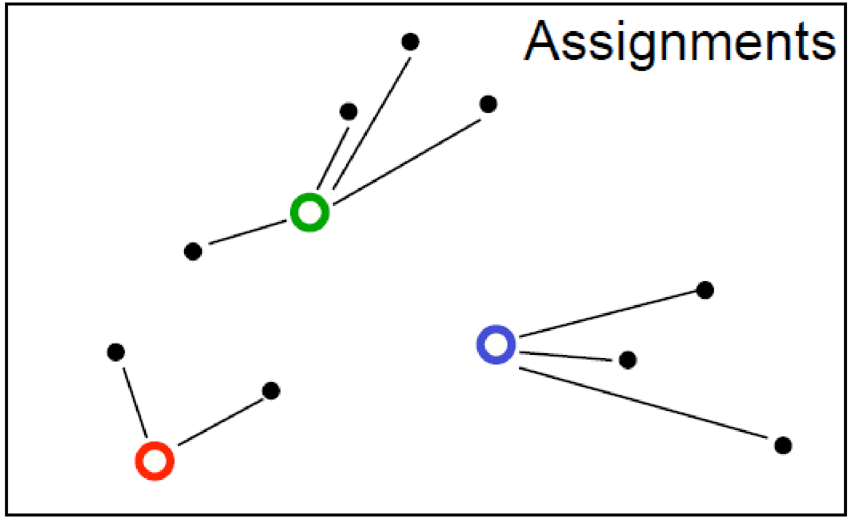
$$\min_{\mu} \sum_{i=1}^{t} ||y_i - \mu||^2 \quad \Longrightarrow \quad \mu^* = \frac{1}{t}\sum_{i=1}^{t} x_i$$

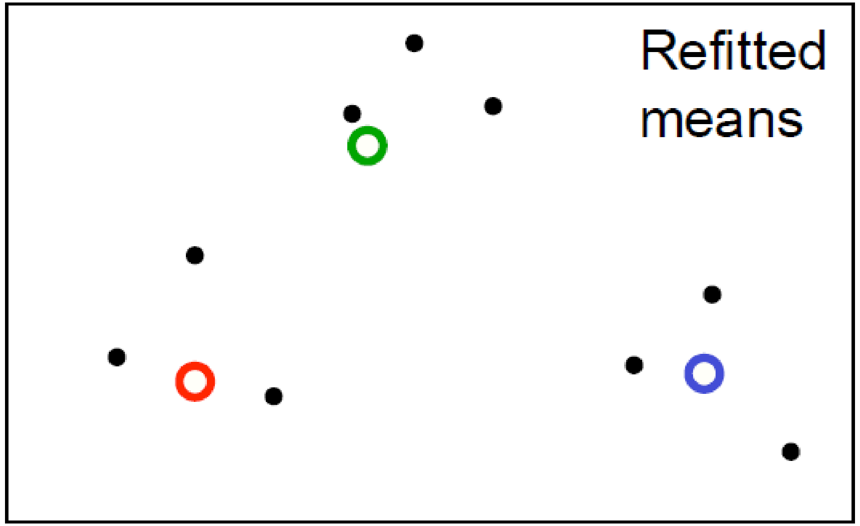
- 最优聚类中心：当前cluster中所有数据点的质心 (算数平均值)

K-means 算法

- 最优聚类中心：当前cluster中所有数据点的质心 (算数平均值)

$$\min_{\mu} \sum_{i=1}^{t} ||y_i - \mu||^2 \Longrightarrow \mu^* = \frac{1}{t}\sum_{i=1}^{t} x_i$$

