

机器学习

Machine Learning

北京航空航天大学计算机学院

School of Computer Science and Engineering, Beihang University

黄迪 刘庆杰 陈佳鑫

2025年秋季学期

Fall 2025

聚类

Clustering

什么是聚类？

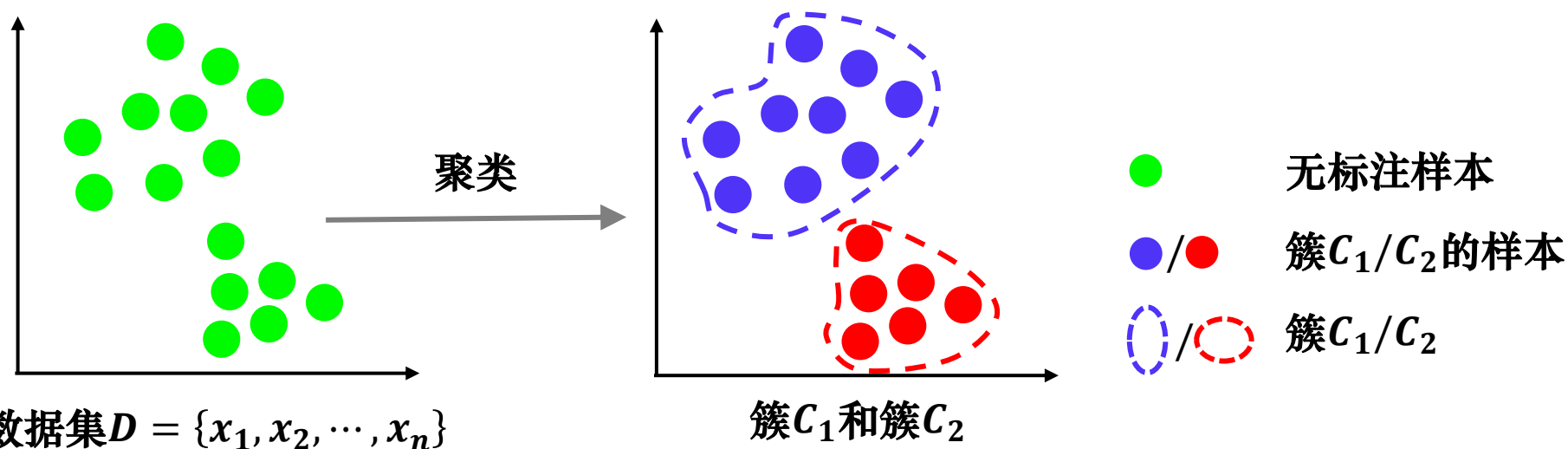
- 聚类的定义
- 聚类的性能度量
- 聚类的应用

什么是聚类?

● 聚类的定义

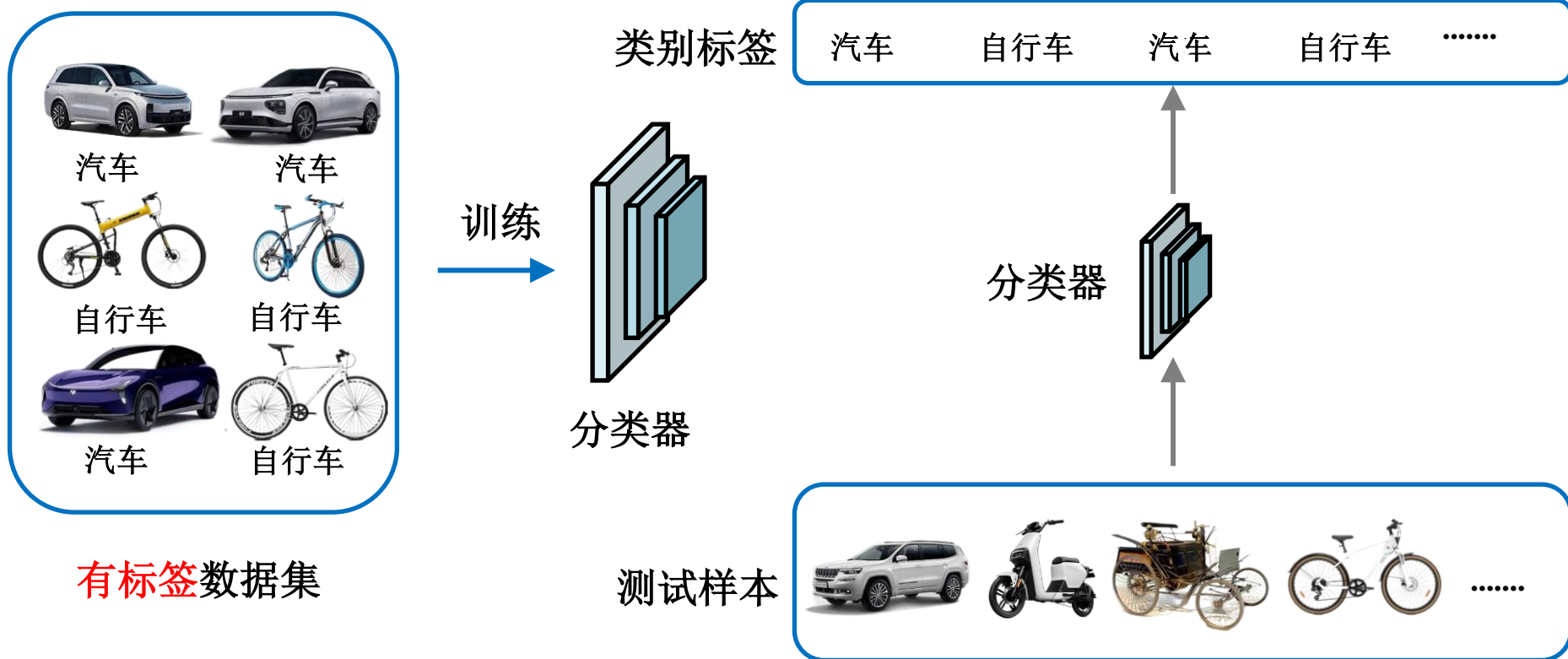
➤ 一种无监督学习任务

- 按照某个特定标准 (如距离), 将数据集中的无标注样本划分为若干个不相交的子集, 每个子集称为一个“簇” (Cluster)。
- 希望簇内样本的相似性尽可能大, 簇间样本的差异性尽可能大。



什么是聚类？

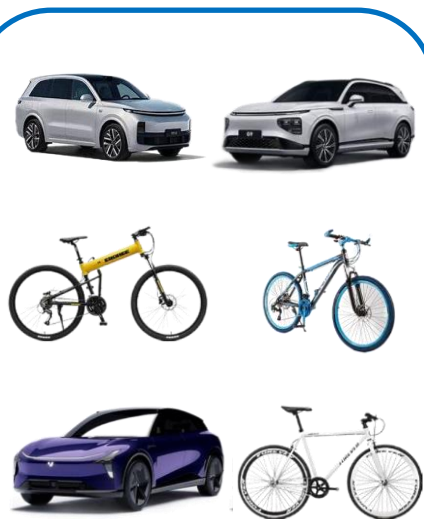
● 分类（监督学习）



当数据集**无标签**时，如何划分类别？

什么是聚类?

● 聚类 (无监督学习)



无标签数据集

聚类



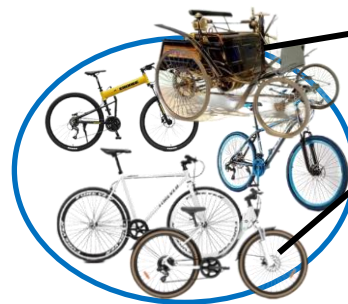
簇 C_1 (汽车)



簇 C_2 (自行车)



簇 C_1 (汽车)

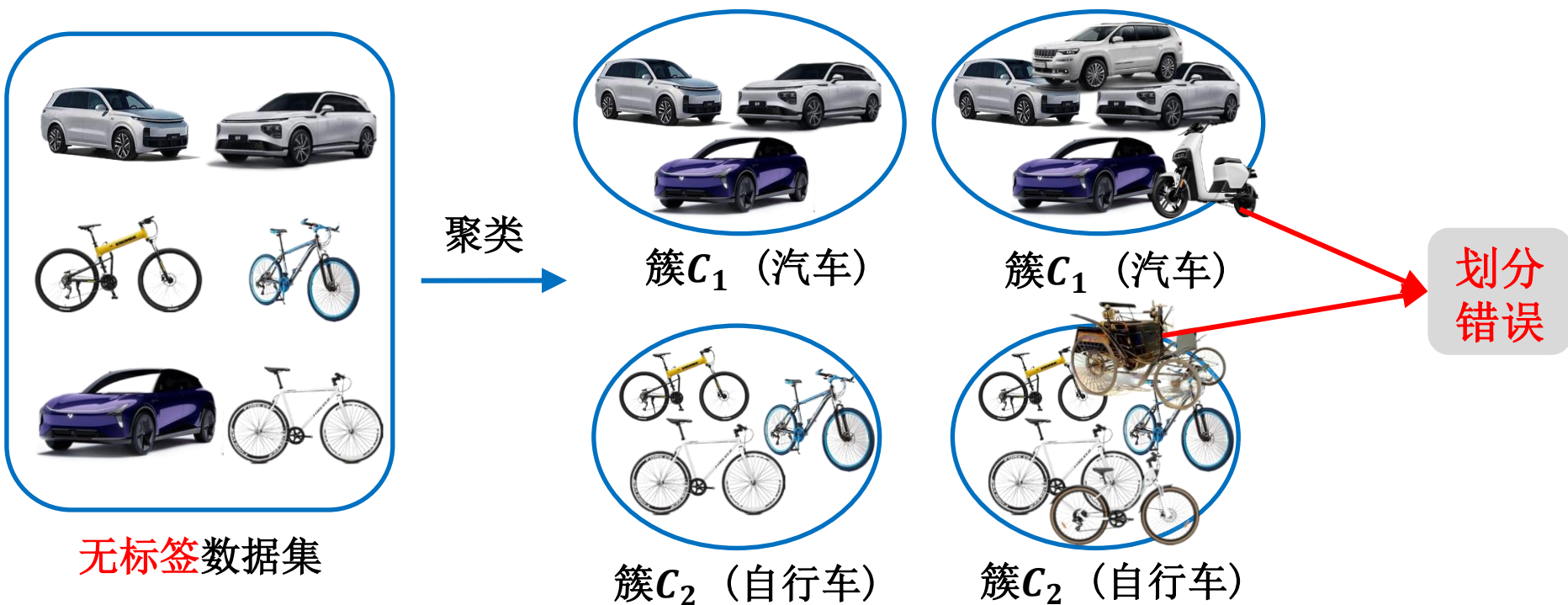


簇 C_2 (自行车)

测试
样本

什么是聚类？

● 聚类（无监督学习）



➤ 聚类通常比分类的**准确性低**

➤ 但是聚类更加**灵活**，具有处理**无标签**数据的能力；可以用于无标签数据处理中，辅助**分析**和**决策**

聚类形式化描述

- 假定样本集 $D = \{x_1, x_2, \dots, x_m\}$ 包含 m 个无标记样本，每个样本 $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ 是一个 n 维的特征向量，聚类算法将样本集 D 划分成 k 个不相交的簇 $\{C_l | l = 1, 2, \dots, k\}$ 。其中 $C_{l'} \cap C_l = \emptyset, l' \neq l$ ，且 $D = \bigcup_{l=1}^k C_l$ 。
- 用 $\lambda_j \in \{1, 2, \dots, k\}$ 表示样本 x_j 的“簇标记” (Cluster Label)，即 $x_j \in C_{\lambda_j}$ 。于是，聚类的结果可用包含 m 个元素的簇标记向量 $\lambda = \{\lambda_1; \lambda_2; \dots; \lambda_m\}$ 表示。

聚类的性能度量

- 聚类“有效性指标” (Validity Index)

同一簇样本相似度高，不同簇样本差异度大。即“簇内相似度” (Intra-cluster Similarity) 高，“簇间相似度” (Inter-cluster Similarity) 低。

- 外部指标 (External Index)

将聚类结果与某个“参考模型”进行比较

- 内部指标 (Internal Index)

直接考察聚类结果而不用任何参考模型

聚类的性能度量

● 聚类“有效性指标” (Validity Index)

对数据集 $D = \{x_1, x_2, \dots, x_m\}$ ，假定通过聚类得到的簇划分为 $C = \{C_1, C_2, \dots, C_k\}$ ，参考模型的簇划分为 $C^* = \{C_1^*, C_2^*, \dots, C_s^*\}$ 。相应地，令 λ 与 λ^* 分别表示与 C 和 C^* 对应的簇标记向量。

定义

$$a = |SS|, SS = \{(x_i, x_j) | \lambda_i = \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}$$

$$b = |SD|, SD = \{(x_i, x_j) | \lambda_i = \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}$$

$$c = |DS|, DS = \{(x_i, x_j) | \lambda_i \neq \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}$$

$$d = |DD|, DD = \{(x_i, x_j) | \lambda_i \neq \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}$$

$$a + b + c + d = m(m - 1)/2$$

聚类的性能度量

● 聚类“有效性指标” (Validity Index)

外部指标:

➤ Jaccard系数 (Jaccard Coefficient, JC)

$$JC = \frac{a}{a+b+c}$$

➤ FM指数 (Fowlkes and Mallows Index, FMI)

$$FMI = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}$$

➤ Rand指数 (Rand Index, RI)

$$RI = \frac{2(a+d)}{m(m-1)}$$

[0,1]区间内
越大越好

聚类的性能度量

● 聚类“有效性指标” (Validity Index)

内部指标:

➤ DB指数 (Davies-Bouldin Index, DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left(\frac{avg(C_i) + avg(C_j)}{d_{cen}(\mu_i, \mu_j)} \right)$$

越小越好

➤ Dunn指数 (Dunn Index, DI)

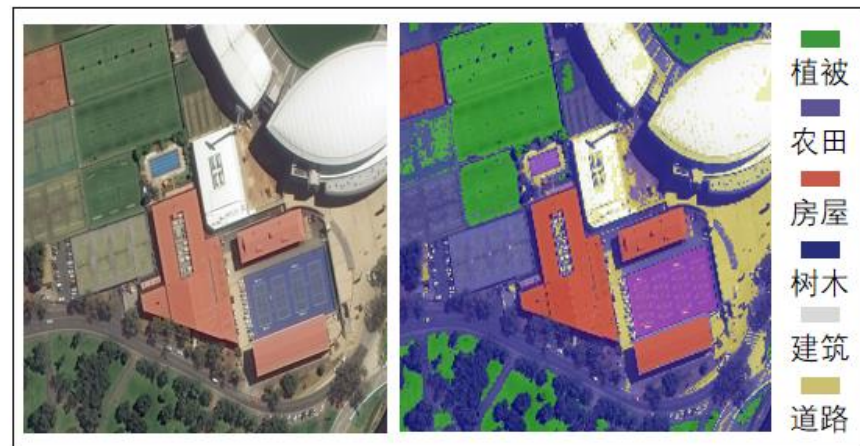
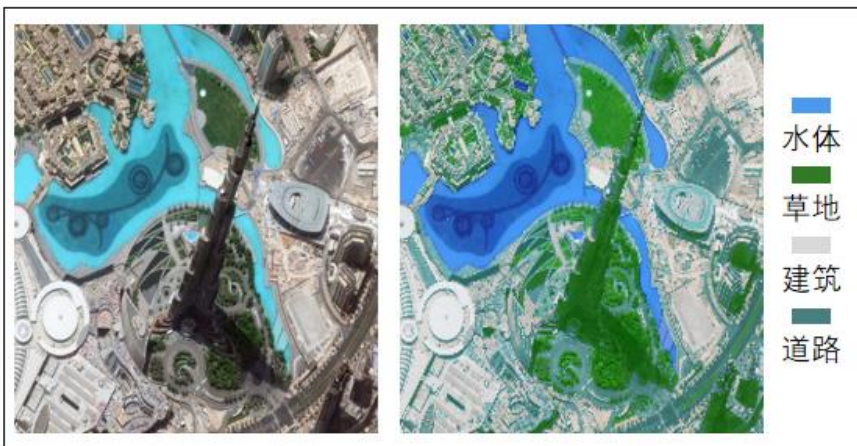
$$DI = \min_{1 \leq i \leq k} \left\{ \min_{j \neq i} \left(\frac{d_{min}(C_i, C_j)}{\max_{1 \leq l \leq k} diam(C_l)} \right) \right\}$$