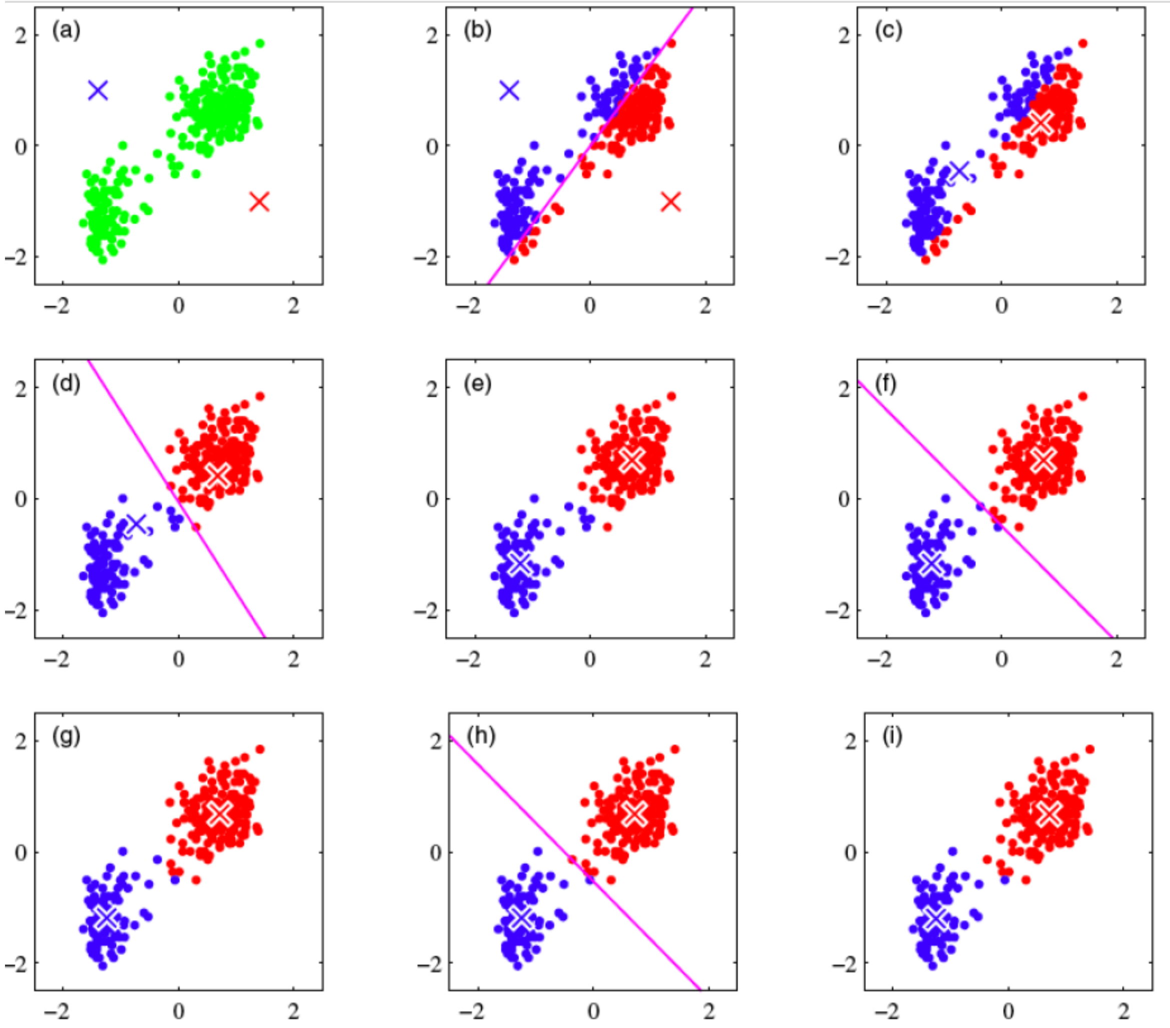




2.1 K-means 算法

例子：

- 迭代直至收敛：
  - ➢ 分配步骤：将每一个数据点分配至距离最近的中心
  - ➢ 重拟合步骤：根据新的分配重新计算聚类中心

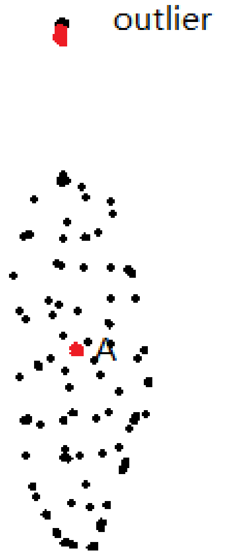


2.2 K-medoids 算法

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \ldots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$



K-means算法的缺点：

- 聚类中心 μ不一定属于数据集
- K-means 由于使用了L2距离函数，容易被outlier和noisy data影响

2.2 K-medoids 算法

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \dots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\mu_1,\dots,\mu_k \subseteq X} \sum_{i=1}^{n} \min_{j=1,\dots,k} ||x_i - \mu_j||_1$$

K-medoids算法：

- 限制聚类中心必须来自数据点
- 使用L1函数作为距离函数(L2->L1)
  - 相似的想法：Ridge Regression -> Lasso

K-medoids 算法

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \dots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\mu_1, \dots, \mu_k \subseteq X} \sum_{i=1}^{n} \min_{j=1,\dots,k} ||x_i - \mu_j||_1$$

- 初始化：对每个cluster，任意选择数据集中一个点作为cluster中心
- 迭代直至收敛：
  - ➢ 分配步骤：将每一个数据点分配至距离最近的中心(用L1距离函数)
  - ➢ 重拟合步骤：对于每一个cluster，选择离其他点最近的点作为新的中心

K-medoids 算法

$$\min_{\mu_1,\ldots,\mu_k \subseteq X} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||_1$$

- 初始化：对每个cluster，任意选择数据集中一个点作为cluster中心
- 迭代直至收敛：
  - ➢ 分配步骤：将每一个数据点分配至距离最近的中心(用L1距离函数)
  - ➢ 重拟合步骤：对于每一个cluster，选择离其他点最近的点作为新的中心

K-means 更新中心复杂度：O(n)

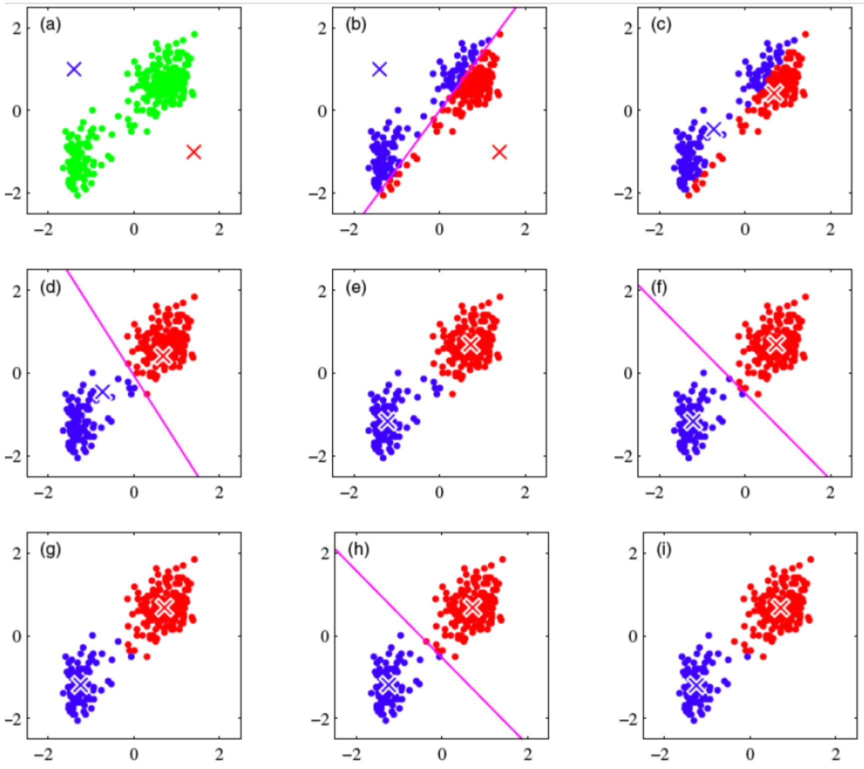
$$\mu^* = \frac{1}{t}\sum_{i=1}^{t} x_i$$

K-medoids 更新中心复杂度：O(n^2)

2.2 K-medoids 算法

	K-means	K-medoids
距离函数	L2 函数	L1 函数
聚类中心	Cluster中所有数据的质心	Cluster中最中心的数据点
Robust?	否	是
更新聚类中心复杂度	O(n)	O(n^2)
初始化敏感?	是	是

要点总结



2.1 K-means/K-medoids 算法目标函数

2.2 K-means/K-medoids 算法的区别以及联系

2.3 K-means的迭代求局部最优解算法

03 K-means的扩展：Soft K-means

3.1 K-means 新视角

3.2 高斯混合模型

K-means 新视角

目标：将n个数据点分成k类

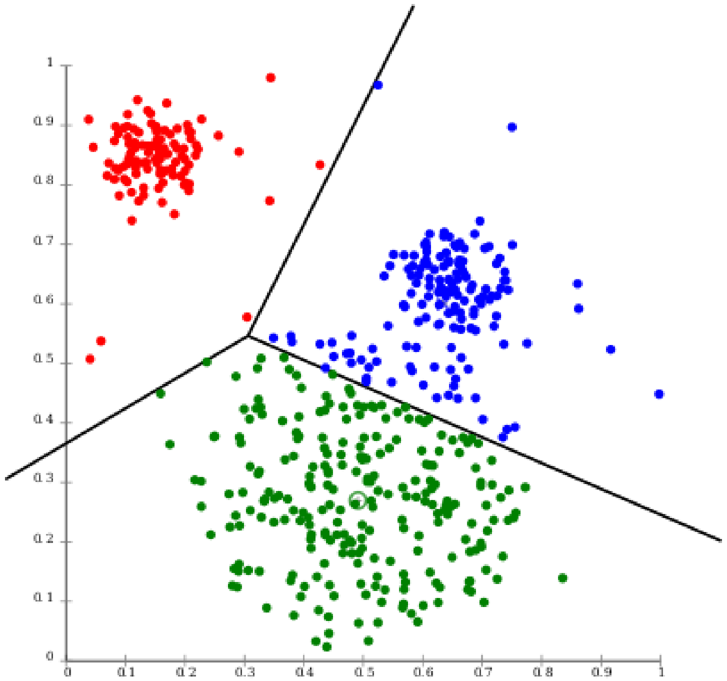
- 给定：$X = \{x_i\}, i = 1, \ldots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\mu_1,\ldots,\mu_k} \sum_{i=1}^{n} \min_{j=1,\ldots,k} ||x_i - \mu_j||^2$$

$$\Downarrow$$

$$\min_{\{\mu_j\},\{r_{ij}\}} \sum_{i=1}^{n} \sum_{j=1}^{k} r_{ij} ||x_i - \mu_j||^2$$