越大越好

聚类的应用

● 遥感图像分割

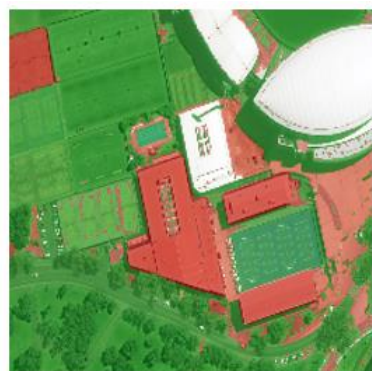
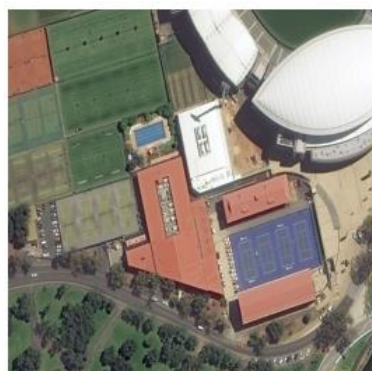
- 遥感图像分割是将遥感图像中的像素划分为**具有相似特征的区域**的过程。传统的图像分割通常基于聚类的思想实现。
- 应用于资源管理、环境监测、灾害评估、土地利用规划等。



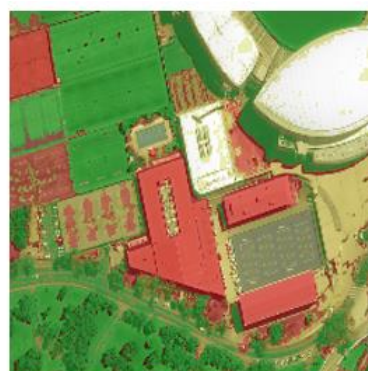
聚类的应用

● 遥感图像分割

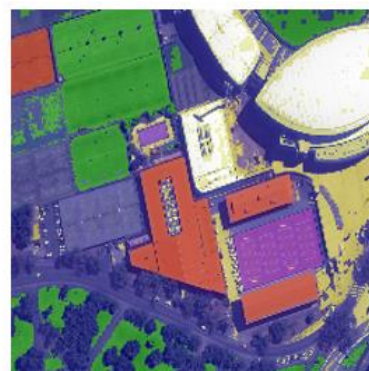
聚类中心增多



K = 3



K = 4



K = 6

植被
农田
房屋
树木
建筑
道路

➤ 将每个像素点视为一个无标注样本，进行聚类。每个样本（像素点）是维度为3的颜色向量。

➤ 除了利用颜色信息，也可以使用经过神经网络提取的特征向量作为样本，充分利用语义信息进行聚类。

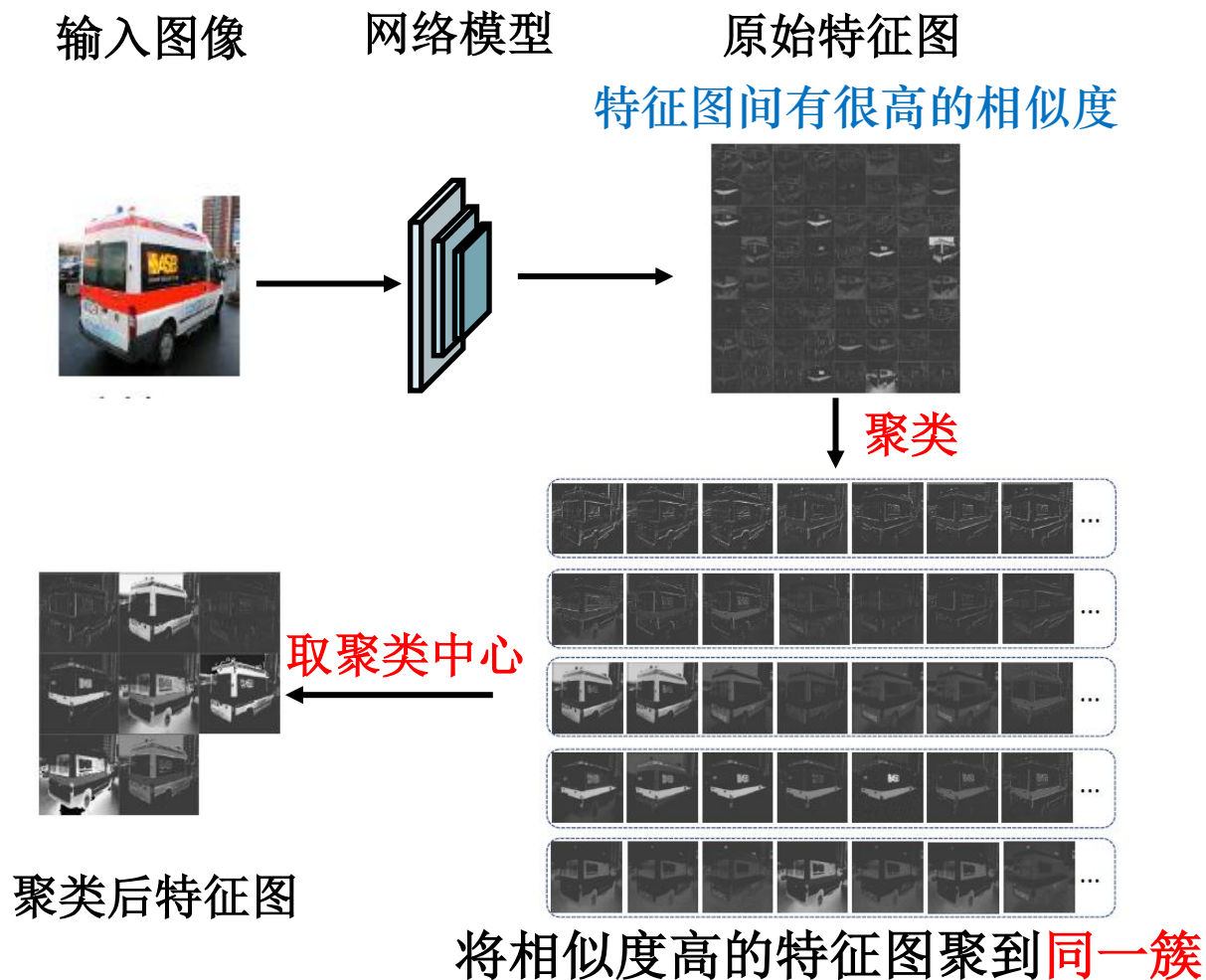
聚类的应用

● 神经网络压缩

- 深度神经网络需要充足的存储资源和计算资源。例如，ResNet-50在处理一张图像时，需要超过95MB的存储空间和38亿次浮点运算。
- 但是，神经网络模型在实际部署中往往受到计算资源和存储资源的**限制**。例如，一些物联网（IoT）设备通常只有**几MB**的RAM。
- 为了解决这个问题，一种常用的方法是对神经网络进行**压缩**。传统的神经网络压缩算法通过对特征图进行聚类，然后应用滤波器剪枝，来实现减少神经网络的冗余。例如，ResNet-50可以通过模型压缩减少到5MB甚至更小。

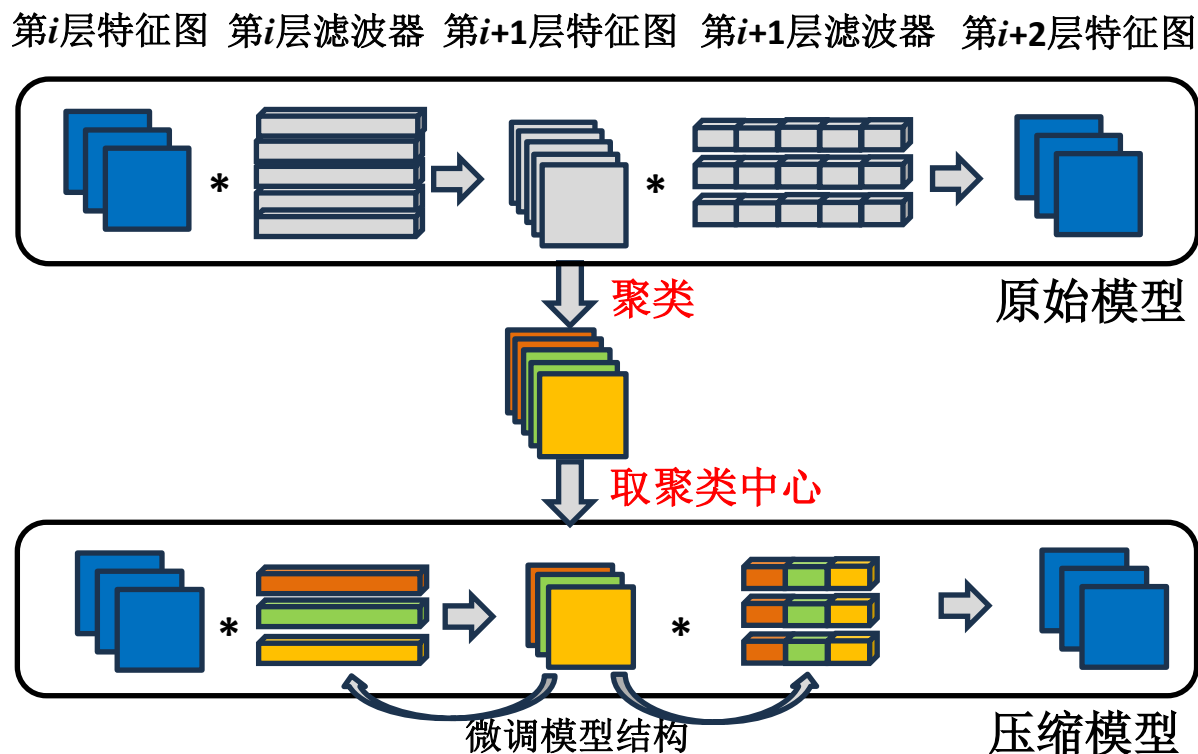
聚类的应用

● 神经网络压缩



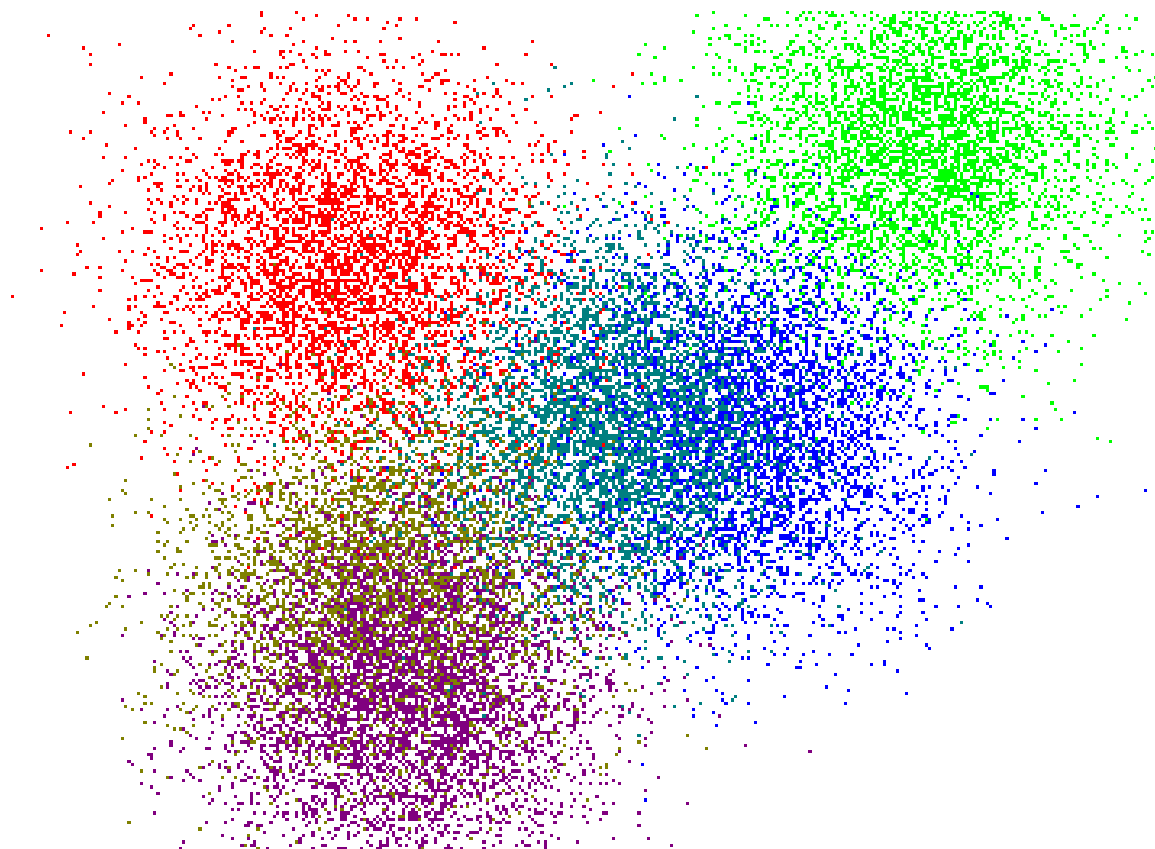
聚类的应用

● 神经网络压缩



聚类方法

- K-Means 聚类
- 高斯混合聚类
- 层次聚类
- 密度聚类
- 谱聚类



K均值算法

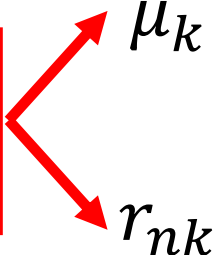
- K均值算法的基本概念
- K均值算法的基本思想
- K均值算法的具体求解

K均值算法的基本概念

- K均值算法 (K-Means) ——最常用的聚类算法

➤ 定义：给定D维空间上的数据集 $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ，这些数据对应类别未知。K均值算法将数据集划分成K类，各类的聚类中心记为 μ_1, \dots, μ_k ，并将每一个样本 \mathbf{x}_n 划归到离该样本最近的聚类中心

➤ 准则函数：各数据点到对应聚类中心距离之和

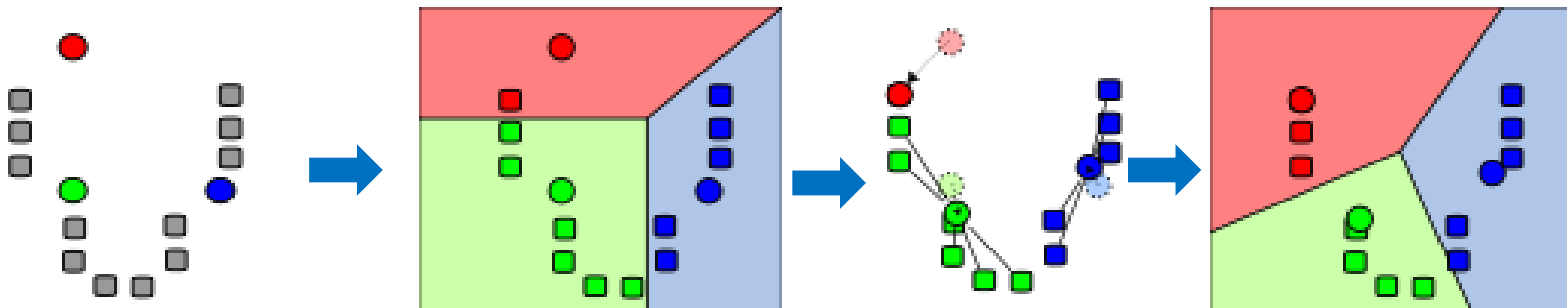
$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \mu_k\|^2$$


当 \mathbf{x}_n 属于第 k 个聚类时， $r_{nk} = 1$
记 r_n 为中间变量，可以看作是“隐变量”

K均值算法的具体求解

● K均值算法的一般流程

- **步骤1.** 初始化选择K个初始聚类中心
- **步骤2.** 将每个数据点划分给最近的聚类中心 μ_k ，得到**聚类标注** r_n
- **步骤3.** 最小化准则函数，重新计算**聚类中心** μ_k
- **步骤4.** 迭代步骤2和3，直到满足终止条件
 - 聚类中心不再发生显著变化或达到最大迭代次数



K均值算法的具体求解

● 迭代优化策略

➤ 步骤2：对于给定的 μ_k ，按照最优化准则产生 r_{nk}

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \mu_k\|^2$$

划分

➔

$$r_{nk} = \begin{cases} 1, & \text{若 } k = \arg \min_J \|\mathbf{x}_n - \mu_k\|^2 \\ 0, & \text{其他} \end{cases}$$