$$s.t. \quad \forall i \in [n], \sum_{j=1}^{k} r_{ij} = 1, r_{ij} \in \{0,1\}$$



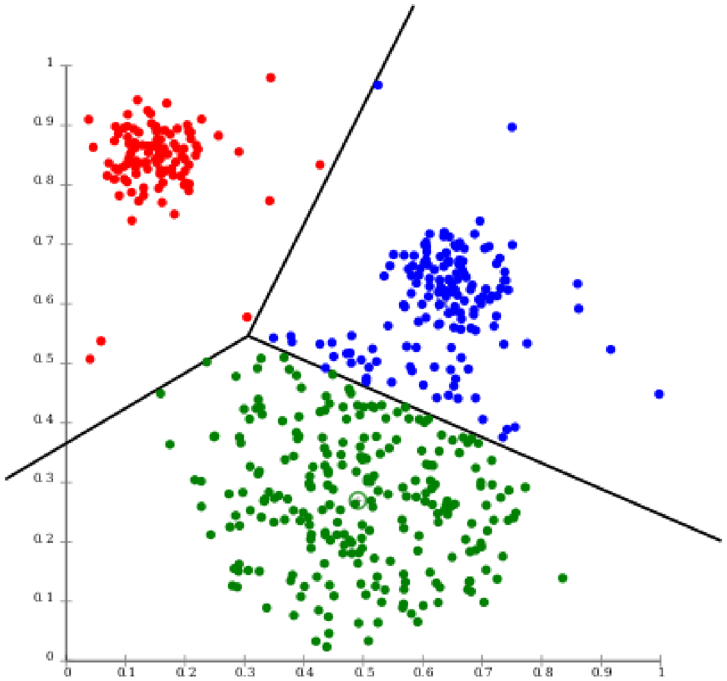
K-means 新视角

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \ldots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\{\mu_j\},\{r_{ij}\}} \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}||x_i - \mu_j||^2$$

$$s.t. \quad \forall i \in [n], \sum_{j=1}^{k} r_{ij} = 1, r_{ij} \in \{0, 1\}$$



思考：
为什么这两个形式等价?

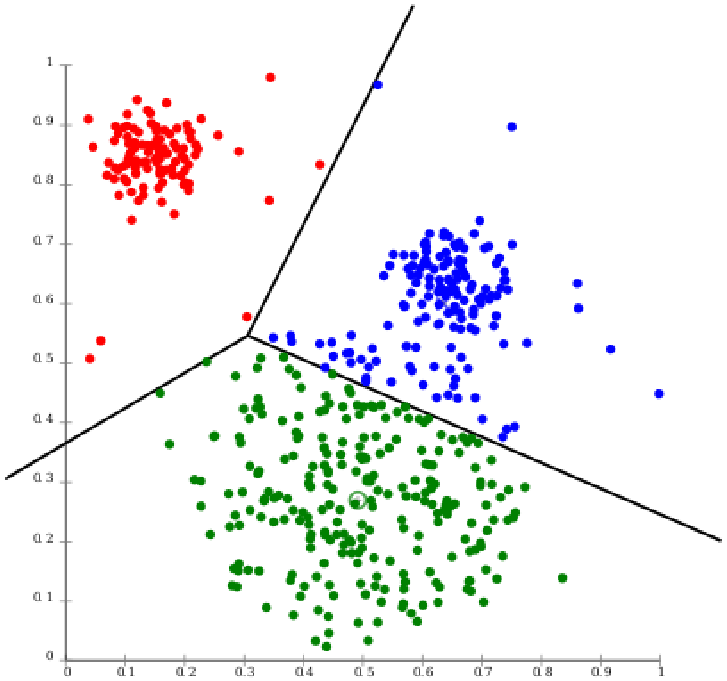
K-means 新视角

目标：将n个数据点分成k类

- 给定：$X = \{x_i\}, i = 1, \ldots, n \subseteq R^d$
- 给定：$k \in N$

$$\min_{\{\mu_j\},\{r_{ij}\}} \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}||x_i - \mu_j||^2$$

$$s.t. \quad \forall i \in [n], \sum_{j=1}^{k} r_{ij} = 1, r_{ij} \in \{0,1\}$$



推导：

K-means 新视角

K-means算法更新公式:

$$\min_{\{\mu_j\},\{r_{ij}\}} \quad \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}||x_i-\mu_j||^2$$

$$s.t. \quad \forall i\in[n], \sum_{j=1}^{k} r_{ij}=1, r_{ij}\in\{0,1\}$$

- 分配步骤：将每一个数据点分配至距离最近的中心

$$\forall i\in[n], r_{ij}=1 \quad \text{iff} \quad j=\operatorname{argmin}_{j'} \quad ||x_i-\mu_{j'}||$$

- 重拟合步骤：对于每一个cluster，选择离其他点最近的点作为新的中心

$$\mu_j=\frac{\sum_{i=1}^{n} r_{ij}x_i}{\sum_{i=1}^{n} r_{ij}}$$

K-means算法目标函数：

$$\min_{\{\mu_j\},\{r_{ij}\}} \quad \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}||x_i-\mu_j||^2$$

$$s.t. \quad \forall i \in [n], \sum_{j=1}^{k} r_{ij}=1, r_{ij}\in\{0,1\}$$

- Hard assignment: 每个点只能属于一个cluster

$$\forall i \in [n], r_{ij}=1 \quad \text{iff} \quad j=\operatorname{argmin}_{j'} \quad ||x_i-\mu_{j'}||$$

- Soft assignment: 每个点以一个概率属于任意一个cluster：

$$\forall i \in [n], 0 \le r_{ij} \le 1, \sum_{j=1}^{k} r_{ij}=1$$

Gaussian Mixture Model for Clustering (a. k. a. Soft K-means Clustering)

- Soft assignment: 每个点以一个概率属于任意一个cluster：

$$\forall i \in [n], 0 \le r_{ij} \le 1, \sum_{j=1}^{k} r_{ij} = 1$$

- $r_{ij} = x_i$ 属于第 j 个cluster的概率

Gaussian Mixture Model for Clustering (a. k. a. Soft K-means Clustering)

- 分配步骤：将每一个数据点分配以不同的概率分配到不同的cluster

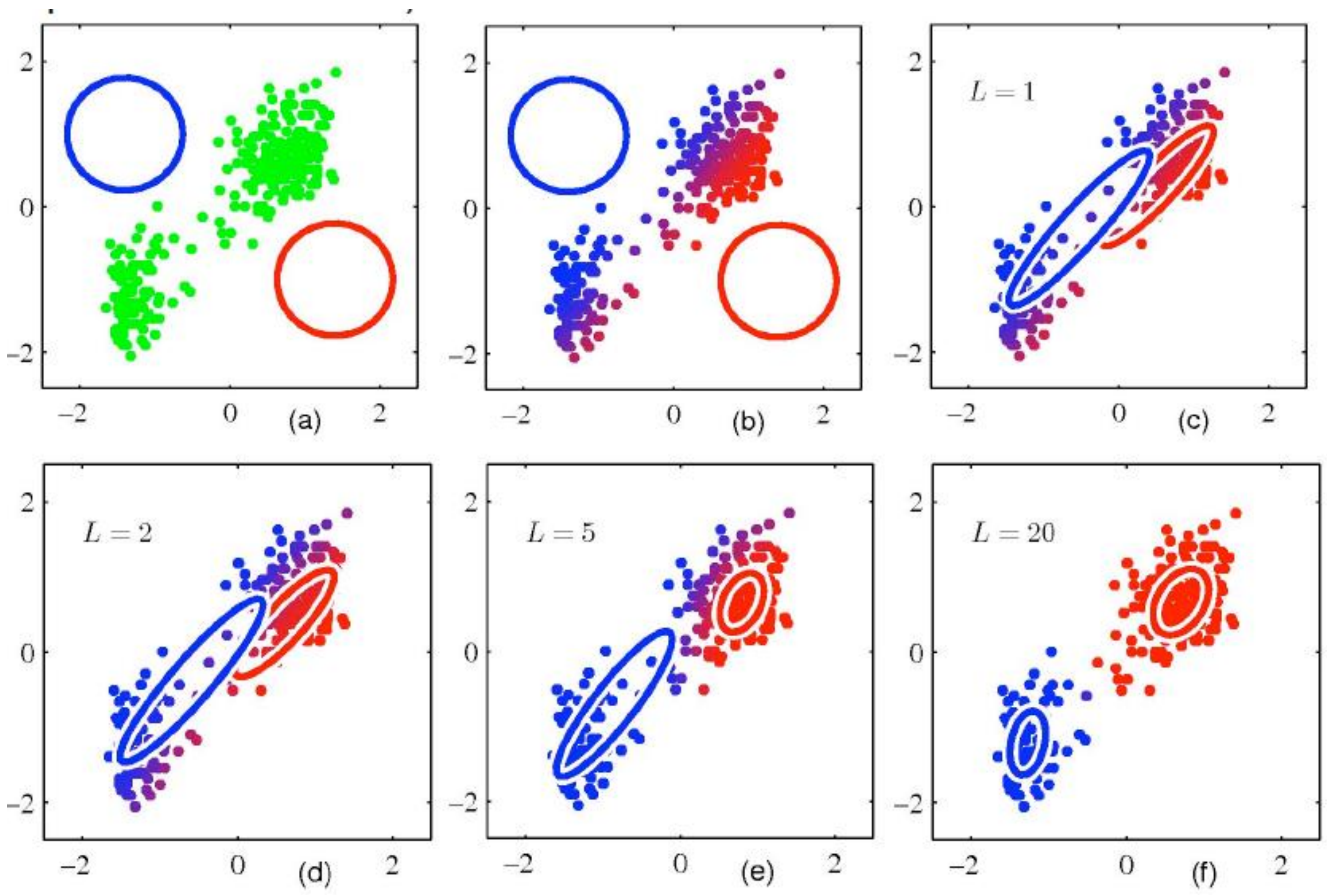
$$r_{ij} = \frac{\exp(-\beta||x_i - \mu_j||^2)}{\sum_{j'=1}^{k} \exp(-\beta||x_i - \mu_{j'}||^2)}$$