K均值算法的具体求解

● 迭代优化策略

➤ 步骤3：对于给定的 r_{nk} ，按照最优化准则产生 μ_k

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \mu_k\|^2$$

对 μ_k 求导



$$2 \sum_{n=1}^N r_{nk} (\mathbf{x}_n - \mu_k) = 0$$

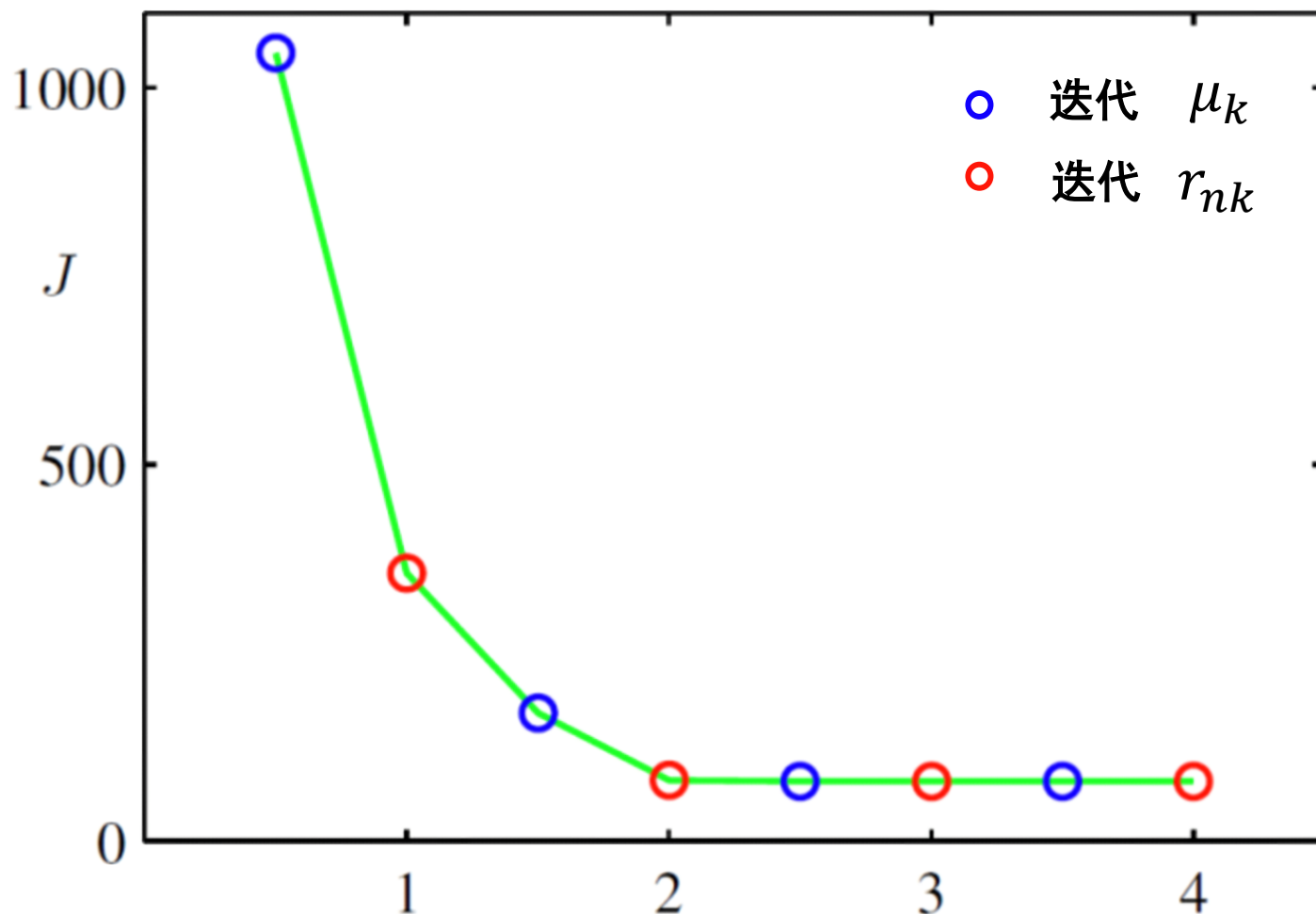
求解得



$$\mu_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

K均值算法的具体求解

● 准则函数变化趋势



K均值算法过程可视化

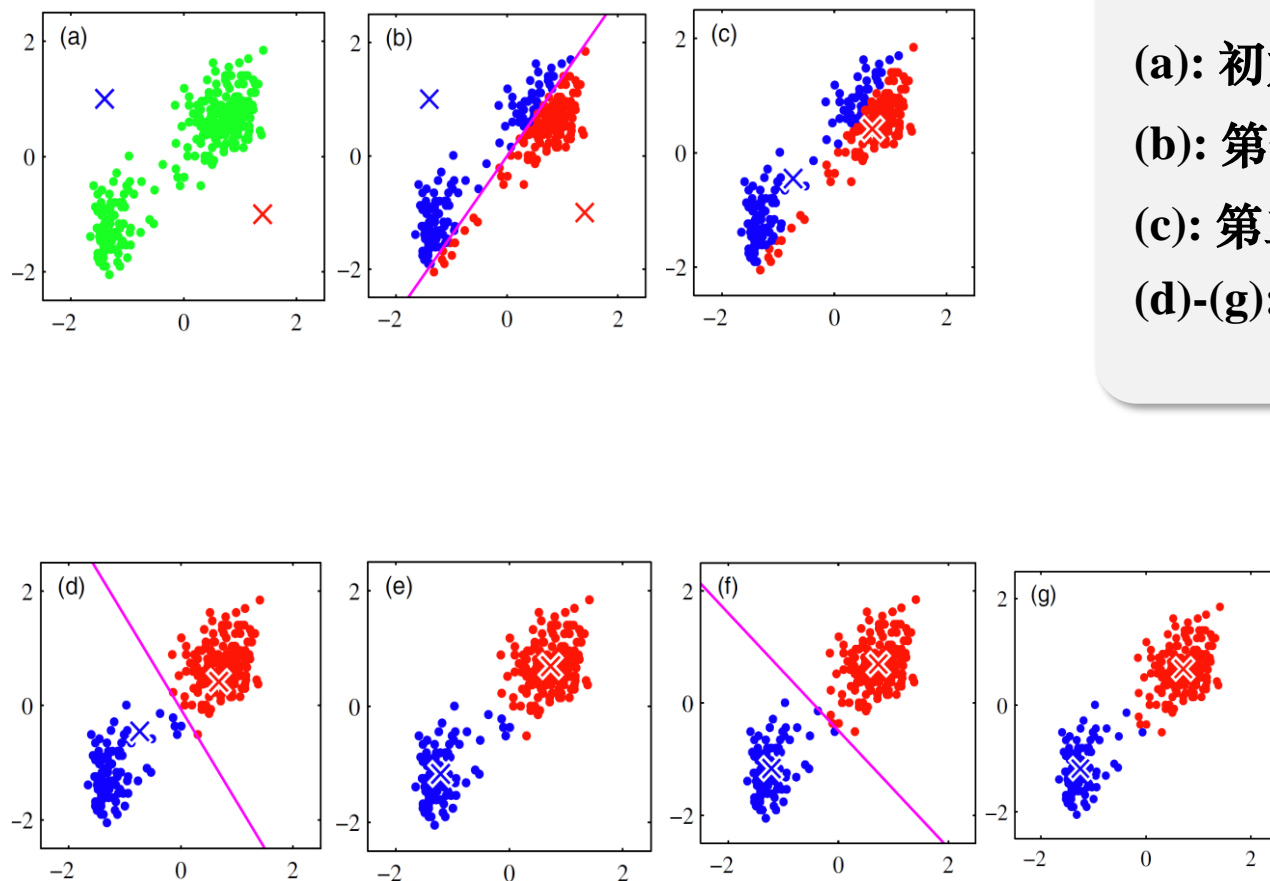
K均值算法流程

(a): 初始化各类代表点 μ_k

(b): 第一步更新 r_{nk}

(c): 第二步计算 μ_k

(d)-(g): 迭代两个步骤直至停止条件



K均值算法的优缺点

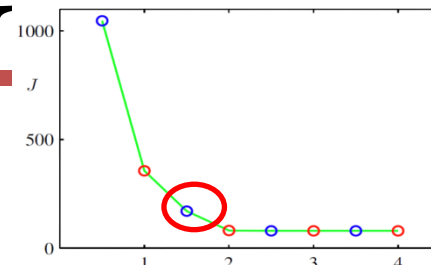
● 优点：

- 简单易懂、易于理解和实现
- 计算成本低、一般可以快速收敛
- 适应性强，各类大小大致相等且形状为球形时效果较好

● 缺点：

- 需要预先指定聚类中心K的数量，对**初始化**中心点敏感
- 欧几里德距离假设所有变量在距离计算中同等重要，**限制了能处理的数据变量**
- 不适用于**非线性**数据集

K均值算法的改进



● 选择初始化聚类数量K

- 肘部法则：尝试不同的进行实验，收敛后的准则函数可能会呈现一条类似于人的肘部的曲线。

● 迭代自组织数据分析算法 ISODATA

- 改进：增加对聚类结果的“合并”和“分裂”操作，设定算法运行控制参数。
- 基本思想：引入人的决策，1). 两类聚类中心距离小于某阈值时进行合并；2). 当某类标准差大于设定阈值或其样本数目超过设定阈值时进行分裂；3). 在某类样本数目少于设定阈值时，重新划分其到其他类别。

K均值算法的改进

● K-means++

- 改进： 2007年由Arthur和Vassilvitskii提出的K-means++针对K-means的聚类中心初始化做了改进。
- 基本思想： 假设已选取了 n 个初始聚类中心($0 < n < k$)，在选取第 $n+1$ 个中心时： 距离当前 n 个聚类中心越远的点会有更高的概率被选为第 $n+1$ 个聚类中心，记候选中心 x 被选为聚类中心的概率为 $P(x)$. 根据 $P(x)$ 用轮盘法选出第 $n+1$ 个聚类中心。

$$P(x) = \frac{D(x)^2}{\sum_{x \in X} D(x)^2}$$

$D(x)$ 表示候选中心 x 到已有中心的最远距离

K均值算法的改进

● Kernel K-means

- 改进：传统K-means采用欧式距离进行样本间的相似度量，显然并不是所有的数据集都适用于这种度量方式。
- 基本思想：参照支持向量机中核函数的思想，将所有样本**映射到另外一个特征空间(一般为高维)中**，希望在这个特征空间中数据可以变得更容易分离或更好的结构化，并在新的特征空间中进行聚类。

高斯混合模型

- 认识高斯混合模型
- 高斯混合模型的极大似然估计
- 高斯混合模型的EM求解
- 高斯混合模型的应用

高斯混合模型

(Gaussian Mixture Model)

- 什么是高斯混合模型 (Gaussian Mixture Model, GMM)
 - 是一种统计模型，用于表示一组数据是由多个高斯分布混合而成的。具体地，高斯混合模型是多个高斯分布的线性组合，每个高斯分布称为一个**混合成分**，每个混合成分都有一个对应的**混合系数**，所有**混合系数的和为1**

$$p(x) = \sum_i \alpha_i \mathcal{N}(x|\mu_i, \Sigma_i), \sum_i \alpha_i = 1$$

混合系数

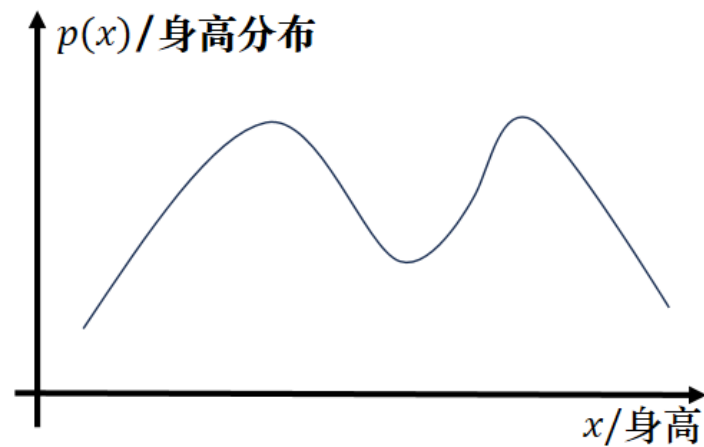
混合成分

- GMM能够捕捉数据的**多峰特性**，即数据集中可能存在多个簇，每个簇的分布可以用一个高斯分布来描述
- GMM广泛应用于聚类分析、图像分割、语音识别、数据降维等领域

高斯混合模型

● 例子

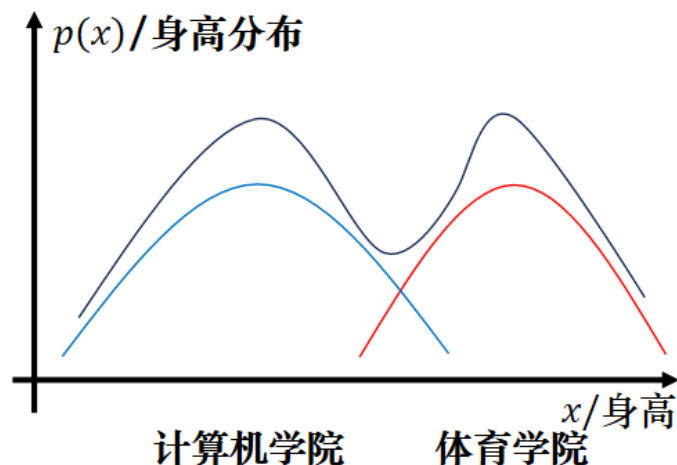
- 统计学校各院系同学的身高数据。
从学校的档案馆中随机抽取了N
(如N=2000) 名同学的身高信息，
但并不知道学生属于哪个院系。假设每个学院同学的身高服从高斯分布
- 问：如何从中得到每个学生的院系划分（聚类簇）及各院系的学生身高分布？



高斯混合模型

● 例子

- 统计学校各院系同学的身高数据。从学校的档案馆中随机抽取了 N （如 $N=2000$ ）名同学的身高信息，但并不知道学生属于哪个院系。假设每个学院同学的身高服从高斯分布
- 问：如何从中得到每个学生的院系划分（聚类簇）及各院系的学生身高分布？



高斯混合模型

● 符号定义

- **观测值**: 每个学生的身高 $X = \{x_1, x_2, \dots, x_N\}$
- **分布参数**: 第 k 个院系的学生身高服从高斯分布 $\mathcal{N}(\mu_k, \sigma_k^2)$
- **求解目标**: 总体身高分布 $p(x)$

● 引入中间变量建模GMM

- **K-Means**使用 r_n 表示每个样本 x_n 所属的聚类簇。GMM引入
隐变量 $Z = \{z_1, z_2, \dots, z_N\}$ 表示每个学生所属的院系

高斯混合模型

● 隐变量 z 的含义

- z 在实际观测中**不可知**。假设共有 K 个不同的院系，可以用一个 K 维**独热向量** (one-hot vector)来表示。 z 只在第 k 维为**1**，其他维均为**0**，代表学生 x 属于第 k 个院系。
- GMM通过引入隐变量 z 来表示**样本 x 来自第 k 个高斯分布**，用于简化问题的求解。

$$z = \begin{pmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

$z \sim \text{Multi}(\pi)$ ，其概率分布满足：

$$\begin{aligned} p(z_k = 1) &= \pi_k \\ \text{s.t. } 0 &\leq \pi_k \leq 1, \sum_k \pi_k = 1. \end{aligned}$$

高斯混合模型

- 引入隐变量 z 进行建模，学生身高分布的**概率密度函数** $p(x)$ 如下：

$$p(x; \theta) = \sum_z p(x, z; \theta)$$

$$= \sum_z p(x|z; \theta)p(z)$$

$$= \sum_{k=1}^K p(x|z_k = 1; \theta_k)p(z_k = 1)$$

$$= \sum_{k=1}^K \mathcal{N}(x; \mu_k, \sigma_k^2) \pi_k$$

全概率公式

代入似然表达式

θ 表示高斯分布的均值，方差。

$p(x)$ 为多个高斯模型的加权和

$$\mathcal{N}(x; \mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}}$$

高斯混合模型

- 引入隐变量 z 进行建模，学生身高分布的**概率密度函数** $p(x)$ 如下：

$$p(x; \theta) = \sum_z p(x, z; \theta)$$

$$= \sum_z p(x|z; \theta)p(z)$$

$$= \sum_{k=1}^K p(x|z_k = 1; \theta_k)p(z_k = 1)$$

$$= \sum_{k=1}^K \mathcal{N}(x; \mu_k, \sigma_k^2) \pi_k$$

θ 表示高斯分布的均值，方差。

全概率公式

代入似然表达式

如何估计？

$$\mathcal{N}(x; \mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}}$$

混合高斯模型的极大似然估计

- 极大似然估计的目标：通过最大化**对数似然函数**，求解 $\{\pi, \mu, \Sigma\}$

➤ $\pi = \{\pi_1, \dots, \pi_K\}$: K 个高斯成分的混合系数;

➤ $\mu = \{\mu_1, \dots, \mu_K\}$: K 个高斯分布的均值;