- 重拟合步骤：对于每一个cluster，选择离其他点最近的点作为新的中心

$$\mu_j = \frac{\sum_{i=1}^{n} r_{ij} x_i}{\sum_{i=1}^{n} r_{ij}}$$

加权版的K-means！中心更新公式仍然为质心公式，只不过现在是加权质心！

Soft K-means vs. (Hard) K-means

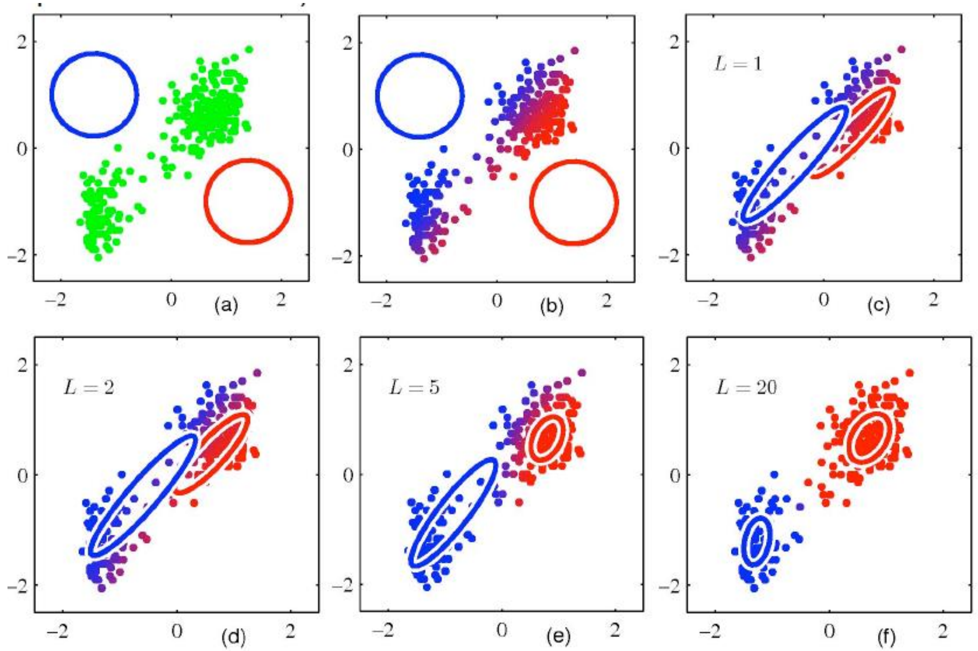


Soft K-means vs. (Hard) K-means

- 当$\beta \rightarrow \infty$时，Soft K-means 退化成 (Hard) K-means
- Soft K-means 的目标函数等价于最大化混合高斯模型的似然函数
- 实际应用中Soft K-means 收敛速度比 (Hard) K-means 慢
- Soft K-means 应用往往更广，概率化的cluster assignment 理解为分配置信度

03 K-means 的扩展：Soft K-means

要点总结



3.1 K-means的基于置信度的推导

3.2 Soft K-means 与 (Hard) K-means的区别以及联系

04

层次聚类

4.1 Single-Linkage 算法

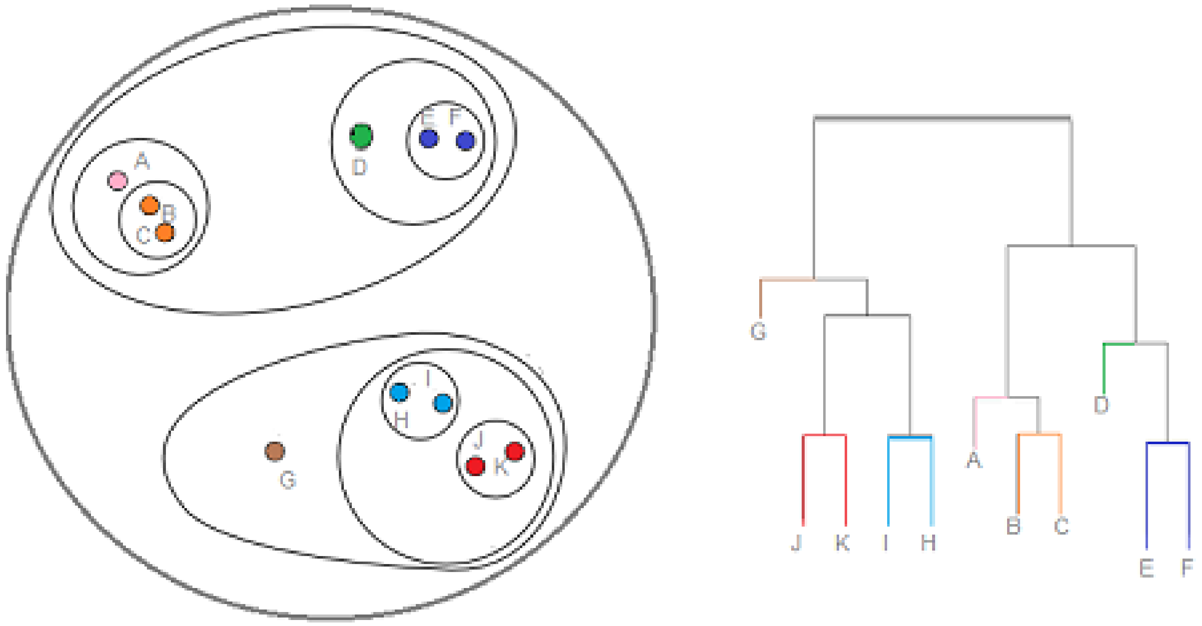
4.2 Complete-Linkage 算法

层次聚类 (Hierarchical Clustering) vs. 扁平聚类 (Partitional Clustering)