➤ $\Sigma = \{\Sigma_1, \dots, \Sigma_K\}$: K 个高斯分布的协方差;

一般情况下，样本 x 是一个 n 维向量，此时的方差为 $n * n$ 维的协方差矩阵。

- N 个数据点的**对数似然函数**:

➤
$$\ln p(X; \pi, \mu, \Sigma) = \ln \left(\prod_{n=1}^N p(x_n; \pi, \mu, \Sigma) \right)$$

$p(X; \pi, \mu, \Sigma)$ 为 N 个数据点概率分布的连乘。

$$= \operatorname{argmax}_{\pi, \mu, \Sigma} \sum_{n=1}^N \ln \sum_z p(x_n | z; \mu, \Sigma) p(z; \pi)$$

混合高斯模型的极大似然估计

- N 个数据点的极大似然估计求解：

$$\pi^*, \mu^*, \Sigma^* = \operatorname{argmax}_{\pi, \mu, \Sigma} \ln p(X; \pi, \mu, \Sigma) = \operatorname{argmax}_{\pi, \mu, \Sigma} \sum_{n=1}^N \ln \sum_z p(x_n | z; \mu, \Sigma) p(z; \pi)$$

- 分别对 π , μ , Σ 求偏导，并令偏导数等于零，这样能否求解？

存在隐变量

数据点 x_n 依赖于隐变量 z ，由于 z 是未观测到的，需要对 z 求和来消除影响。这导致了 \ln 函数中存在求和项，无法得到解析解。

非凸性

由于GMM的对数似然函数通常是非凸的，直接使用极大似然估计求解可能会陷入局部极大值而无法找到全局最优解。

EM算法求解混合高斯问题

- 因此，引入EM算法

- EM算法是一种分步迭代优化算法，适用于包含隐变量的极大似然估计问题
- 在高斯混合问题中，EM算法通过迭代更新隐变量 z_n 的估计值和GMM的模型参数，使得对数似然函数逐步逼近最大值。通过计算 z_n 的估计值，可以消除隐变量的影响，去掉ln函数中的求和项
- 具体地，EM算法包含两个步骤。E-step：固定GMM的模型参数，计算隐变量 z_n 的估计值，即样本属于每个高斯分布的后验概率；M-step：已知隐变量 z_n 的估计值，通过极大似然估计更新GMM的模型参数

EM算法推导

- E-step (固定GMM的模型参数，调整样本属于每个高斯分布的后验概率)

$$\gamma_{nk} = p(z_k = 1 | x_n)$$

$$= \frac{p(z_k = 1)p(x_n | z_k = 1)}{\sum_{j=1}^K p(z_j = 1)p(x_n | z_j = 1)}$$

$$= \frac{\pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}$$

在EM算法中， γ_{nk} 表示样本 x_n 属于第 k 个高斯分布的后验概率，也称为责任度 (Responsibility)，是一种“软”分配。

与之相对，K-Means使用 γ_{nk} 表示样本 x_n 是否属于第 k 个聚类簇，是一种“硬”分配。

EM算法推导

- M-step (固定 γ_{nk} , 调整GMM的模型参数)

$$\ln p(X|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln p(x_n|z_n; \mu, \Sigma) + \ln p(z_n|\pi)$$

对 μ_k 求导
→

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} x_n \quad N_k = \sum_{n=1}^N \gamma_{nk}$$

N_k 可以看作第 k 个高斯分布所包含的样本数。

对 Σ_k 求导
→

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} (x_n - \mu_k)(x_n - \mu_k)^T$$

对 π_k 求导
→

$$\pi_k = \frac{N_k}{N}$$

对 π 求导时必须考虑一个约束条件： $\sum_{k=1}^K \pi_k = 1$, 使用拉格朗日乘子法。

EM算法求解混合高斯问题

选取合适的K并初始化参数: π^0, μ^0, Σ^0

交替执行E-step和M-step步骤直至收敛:

K-Means可以看作是EM算法的一种特例。它交替执行两步走策略：在第一步（对应E-step），将每个样本分配给最近的聚类中心；在第二步（对应M-step），通过最小化准则函数，重新计算聚类中心。

E-step:

$$\gamma_{nk}^{(t+1)} = \frac{\pi_k^{(t)} \mathcal{N}(x_n | \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_{j=1}^K \pi_j^{(t)} \mathcal{N}(x_n | \mu_j^{(t)}, \Sigma_j^{(t)})}$$

调整样本属于每个高斯分布的后验概率

M-step:

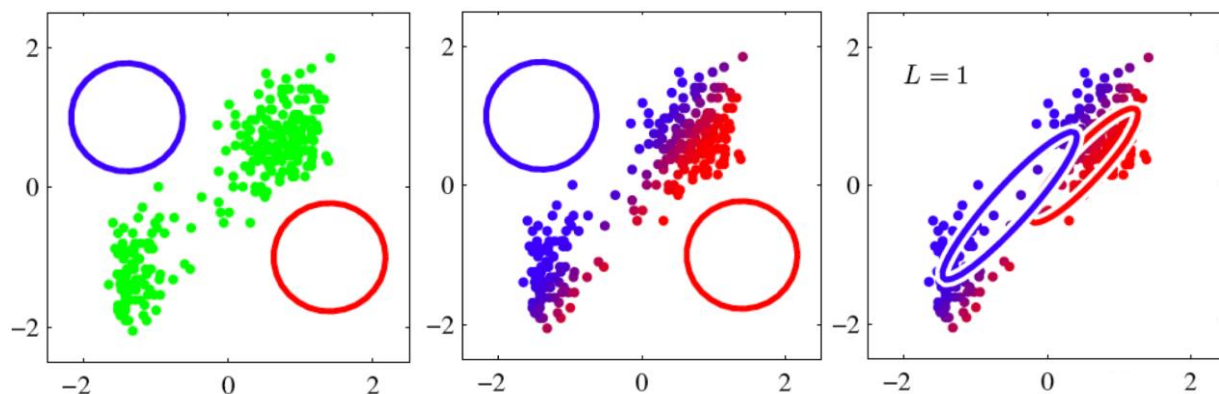
$$\mu_k^{(t+1)} = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk}^{(t+1)} x_n \quad N_k = \sum_{n=1}^N \gamma_{nk}^{(t+1)}$$

$$\Sigma_k^{(t+1)} = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk}^{(t+1)} (x_n - \mu_k^{(t+1)}) (x_n - \mu_k^{(t+1)})^T$$

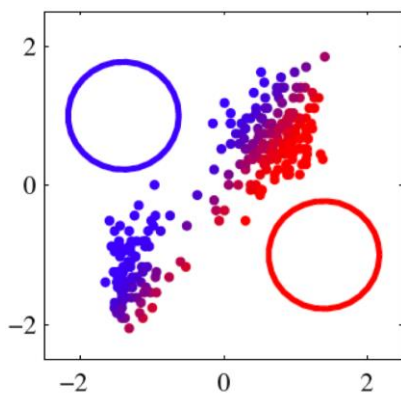
$$\pi_k^{(t+1)} = \frac{N_k}{N}$$

最优化GMM的模型参数估计

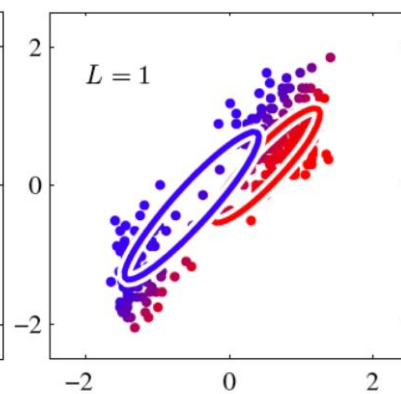
EM过程图示



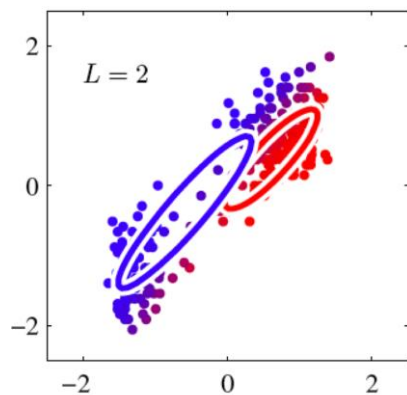
(a)



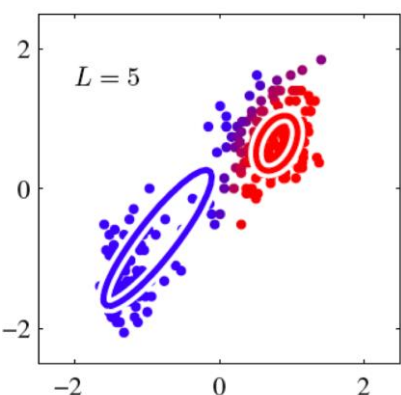
(b)



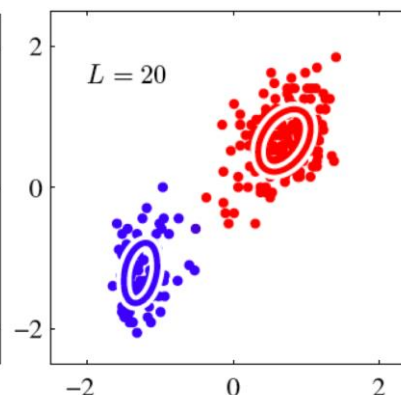
(c)



(d)



(e)



(f)

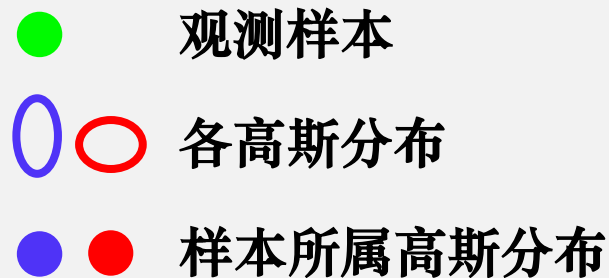
EM算法流程

(a): 初始化高斯混合分布的模型参数

(b): E-步计算 γ_{nk}

(c): M-步更新 μ_k, Σ_k, π_k

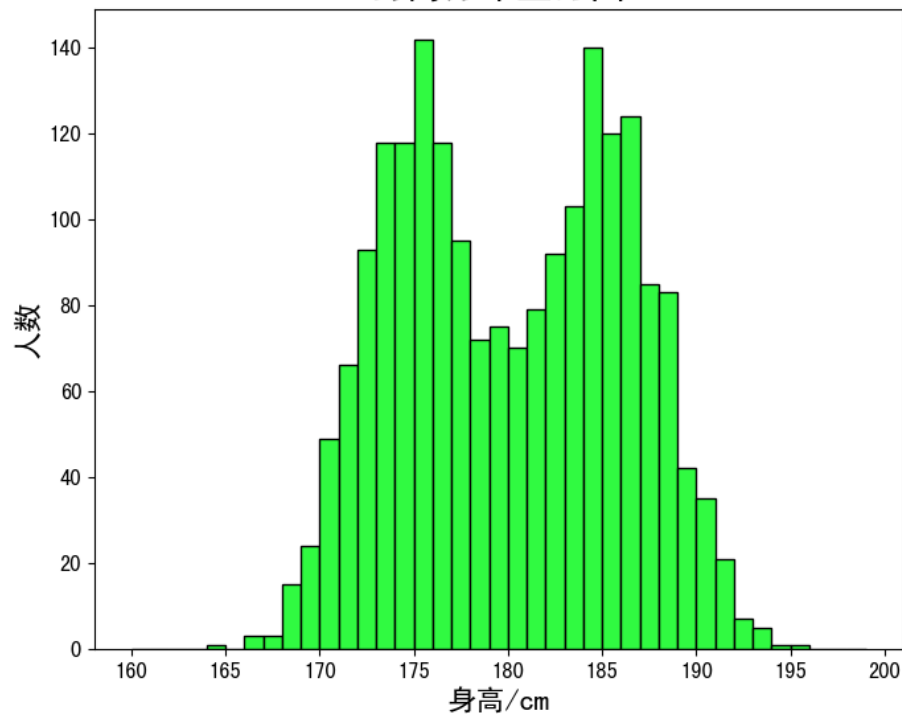
(d)-(f): 迭代执行E-M过程直至达到停止条件



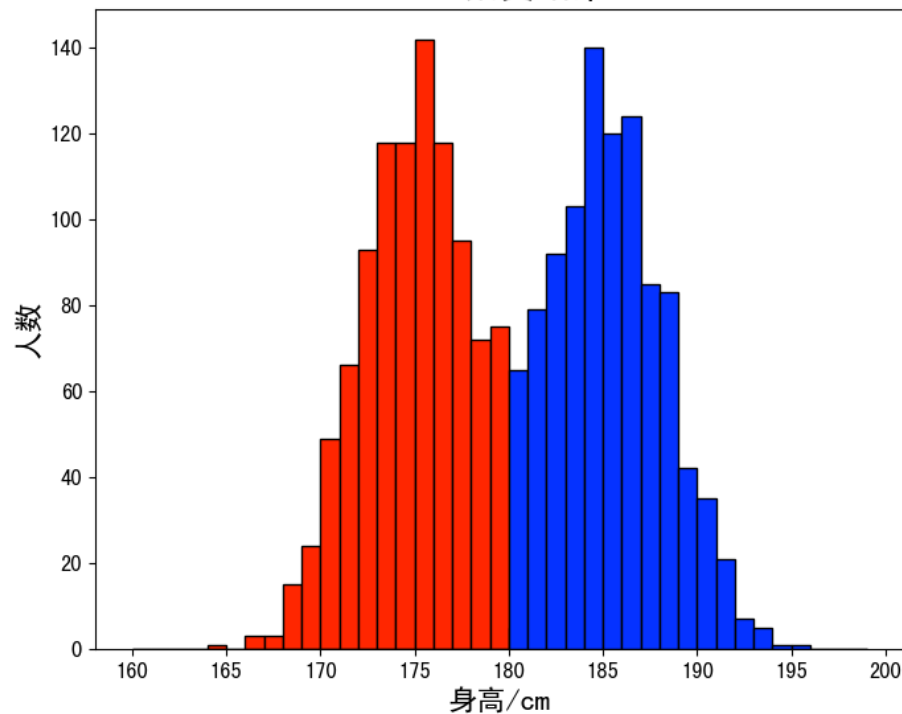
高斯混合模型的应用

● 不同学院学生的身高分布问题

身高分布直方图



GMM 聚类结果

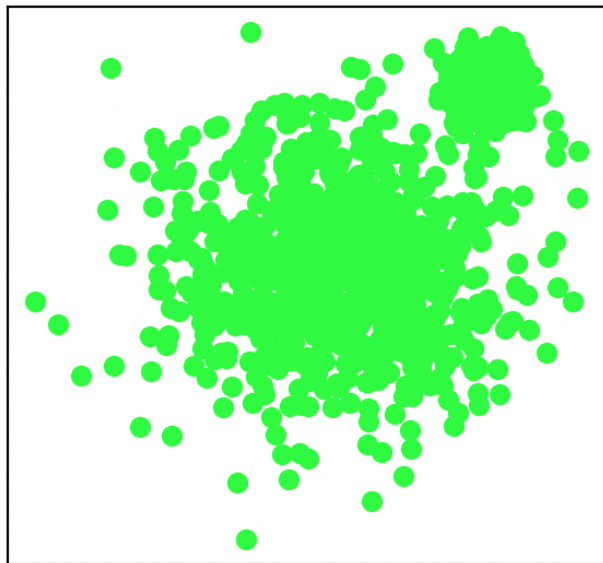


GMM的均值: [174.98, 185.22]

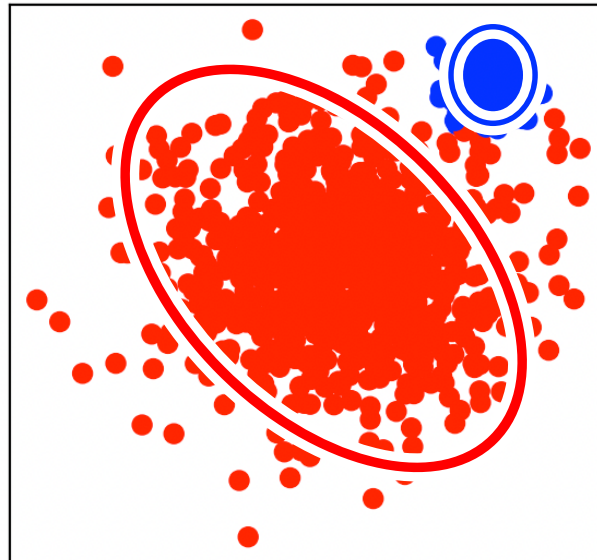
GMM的方差: [9.04, 9.69]