- K-means 以及 Soft K-means 都属于扁平化的聚类：
  - 类别与类别之间属于同一层次，没有嵌套关系
  - 一个数据只能属于一个类别
- 层次聚类:
  - 类别与类别之间有包含/嵌套关系
  - 一个数据可以属于多个类别

层次聚类

Single-Linkage 算法：

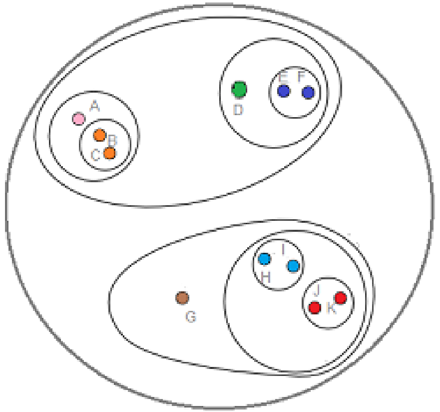


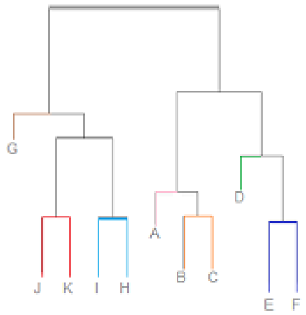
- 算法构造一棵二叉树
- 二叉树的叶节点代表数据
- 二叉树的每一个内部节点代表一个cluster

4 层次聚类

Single-Linkage 算法：

- 初始化：算法将每个数据看成一个类别
- 迭代直至只有一个类：
  - 选择距离最近的两个类进行合并
  - 将被合并的两个类从现有类中删除
  - 将合并后得到的新类加入现有类中

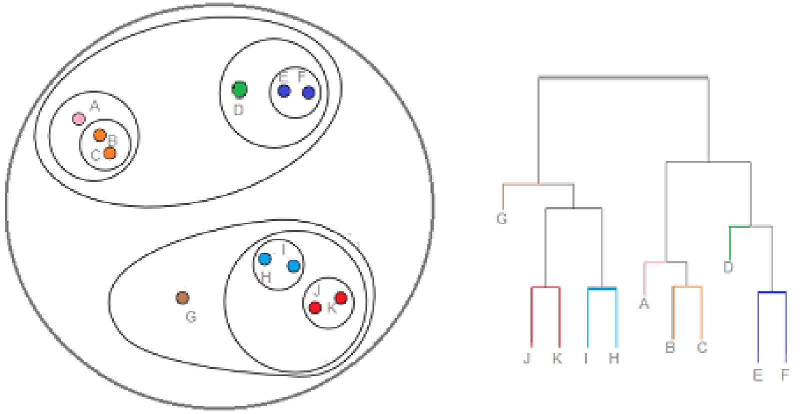




层次聚类

Single-Linkage 算法：

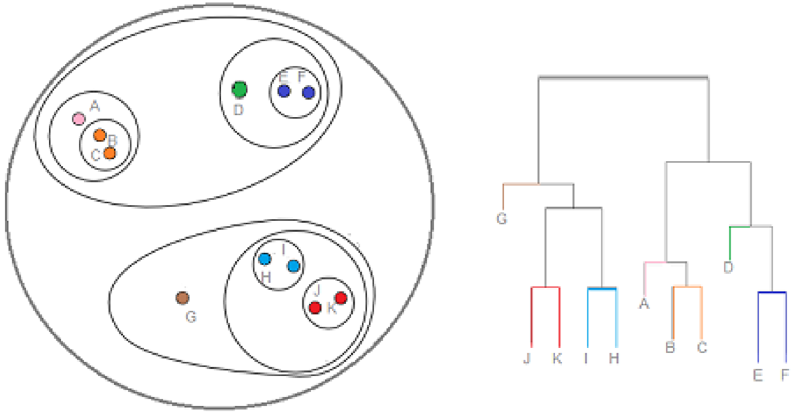
- 初始化：算法将每个数据看成一个类别
- 迭代直至只有一个类：
  - 选择距离最近的两个类进行合并
  - 将被合并的两个类从现有类中删除
  - 将合并后得到的新类加入现有类中



问题：假设一共有n个数据点，需要进行多少次合并操作?