GMM的混合系数: [0.496, 0.504]

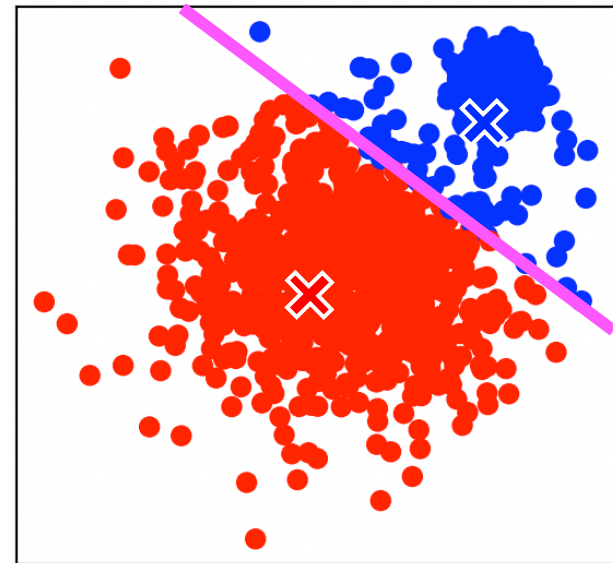
GMM v.s. K-Means



初始样本



GMM



K-Means

GMM v.s. K-Means

- K-Means和GMM都是常用的聚类算法。

K-Means

GMM

聚类类型

执行**硬聚类**

执行**软聚类**，通过 $\operatorname{argmax}_{k \in \{1, 2, \dots, K\}} \gamma_{nk}$ 变为硬聚类

隐变量

使用 r_n 表示每个样本 x_n 所属的聚类簇

使用 z 来表示样本 x 来自第 k 个高斯分布

聚类步骤

- 1) 将每个样本分配给最近的聚类中心（**“硬”分配**）；
- 2) 最小化准则函数，重新计算聚类中心；

- 1) 计算样本属于每个高斯分布的后验概率（**“软”分配**）；
- 2) 最优化GMM的模型参数估计；

适用场景

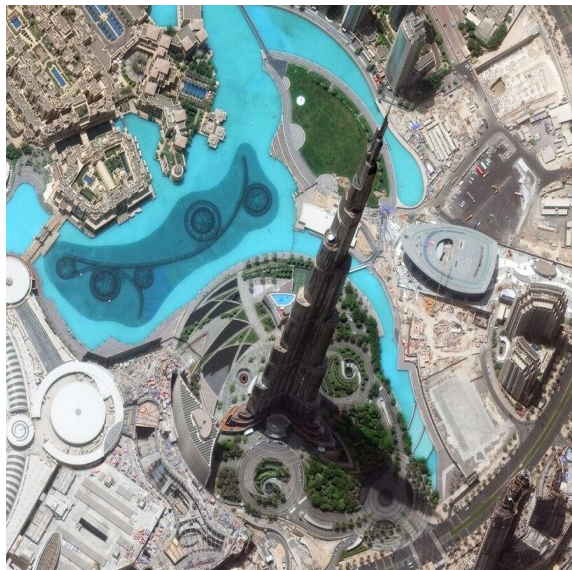
分布较为均匀、形状接近球形
的数据；计算效率较高，适合处理大规模的数据集

分布是多个高斯分布混合而成的、形状复杂的数据；可以执行软聚类；计算复杂度较高

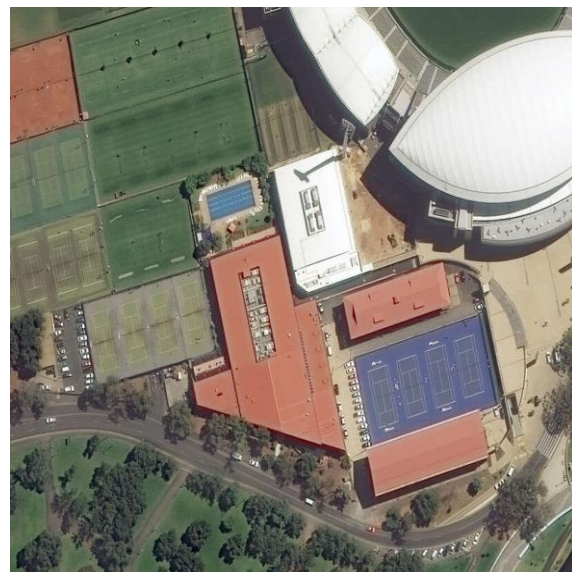
高斯混合模型的应用

● 遥感图像分割

- 遥感图像分析为资源管理、环境监测、灾害评估、土地利用规划等领域提供了重要信息和决策支持。遥感图像分割是将遥感图像中的像素划分为**具有相似特征的区域**的过程。通过对遥感图像进行分割，可以精确地识别和定位地物。



遥感图像A



遥感图像B

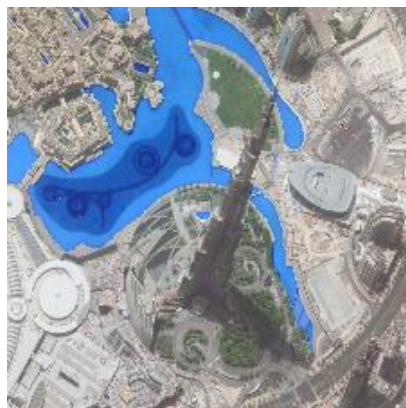
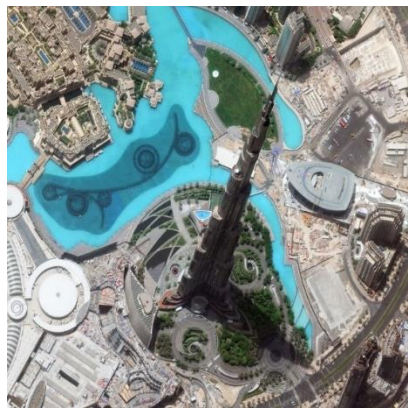
数据源：<https://www.o-map.cn/>

高斯混合模型的应用

● 利用GMM进行遥感图像分割的步骤：

- 输入为 $H \times W \times 3$ 的遥感图像，将每个像素点视为一个无标注样本（总共 HW 个），每个样本（像素点）是维度为3的颜色向量；
- 选取合适的聚类数量 K ，初始化GMM的模型参数 π, μ, Σ ；
- 交替执行E-step和M-step，直至满足收敛条件或达到最大迭代次数 N ；
- 通过计算 $\arg\max_{k \in \{1, 2, \dots, K\}} \gamma_{nk}$ ，将每个样本（像素点）分配给概率最高的聚类簇，得到最终的分割结果。

GMM对遥感图像的分割结果



$K = 2$

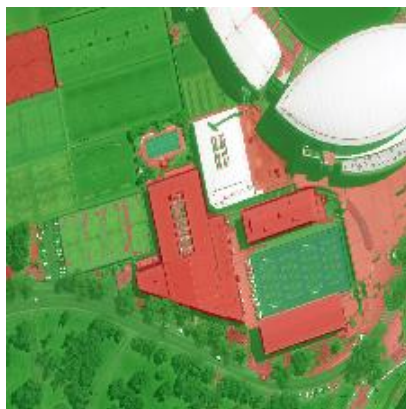
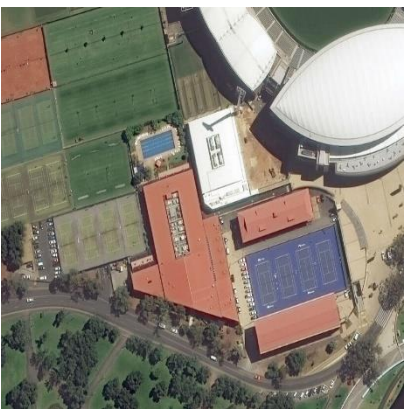


$K = 3$

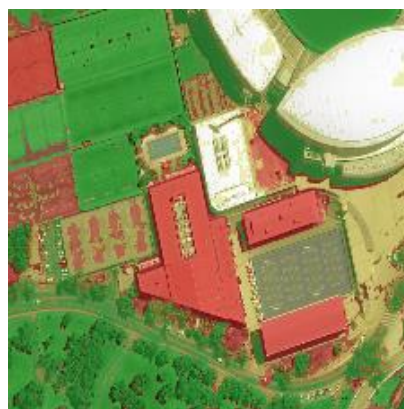


$K = 4$

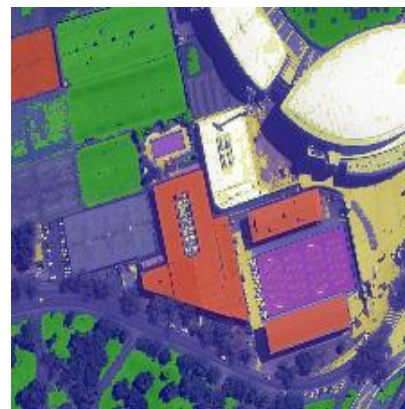
水体
草地
建筑
道路



$K = 3$



$K = 4$



$K = 6$

植被
农田
房屋
树木
建筑
道路

EM算法

- EM算法的介绍
- EM算法的基本思路
- EM算法的具体求解
- EM算法收敛性证明

EM算法的介绍

● EM算法的介绍

- 回顾：高斯混合模型通过EM算法分步迭代优化，求解含有隐变量的极大似然估计。似然函数为

$$\ell(\pi, \mu, \Sigma) = \ln \sum_z p(X|z; \mu, \Sigma) p(z; \pi)$$

- 一般的含有隐变量的极大似然估计问题，似然函数形式如下

$$\begin{aligned} \ell(\theta) &= \ln p(X; \theta) = \ln \sum_Z p(X, Z | \theta) \\ &= \ln \sum_Z p(X|Z, \theta) p(Z|\theta) \end{aligned}$$

记总体概率密度函数为 $p(X; \theta)$ ，样本为 $X = \{x_1, \dots, x_N\}$

ln函数中存在求和项，无法得到解析解。

- 期望最大化（Expectation Maximization, EM）算法：含隐变量的极大似然估计问题求解方法。

EM算法的基本思路

● 基本思路

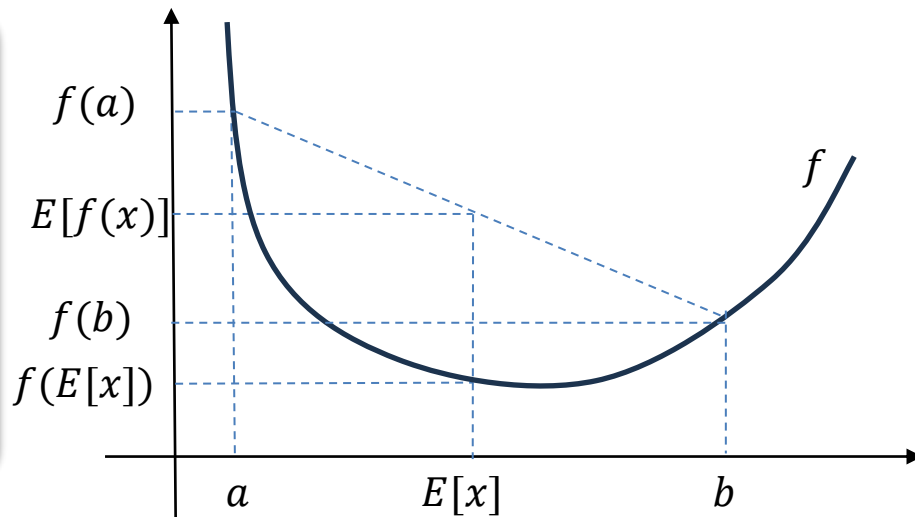
➤ **步骤1.** 得到下界：通过为似然函数寻找有解析解形式的下界进行参数估计

■ GMM求解时去掉ln函数中的求和号，得到似然函数的解析解。故考虑将下界表示为该形式。

Jensen不等式： 如果一个函数 $f(x)$ 为凸函数，即 $f(x)'' \geq 0$ ，则有 $E[f(x)] \geq f(E[x])$ ，当且仅当 $x = E[x]$ ，即 x 为常量时，等号成立。

对于 $-\ln(x)$ 该式成立： $E[-\ln(x)] \geq -\ln(E[x])$

即： $E[\ln(x)] \leq \ln(E[x])$



EM算法的具体求解

- 步骤1.利用Jensen不等式，取得下界

- 要最大化 $\ell(\theta)$ ，可以考虑利用迭代的方式，当前迭代得到的 θ 能使上一步的 $\ell(\theta^{(t)})$ 变大，即 $\ell(\theta) > \ell(\theta^{(t)})$. 为此，考虑两者的差

$$\ell(\theta) - \ell(\theta^{(t)}) = \ln \sum_Z p(X|Z, \theta)p(Z|\theta) - \ln p(X|\theta^{(t)})$$

- 引入隐变量的概率分布 $Q(Z)$ ，以通过Jensen不等式更好地建立 $\ell(\theta)$ 的下界

$$\ell(\theta) - \ell(\theta^{(t)}) = \ln \sum_Z Q(Z) \frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)} - \ln p(X|\theta^{(t)})$$

EM算法的具体求解

● 步骤1.利用Jensen不等式，取得下界

➤ 引入隐变量的概率分布 $Q(Z)$ 并通过Jensen不等式建立 $\ell(\theta)$ 的下界

$$\ell(\theta) - \ell(\theta^{(i)}) = \ln \sum_Z Q(Z) \frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)} - \ln p(X|\theta^{(i)})$$

即 $\frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)}$ 关于隐变量 Z 的期望

$E_{Z \sim Q(Z)} \left[\frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)} \right]$

↓ Jensen不等式: $E[\ln(x)] \leq \ln(E[x])$

$$\geq \sum_Z Q(Z) \ln \frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)} - \ln p(X|\theta^{(i)})$$
$$= \sum_Z Q(Z) \ln \frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)p(X|\theta^{(i)})}$$

EM算法的具体求解

- 步骤1.利用Jensen不等式，取得下界

- 引入隐变量的概率分布 $Q(Z)$ 并通过Jensen不等式建立 $\ell(\theta)$ 的下界

$$\ell(\theta) \geq \ell(\theta^{(i)}) + \sum_Z Q(Z) \ln \frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)p(X|\theta^{(i)})}$$

- 令 $B(\theta, \theta^{(i)}) = \ell(\theta^{(i)}) + \sum_Z Q(Z) \ln \frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)p(X|\theta^{(i)})}$

- 则有 $\ell(\theta) \geq B(\theta, \theta^{(i)})$ ，即 $B(\theta, \theta^{(i)})$ 是 $\ell(\theta)$ 的一个下界。

- 故当 $B(\theta, \theta^{(i)})$ 增大时， $\ell(\theta)$ 也将同时增加。

EM算法的具体求解

● 步骤1.利用Jensen不等式，取得下界

➤ Jensen不等式 $E[\ln(x)] \leq \ln(E[x])$ 取等条件为： $x = E[x]$, x 为常数

➤ 即 $E_{Z \sim p(Z|X, \theta^{(i)})} \left[\frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)} \right]$ 即 $\frac{p(X|Z, \theta)p(Z|\theta)}{Q(Z)} = \frac{p(X, Z|\theta)}{Q(Z)} = c$

➤ 移项得 $p(X, Z|\theta) = c \cdot Q(Z)$

$$Q(Z) = \frac{p(X, Z|\theta)}{c}$$

移项

同时基于 z 求和

$$\sum_Z p(X, Z|\theta) = c \cdot \sum_Z Q(Z)$$

$$\sum_Z p(X, Z|\theta) = c$$

z 密度函数和为1

代入左式的 c

$$= \frac{p(X, Z|\theta)}{\sum_Z p(X, Z|\theta)}$$

$$= \frac{p(X, Z|\theta)}{p(X|\theta)} = p(Z|X, \theta)$$

Z 的后验概率分布

EM算法的具体求解

● 步骤2.提升下界

➤ 为使 $\ell(\theta)$ 取得最大，更新 $\theta^{(i+1)} = \theta$ ，即 $B(\theta, \theta^{(i)})$ 取得最大的 θ

$$\begin{aligned}\theta^{(i+1)} &= \operatorname{argmax}_{\theta} B(\theta, \theta^{(i)}) \\&= \operatorname{argmax}_{\theta} (\ell(\theta^{(i)}) + \sum_Z p(Z|X, \theta^{(i)}) \ln \frac{p(X|Z, \theta)p(Z|\theta)}{p(Z|X, \theta^{(i)})p(X|\theta^{(i)})}) \\&= \operatorname{argmax}_{\theta} (\sum_Z p(Z|X, \theta^{(i)}) \ln(p(X|Z, \theta)p(Z|\theta))) \\&= \operatorname{argmax}_{\theta} (\sum_Z p(Z|X, \theta^{(i)}) \ln p(X, Z|\theta)) \\&= \operatorname{argmax}_{\theta} Q(\theta, \theta^{(i)})\end{aligned}$$

消去与 θ 求解
无关的 $\theta^{(i)}$

EM算法的具体求解

● EM算法的一般形式

初始化参数: $\theta^{(0)}$

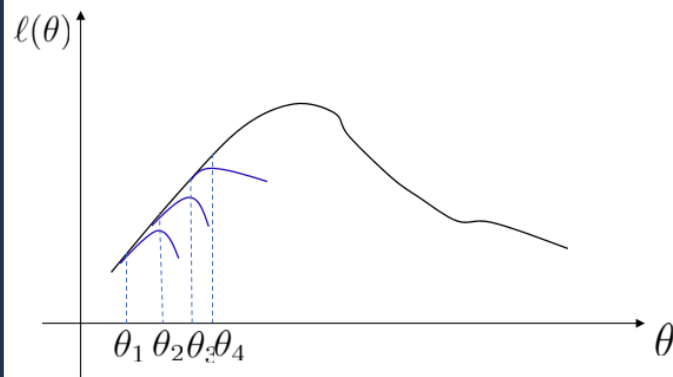
交替执行E-step和M-step步骤直至收敛:

E-step: $Q(Z) = p(Z|X, \theta)$

利用Jensen不等式, 取得下界

M-step: $\theta^{(i+1)} = \operatorname{argmax}_{\theta} Q(\theta, \theta^{(i)})$

最优化参数估计, 提升下界



迭代优化 θ 直至收敛

EM算法的具体求解

● EM算法在高斯混合模型上的求解

初始化参数: $\theta^{(0)}$

交替执行E-step和M-step步骤直至收敛

$$\ell(\theta) = \ln \sum_z p(X|Z, \theta)p(Z|\theta)$$

E-step: $Q(Z) = p(Z|X, \theta^{(i)})$ 代入求解 $\gamma_{nk}^{(i+1)} = \frac{\pi_k^{(i)} \mathcal{N}(x_n | \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{j=1}^K \pi_j^{(i)} \mathcal{N}(x_n | \mu_j^{(i)}, \Sigma_j^{(i)})}$

求Z的密度函数, 令Jensen不等式取等

M-step: $\theta^{(i+1)} = \operatorname{argmax}_{\theta} Q(\theta, \theta^{(i)})$ 代入求解 $\mu_k^{(i+1)} = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk}^{(i+1)} x_n$

$$N_k = \sum_{n=1}^N \gamma_{nk}^{(i+1)} \quad \pi_k^{(i+1)} = \frac{N_k}{N} \quad \Sigma_k^{(i+1)} = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk}^{(i+1)} (x_n - \mu_k^{(i+1)}) (x_n - \mu_k^{(i+1)})^T$$

最优化参数估计

EM算法的收敛性证明

● 单调有界定理

若数列 $\{a_n\}$ 递增有上界（递减有下界），则数列 $\{a_n\}$ 收敛

➤ 似然函数有上界，即证明 $\ell(\theta^{(i)}) \leq \ell(\theta^{(i+1)})$

➤ 选择 $Q(Z) = p(Z|X, \theta)$ 时Jensen不等式成立，即

$$\ell(\theta) = \sum_Z Q(Z) \ln \frac{p(X, Z|\theta)}{Q(Z)}$$

➤ 且 $\theta^{(i+1)}$ 通过最大化上式求得，即

$$\begin{aligned} \ell(\theta^{(i+1)}) &\geq \sum_Z Q(Z) \ln \frac{p(X, Z|\theta^{(i+1)})}{Q(Z)} \\ &\geq \sum_Z Q(Z) \ln \frac{p(X, Z|\theta^{(i)})}{Q(Z)} = \ell(\theta^{(i)}) \end{aligned}$$

取等条件为 $Q(Z) = p(Z|X, \theta)$,
且 $\theta^{(i)} = \theta^{(i+1)}$

取等条件为要求 $\theta^{(i+1)}$ 最大化
 $\sum_Z Q(Z) \ln \frac{p(X, Z|\theta)}{Q(Z)}$

似然函数存在极值，且EM步骤单调收敛，因此必能找到极值

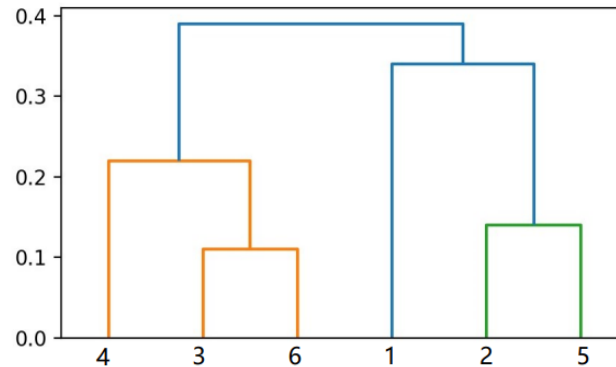
层次聚类

- 认识层次聚类
- AGNES聚类算法
- DIANA聚类算法
- 层次聚类的应用

层次聚类

● 什么是层次聚类

- 层次聚类希望在**不同层次**对数据集进行划分，从而形成**树形**的聚类结构。可以在**不同的尺度（层次）**上展示数据集的聚类情况



- 数据集的划分可以采用“自底向上”的聚合策略，也可采用“自顶向下”的分拆策略。

自底向上的聚合策略：**AGNES**, BIRCH, ROCK...

自顶向下的分拆策略：**DIANA**, Bisecting K-Means, PDDP...

AGNES聚类算法

● AGNES算法

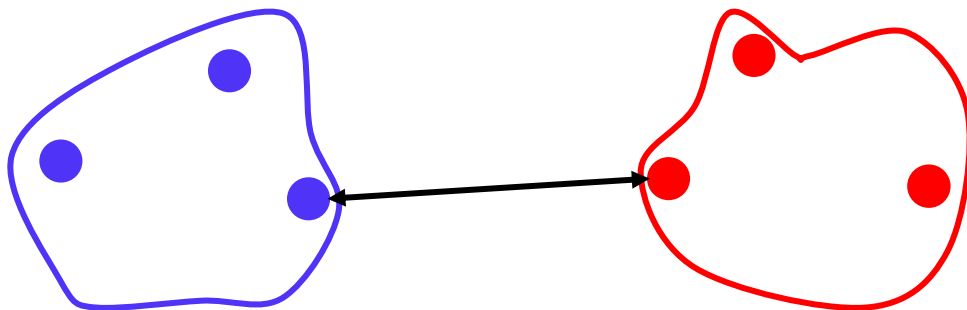
- AGNES (**AG**glomerative **NE**Sting)是一种采用自底向上聚合策略的**层次聚类**算法
- 基本思想：先将数据集中的**每个样本**看作一个**初始聚类簇**，然后逐步找出**距离**最近的两个簇进行合并，该过程不断重复，直到只剩下一个簇为止
- **距离**：每个簇是一个样本集合，因此只需采用关于**集合的某种距离**即可

AGNES聚类算法

● 集合的距离

➤ 给定聚类簇 C_i 与 C_j ，可以定义以下几种距离：

最小距离： $d_{min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的**最近样本**决定



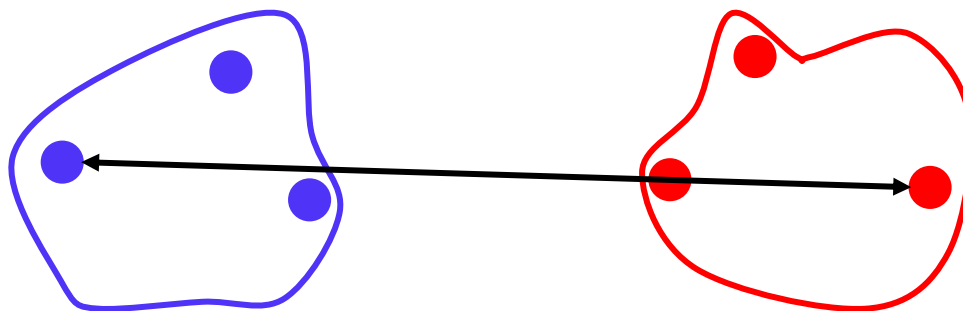
AGNES聚类算法

● 集合的距离

➤ 给定聚类簇 C_i 与 C_j ，可以定义以下几种距离：

最小距离： $d_{min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的最近样本决定

最大距离： $d_{max}(C_i, C_j) = \max_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的最远样本决定



AGNES聚类算法

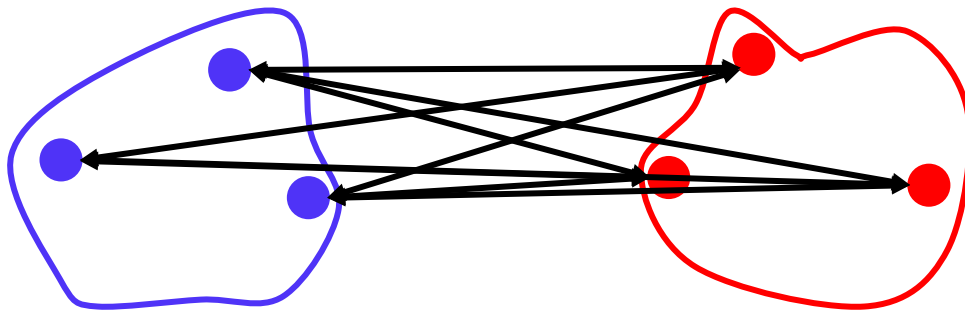
● 集合的距离

➤ 给定聚类簇 C_i 与 C_j ，可以定义以下几种距离：

最小距离： $d_{min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的**最近样本**决定

最大距离： $d_{max}(C_i, C_j) = \max_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的**最远样本**决定

平均距离： $d_{avg}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{z \in C_j} \text{dist}(x, z)$ 由两个簇的**所有样本**决定



AGNES聚类算法

● 集合的距离

➤ 给定聚类簇 C_i 与 C_j ，可以定义以下几种距离：

最小距离： $d_{min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的最近样本决定

最大距离： $d_{max}(C_i, C_j) = \max_{x \in C_i, z \in C_j} \text{dist}(x, z)$ 由两个簇的最远样本决定

平均距离： $d_{avg}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{z \in C_j} \text{dist}(x, z)$ 由两个簇的所有样本决定

样本之间的距离 $\text{dist}(\cdot, \cdot)$ 可以使用不同的度量方式，如欧几里得距离、曼哈顿距离等

AGNES聚类算法

● AGNES算法的流程

- **步骤1.** 初始化每个样本作为一个聚类簇 C_i , $i = 1, 2, \dots, N$;
- **步骤2.** 计算每两个簇 C_i 和 C_j 之间的距离 $d(C_i, C_j)$;
- **步骤3.** 找出距离最近的两个簇进行合并;
- **步骤4.** 重复步骤2.-步骤3., 直到只剩下一个簇。

层次聚类

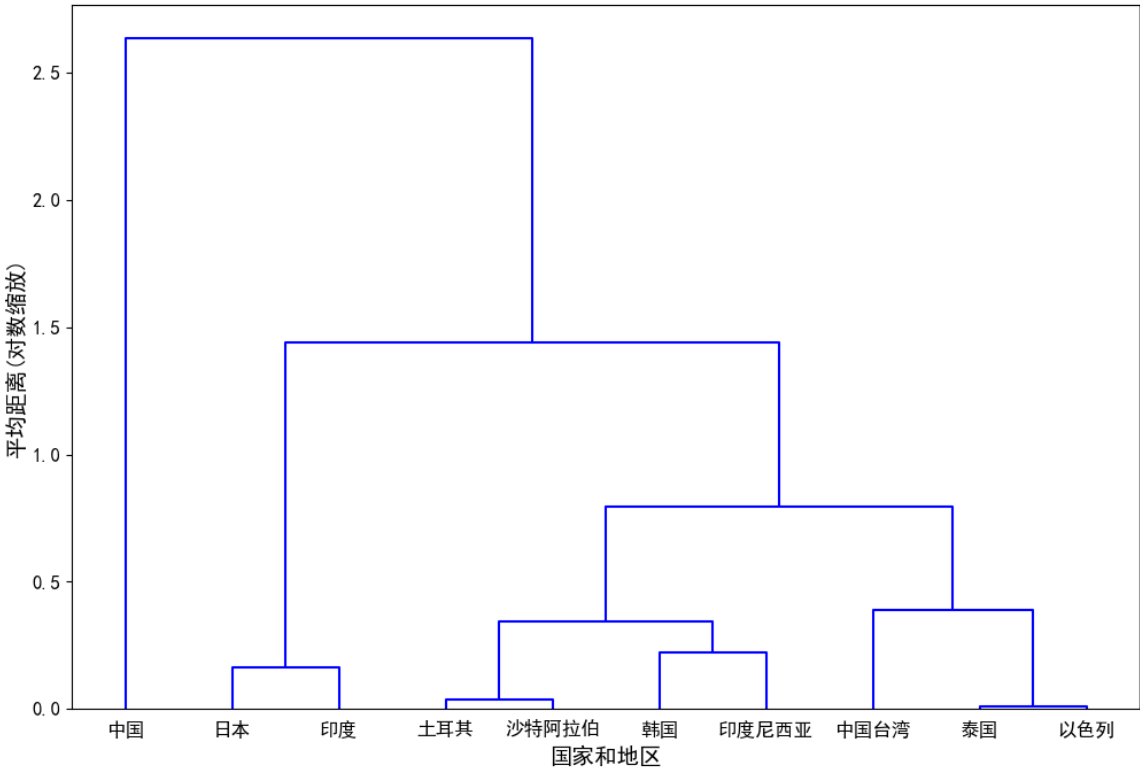
● 例子

- 我们有2023年亚洲前十个国家和地区的GDP总量数据（单位：亿美元），如右表所示。不同国家和地区的经济指标通常呈现分层现象，如发达国家和发展中国家、沿海国家和内陆国家等
- 问：如何根据这十个国家和地区的GDP总量数据进行聚类，并展示不同国家和地区在经济规模上的相似性？

	国家和地区	2023年GDP总量
1	中国	176620
2	日本	42129
3	印度	35721
4	韩国	17128
5	印度尼西亚	13712
6	土耳其	11085
7	沙特阿拉伯	10676
8	中国台湾	7566
9	泰国	5149
10	以色列	5095

层次聚类

● AGNES算法的示例



	国家和地区	2023年GDP总量
1	中国	176620
2	日本	42129
3	印度	35721
4	韩国	17128
5	印度尼西亚	13712
6	土耳其	11085
7	沙特阿拉伯	10676
8	中国台湾	7566
9	泰国	5149
10	以色列	5095

DIANA聚类算法

● DIANA算法

- DIANA (**DI**visive **AN**alysis clustering)是一种采用自顶向下拆分策略的**层次聚类**算法
- 基本思想：先将整个数据集的**所有样本看成一个初始聚类簇**，然后逐步**拆分**成更小的簇，直到每个簇只包含一个样本为止
- **簇的拆分**：在当前簇中，找到两个**最不相似**的样本作为两个新的簇，然后将剩余的样本分配到离它们更近的两个新的簇之一