层次聚类

Single-Linkage 算法：



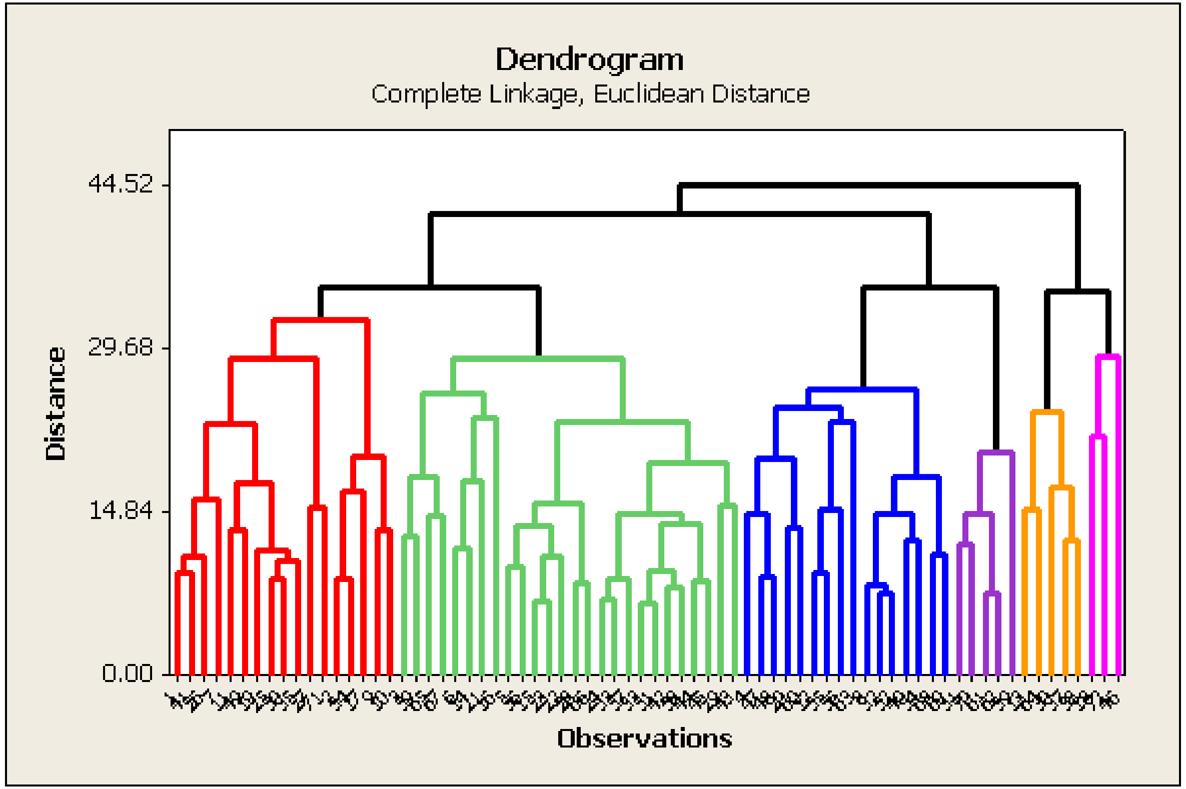
- 初始化：算法将每个数据看成一个类别
- 迭代直至只有一个类：
  - 选择距离最近的两个类进行合并
  - 将被合并的两个类从现有类中删除
  - 将合并后得到的新类加入现有类中

Single-Linkage: $d(S_i, S_j) = \min_{x_i \in S_i, x_j \in S_j} ||x_i - x_j||$

Complete-Linkage: $d(S_i, S_j) = \max_{x_i \in S_i, x_j \in S_j} ||x_i - x_j||$

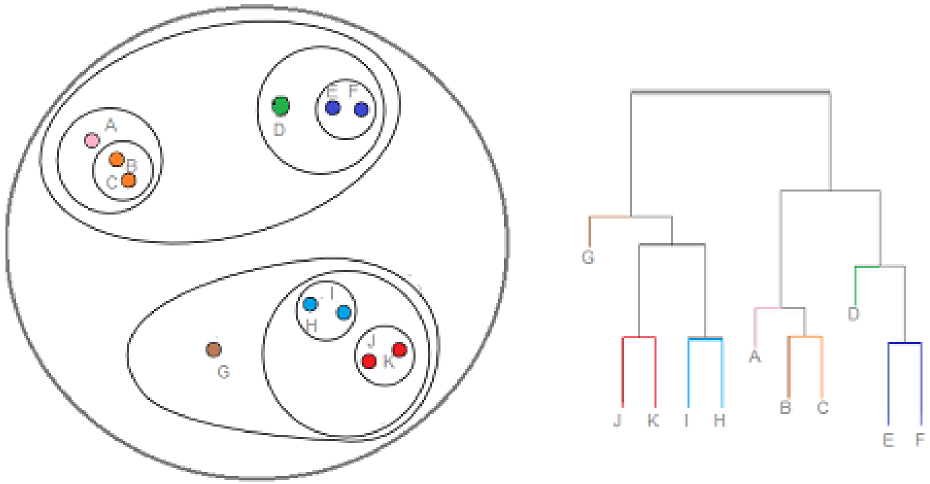
O(n^3) 复杂度，可以用优先队列的数据结构对算法加速：O(n^2 log n)

如何从层次聚类中获得扁平聚类?



在二叉树的某一层切割即可!

要点总结



4.1 层次聚类与扁平聚类的区别

4.2 Single-Linkage/Complete Linkage 算法

4.3 如何从层次聚类获得扁平聚类

THANK YOU！

Machine Learning Engineer
机器学习工程师微专业

稀牛学院 網易 NETEASE