- 自顶向下 v.s. 自底向上：自顶向下方法适合处理中等规模、具有明显全局结构的数据集；自底向上方法适合较小规模、具有明确局部结构的数据集

讨论

● 层次聚类 v.s. K-Means

K-Means

层次聚类

适用场
景

分布较为均匀、形状接近球形
的数据，大规模的数据集。

具有层次结构的数据，较小、中等
规模的数据集。

优点

结果易于解释；计算效率高。

能够直观地展示数据的层次结构，
反应聚类簇逐步聚合或分拆的过程；
无需预先指定聚类簇的数量；对初
始条件不敏感；可以识别数据中的
噪声点。

缺点

需要预先指定聚类簇的数量；
聚类结果依赖于初始中心点的
选择；容易受到噪声点的影响。

计算复杂度较高；聚合或拆分一旦
完成就不能撤销。

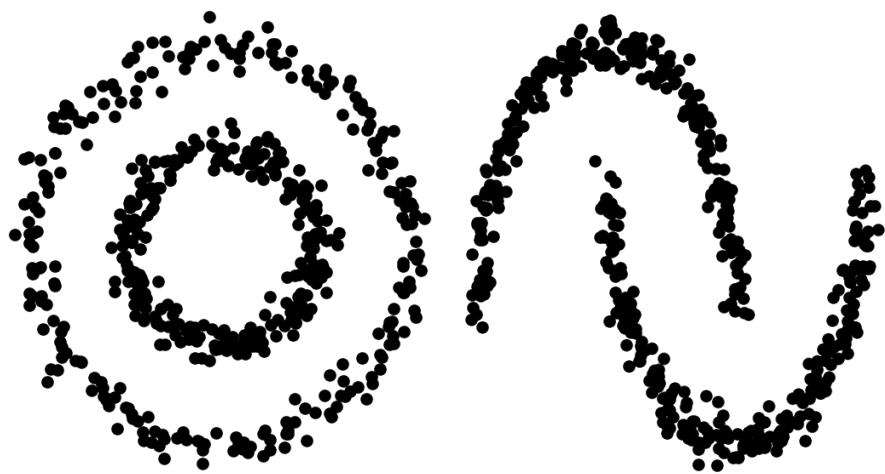
密度聚类

- 认识密度聚类
- DBSCAN算法

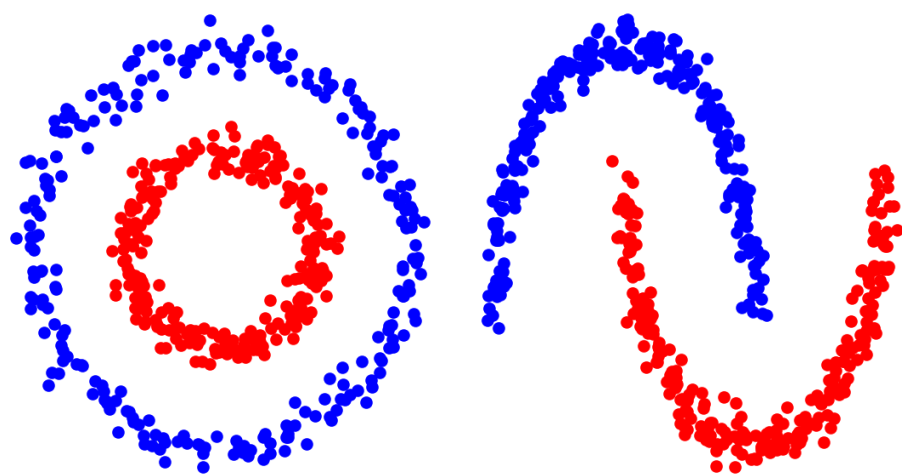
密度聚类

● 非凸集数据

- 如果 S 中任意两点的连线上的点都在集合 S 内，那么集合 S 称为凸集。反之，为非凸集。



输入数据

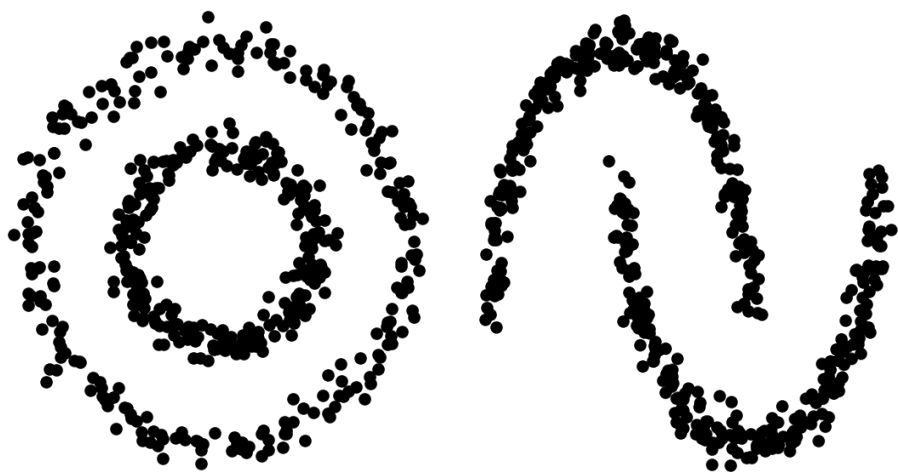


聚类

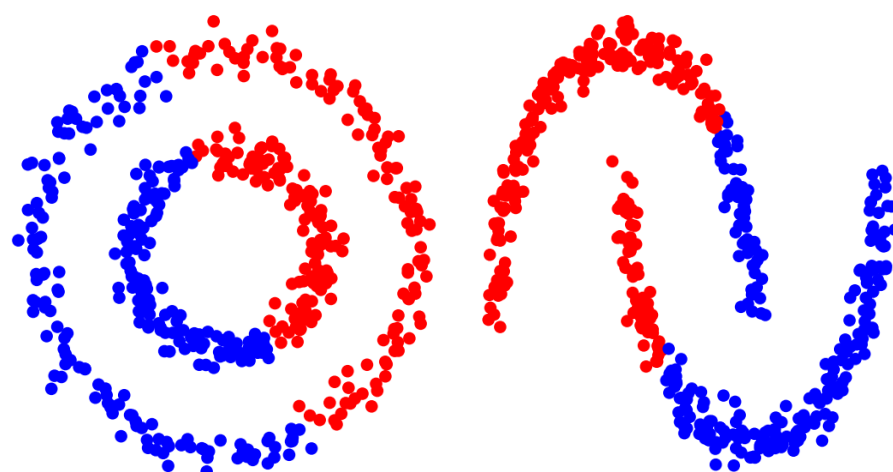
密度聚类

● 非凸集数据

- 如果 S 中任意两点的连线上的点都在集合 S 内，那么集合 S 称为凸集。反之，为非凸集。



输入数据



K-Means聚类 (K=2)

DBSCAN聚类算法

● 密度聚类

- 密度聚类算法从样本**密度**的角度来考察样本之间的可连接性，并基于可连接样本不断扩展聚类簇以获得最终的聚类结果。
密度聚类非常适用于**非凸集数据**。

● DBSCAN算法

- DBSCAN(**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise)是一个比较有代表性的**密度聚类**算法。
- 基本思想：将聚类簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为簇，并可在有噪声的数据中发现**任意形状**的簇。

DBSCAN聚类算法

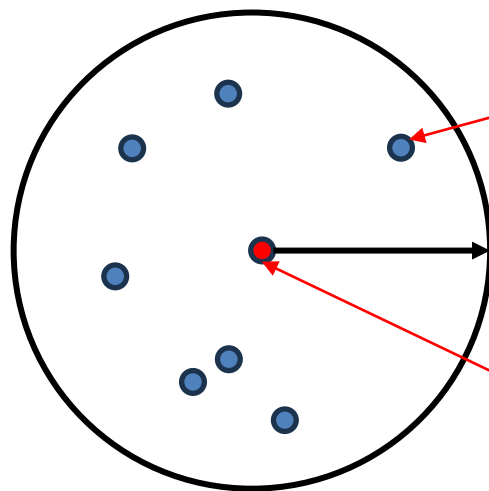
● DBSCAN算法的概念

- **密度**：空间中任意一点的密度是以该点为圆心，以扫描半径构成的圆形区域内包含的点数目。
- 使用**两个超参数**：扫描半径 (eps)和最小包含点数(minPts)来确定簇的数量，而不是预先给定簇的数目
- **扫描半径 (eps)**：用于定位点/检查任何点附近密度的距离度量。
- **最小包含点数(minPts)**：聚集在一起的最小点数（阈值）。如果大于阈值，则该区域被认为是稠密的。

DBSCAN聚类算法

● DBSCAN算法的概念

- **核心点**：以该点为圆心，在半径 ϵ 内含有超过 minPts 数目的点。
- **边界点**：以该点为圆心，在半径 ϵ 内点的数量小于 minPts ，但是落在核心点的邻域内的点。



边界点：若领域内点不超过 MinPts 个($\text{MinPts}=5$)

核心点：领域内点的个数超过 MinPts ($\text{MinPts}=5$)

- **噪声点**：既非核心点也非边界点的点。

DBSCAN聚类算法

● DBSCAN算法的流程

- **步骤1.** 将所有点标记为核心点、边界点或噪声点；
- **步骤2.** 如果选择的点是核心点，则找出所有从该点出发的密度可达对象形成簇；
- **步骤3.** 如果该点是非核心点，将其指派到一个与之关联的核心点的簇中；
- **步骤4.** 重复以上步骤，直到所有点都被处理过。

DBSCAN聚类算法

- DBSCAN算法的示例

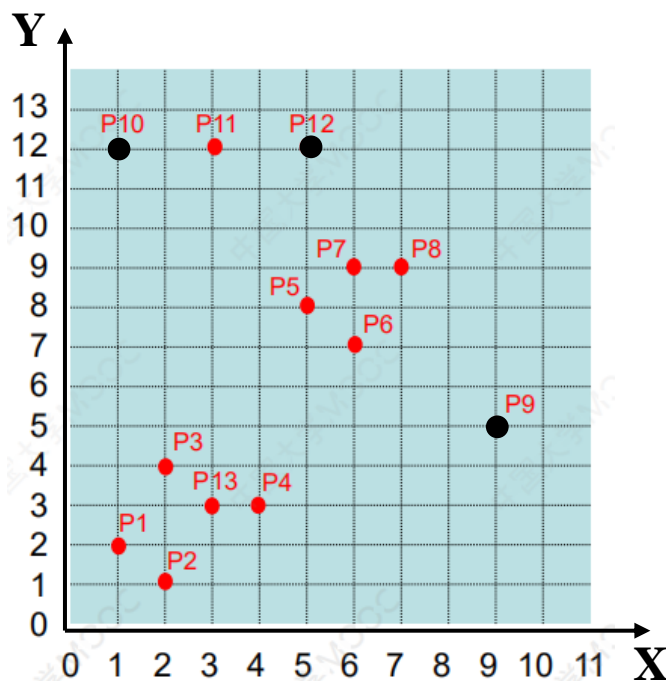
➤ 13个样本点，使用DBSCAN进行聚类

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13
X	1	2	2	4	5	6	6	7	9	1	3	5	3
Y	2	1	4	3	8	7	9	9	5	12	12	12	3

DBSCAN聚类算法

● DBSCAN算法的示例

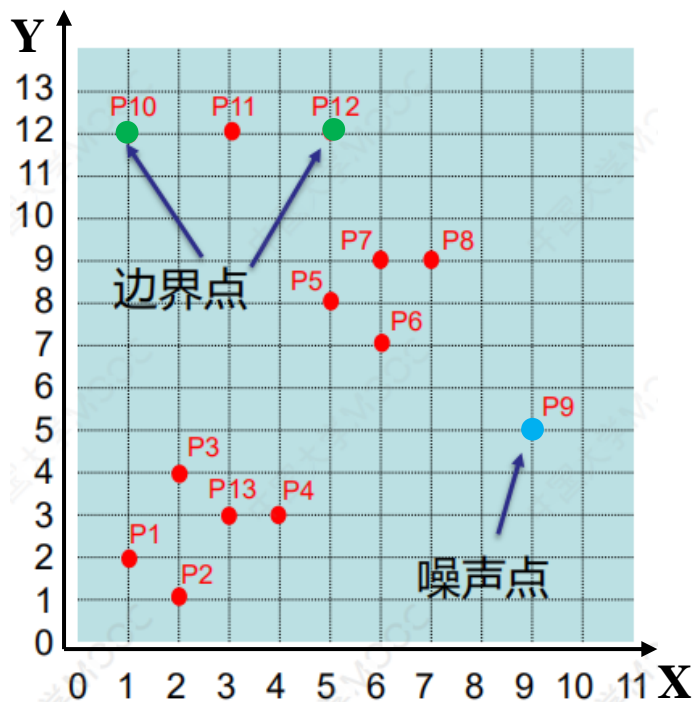
- 对每个点计算其邻域 $\text{eps}=3$ 内的点的集合
- 集合内点的个数超过 $\text{minPts}=3$ 的点为核心点。



DBSCAN聚类算法

● DBSCAN算法的示例

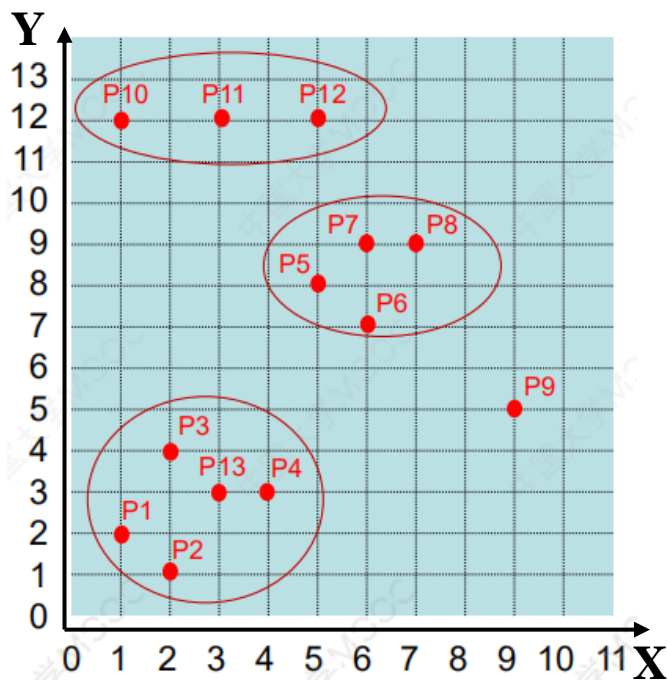
- 查看剩余点是否在核点的邻域内，若在，则为边界点，否则为噪声点。



DBSCAN聚类算法

● DBSCAN算法的示例

- 将距离不超过 $\text{eps}=3$ 的点相互连接，构成一个簇，核心点邻域内的点也会被加入到这个簇中。



讨论

● 密度聚类 v.s. K-Means

	K-Means	密度聚类
适用场景	分布较为均匀、形状接近球形（凸集）的数据，大规模的数据集。	可以发现任意形状的簇，特别适用于非凸集数据，能够有效处理有噪声点的数据集。
优点	结果易于解释；计算效率高。	无需预先指定聚类簇的数量。
缺点	需要预先指定聚类簇的数量；聚类结果依赖于初始中心点的选择；容易受到噪声点的影响。	对参数扫描半径和最小包含点数较为敏感。

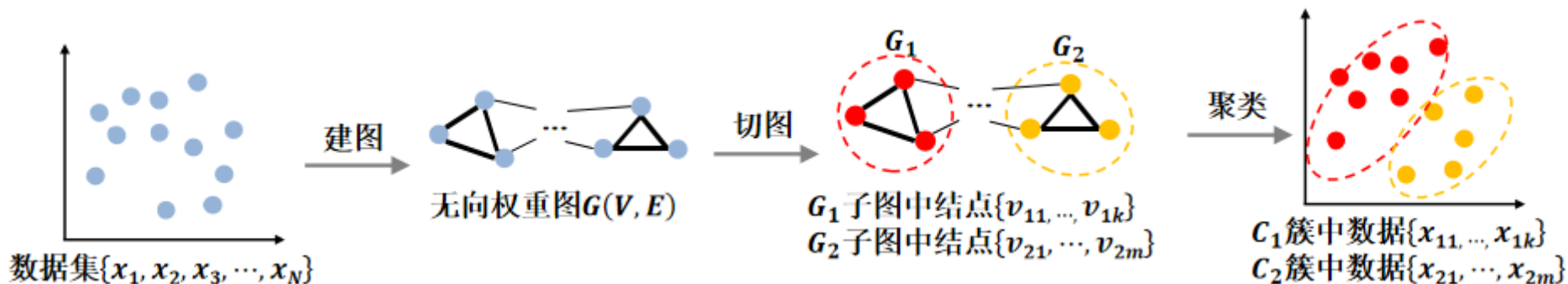
谱聚类

- 谱聚类算法
- 无向图构建
- 无向图切图

谱聚类算法

● 谱聚类算法

- 定义：一种基于图论的聚类方法，适用于从全局视角解决高维数据的聚类任务；通过将数据集建模为图结构并将其切分为多个最优子图，把聚类问题转化为图的最优划分问题
- 基本思想：将数据点之间的相似度建模为图的权重，并对该无向权重图进行切图，使不同的子图间边权重和尽可能的低，而子图内的边权重和尽可能的高



谱聚类算法

● 谱聚类算法流程

输入：样本集 $D = \{x_1, x_2, \dots, x_N\}$ ，降维后维度 k_1 ，簇数量 k_2

过程：

步骤1：根据样本构建无向权重图，并计算邻接矩阵 W 及度矩阵 D ；

步骤2：计算拉普拉斯矩阵 $L = D - W$ ，并标准化得到 $L' = D^{-1/2} L D^{-1/2}$ ；

步骤3：计算 L' 最小的 k_1 个特征值所各自对应的特征向量 f ；

步骤4：将由特征向量 f 组成的矩阵按行标准化，构建 $N \times k_1$ 维特征矩阵 F ；

步骤5：将特征矩阵 F 的每一行视为一个 k_1 维的样本，共 N 个样本，并以 k_2 为簇数量对该样本集进行聚类；

步骤6：得到簇划分 $C = \{c_1, c_2, \dots, c_{k_2}\}$ 。

输出：簇划分 C

无向图构建

● 构建无向权重图

➤ 对于**无向权重图** $G(V, E)$, $V = \{v_1, v_2, \dots, v_N\}$ 为点集, 与 N 个数据样本一一对应; w_{ij} 为 v_i 和 v_j 之间的**权重**, 且 $0 \leq w_{ij} = w_{ji}$; 图中任意点 v_i 的**度** d_i 定义为与其相连的所有边的权重和, 即 $d_i = \sum_{j=1}^n w_{ij}$; **邻接矩阵** W 由图中任意两点之间的权重组成, 距离较远 (近) 的两点之间的边权重较低 (高), 构建邻接矩阵包含以下三种方式

■ ϵ -邻近法: 设置一个距离阈值 ϵ , 邻接矩阵 W 表示为

$$w_{ij} = w_{ji} = \begin{cases} 0 & s_{ij} > \epsilon \\ \epsilon & s_{ij} \leq \epsilon \end{cases}$$

$s_{ij} = \|v_i - v_j\|_2^2$ 为两点间的**欧式距离**

无向图构建

- 全连接法：使用核函数定义边权重，如多项式核函数、高斯核函数、Sigmoid核函数，邻接矩阵 W 表示为

$$w_{ij} = \exp\left(-\frac{\|v_i - v_j\|_2^2}{2\sigma^2}\right)$$

- K 邻近法：使用 K 邻近算法遍历所有样本点，只有任意两个样本点 v_i 和 v_j 互为距离最近的 K 个点时，权重 $w_{ij} > 0$ ，邻接矩阵 W 表示为

$$w_{ij} = w_{ji} = \begin{cases} 0 & , v_i \notin \text{KNN}(v_j) \vee v_j \notin \text{KNN}(v_i) \\ \exp\left(-\frac{\|v_i - v_j\|_2^2}{2\sigma^2}\right) & , v_i \in \text{KNN}(v_j) \wedge v_j \in \text{KNN}(v_i) \end{cases}$$

无向图切图

● 无向图切图

- 目标：将图 $G(V, E)$ 划分为 K 个**无交集**子图，使子图内的点权重和高，子图间的点权重和低；子图点集合为 $\{G_1, G_2, \dots, G_K\}$ ，满足 $G_i \cap G_j = \emptyset$ 且 $G_1 \cup G_2 \cup \dots \cup G_K = V$
- 对于 K 个子图点集合，定义**切图损失**为

$$cut(G_1, G_2, \dots, G_K) = \frac{1}{2} \sum_{k=1}^K W(G_k, \overline{G_k})$$

$\overline{G_k}$ 为 G_k 的补集

其中， $W(G_i, G_j)$ 为任意两个子图点集合 G_i 和 G_j 间的**切图权重**

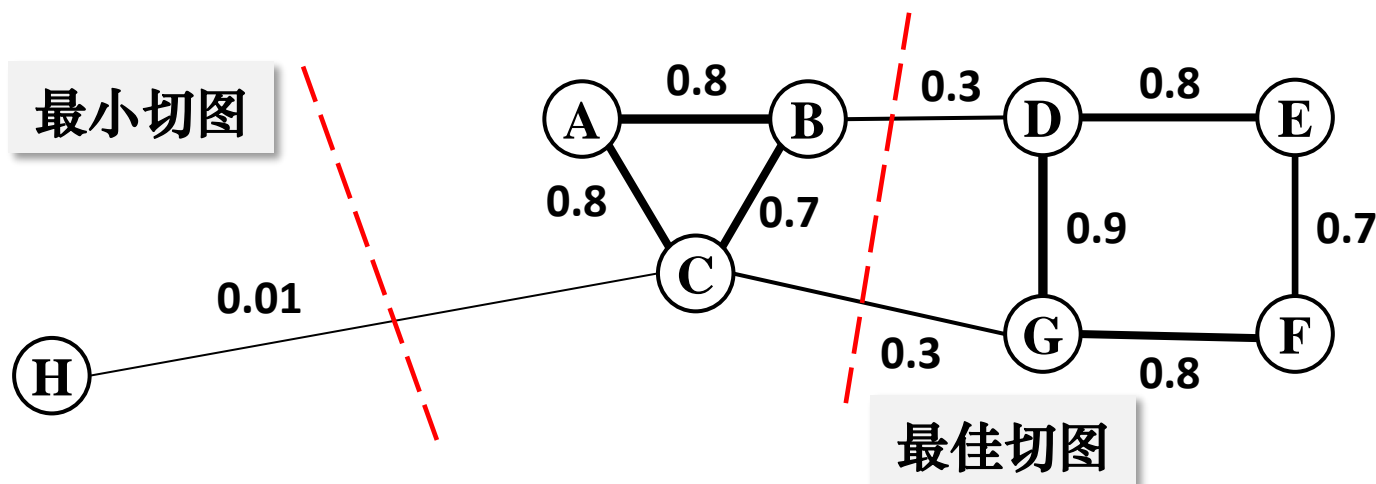
$$W(G_i, G_j) = \sum_{v_m \in G_i, v_n \in G_j} w_{mn}$$

无向图切图

● 最小切图算法 (Minimum Cut)

$$\min(\text{cut}(G_1, G_2, \dots, G_K)) = \min\left(\frac{1}{2} \sum_{k=1}^K W(G_k, \overline{G_k})\right)$$

➤ 最小切图的局限性：若选择一个连接边缘点的最小权重边进行切图，能够满足切图损失最小化，但各子图内点集合分布不均匀，且无法保证各子图内的点权重和高



无向图切图

● 标准化切图算法 (Normalized Cut)

➤ 基本思想：不仅最小化切图损失，同时最大化各子图内点的权重和

$$NCut(G_1, G_2, \dots, G_K) = \frac{1}{2} \sum_{k=1}^K \frac{W(G_k, \overline{G_k})}{vol(G_k)}$$

➤ 引入指示向量 $h_k \in \{h_1, h_2, \dots, h_K\}$ ，对于 N 维 (N 为样本数) 向量 h_k ，其定义为

$$h_{nk} = \begin{cases} 0 & , v_n \notin G_k \\ \frac{1}{\sqrt{vol(G_k)}} & , v_n \in G_k \end{cases}$$

$vol(G_k) = \sum_{m \in G_k} d_m$ 为子图 G_k 内点的度之和

则有

$$NCut(G_1, G_2, \dots, G_K) = \sum_{k=1}^K \frac{cut(G_k, \overline{G_k})}{vol(G_k)} = \sum_{k=1}^K h_k^T L h_k = \sum_{k=1}^K (H^T L H)_{kk} = tr(H^T L H)$$

拉普拉斯矩阵 $L = D - W$ ，用于进行特征值分解，从而实现聚类

无向图切图

● 标准化切图算法

➤ 然而找到满足此优化目标的 H 是一个NP难问题，考虑令矩阵

$H = D^{-1/2}F$ ，则切图损失更新为

$$\begin{aligned} NCut(G_1, G_2, \dots, G_K) &= tr \left(F^T D^{-1/2} L D^{-1/2} F \right) \\ &= tr \left(F^T L' F \right), s.t. F^T F = I \end{aligned}$$

其中， D 为**度矩阵**，是一个对角矩阵，对角元素 d_{ii} 为点 v_i 的度； $L' = D^{-1/2} L D^{-1/2}$ 为**标准化拉普拉斯矩阵**，即 $L'_{ij} = L_{ij} / \sqrt{d_i \times d_j}$

➤ 对于 $tr \left(F^T L' F \right)$ ，目标是找到 L' 最小的 K ($K \ll N$) 个特征值，即进行维度规约，将维度从 N 降到 K ，从而近似解决NP难问题；将 K 个特征值对应的特征向量 f 组成一个 $N \times K$ 维特征矩阵 F ，并对每一行进行一次传统聚类算法，得到最终聚类结果

讨论

● 谱聚类 v.s. K-Means

K-Means		谱聚类
适用场景	分布较为均匀、形状接近球形（凸集）的数据，大规模的数据集。	更高维的任意形状数据，较小、中等规模的数据集。
优点	结果易于解释；计算效率高。	对初始条件不敏感；对噪声点和异常值较为鲁棒；能够对全局信息进行建模。
缺点	需要预先指定聚类簇的数量；聚类结果依赖于初始中心点的选择；容易受到噪声点的影响。	计算复杂度较高。

本章总结

- 本章聚类算法总结
- 其他方法在聚类中的应用

本章算法总结

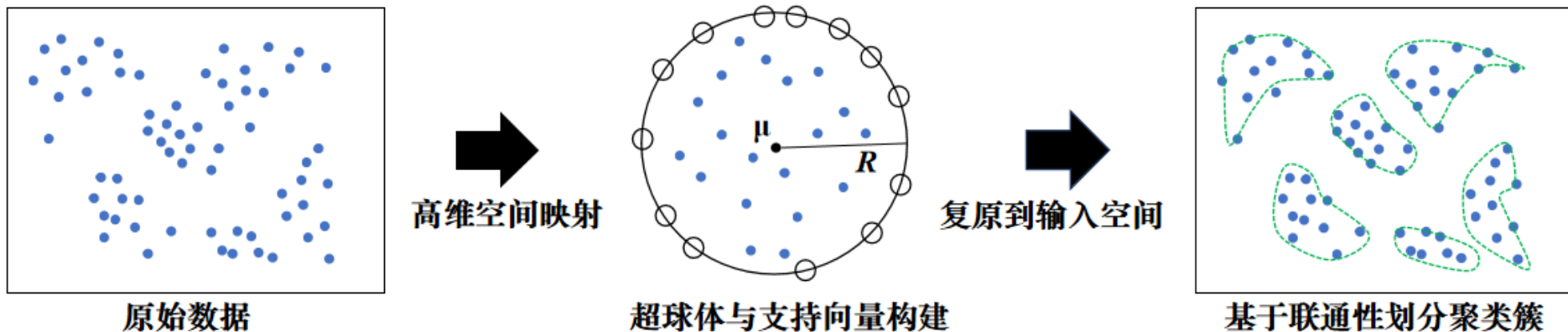
● 本章小结

方法	类别	是否可以处理高维数据	非线性	异常数据抗干扰性	运行效率	超参敏感性
K均值聚类	基于划分的聚类方法	可以处理高维数据	无法处理非线性数据	对噪声与异常点敏感	高	敏感
高斯混合聚类	基于划分的聚类方法	可以处理高维数据	可以处理非线性数据	对噪声与异常点敏感	低	敏感
AGNES	基于层次的聚类方法	难以处理高维数据	难以处理非线性数据	对噪声与异常点敏感	低	不敏感
DBSCAN	基于密度的聚类方法	难以处理高维数据	可以处理非线性数据	可以处理噪声和异常点	高	敏感
谱聚类	基于模型的聚类方法	可以处理高维数据	可以处理非线性数据	可以处理噪声和异常点	低	敏感

其他方法在聚类中的应用

● 支持向量聚类

- 在特征空间中寻找能包围所有训练集的最小超球体，超球体表面的点为支持向量
- 将该超球体逆向映射回输入空间，得到数据分布边界的轮廓
- 在输入空间中基于联通性划分簇

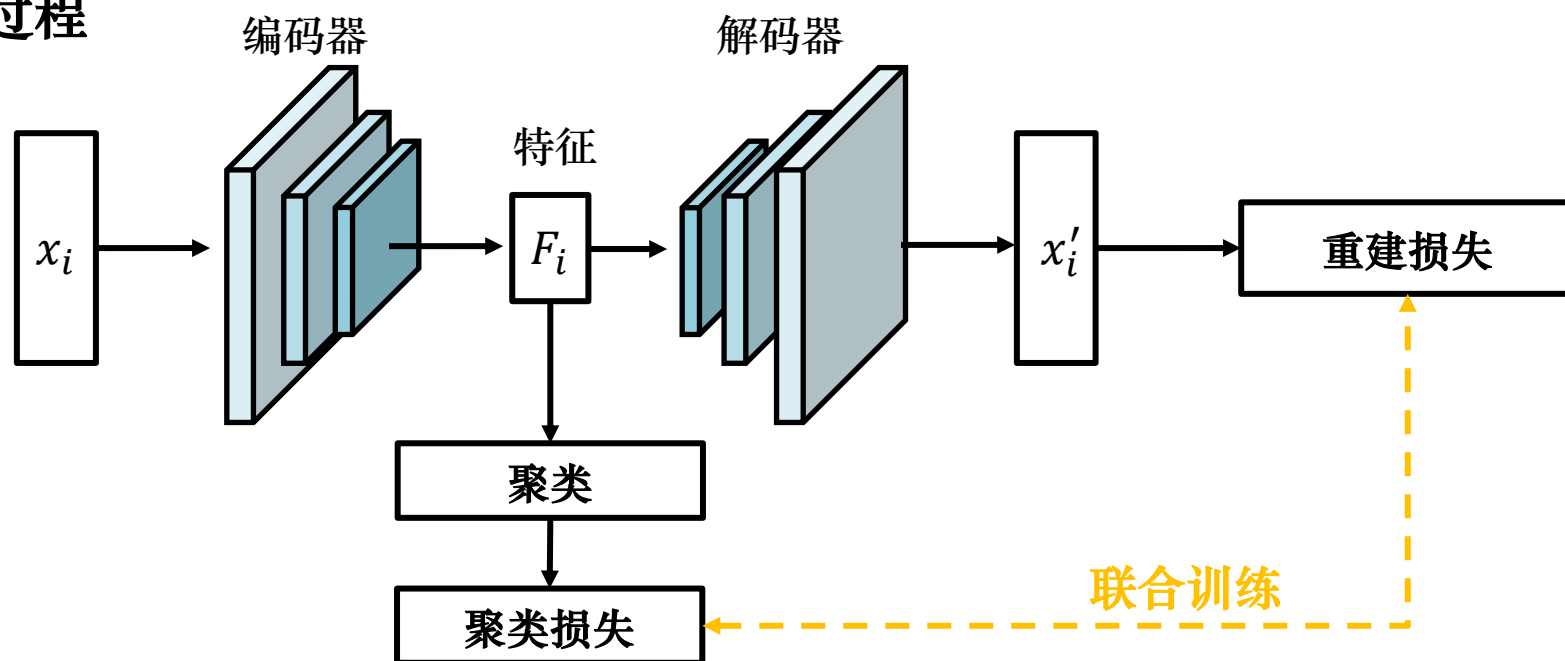


支持向量聚类可以较好地处理非线性数据和高维数据，同时仅需支持向量描述聚类轮廓，计算效率高。本例可以看出，监督学习模型在无监督任务中也可以得到应用

其他方法在聚类中的应用

● 深度学习在聚类中的应用

- 利用深度模型，如自编码器，从数据中提取高层次特征
- 深度学习聚类算法能够进行端到端的训练，同时优化特征编码和特征聚类过程



基于深度学习的聚类算法可以更好地处理非线性和高维数据

机器学习范式

监督学习

无监督学习

半监督学习

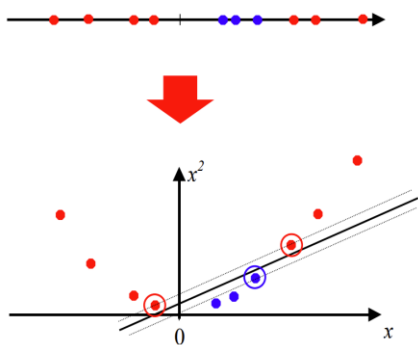
定义

使用有标签样本进行学习，再对无标签样本进行测试

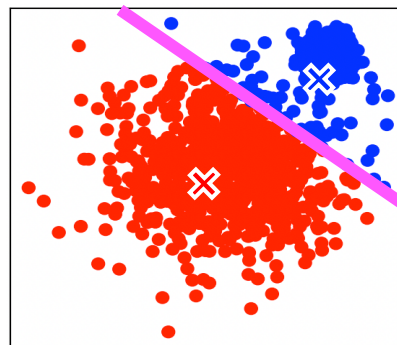
完全使用无标签样本进行学习，基于样本间的相似度，对样本进行类别归纳

利用大量无标签样本辅助有标签样本建立一个更好的学习器

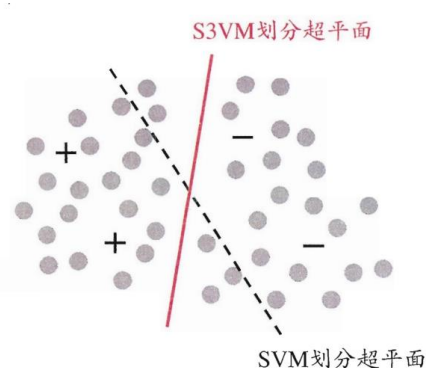
典型方法



支持向量机



聚类



半监督支持向量机