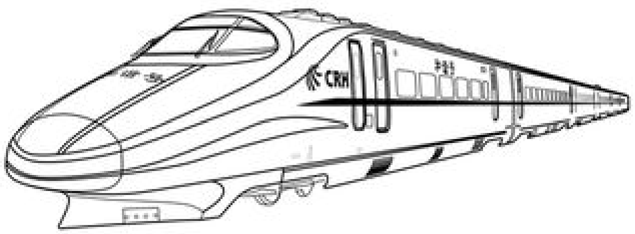
北京交通大学
BEIJING JIAOTONG UNIVERSITY

智能信息技术教育中心
The Education Center of Intelligence Information Technologies

人工智能基础

机器学习

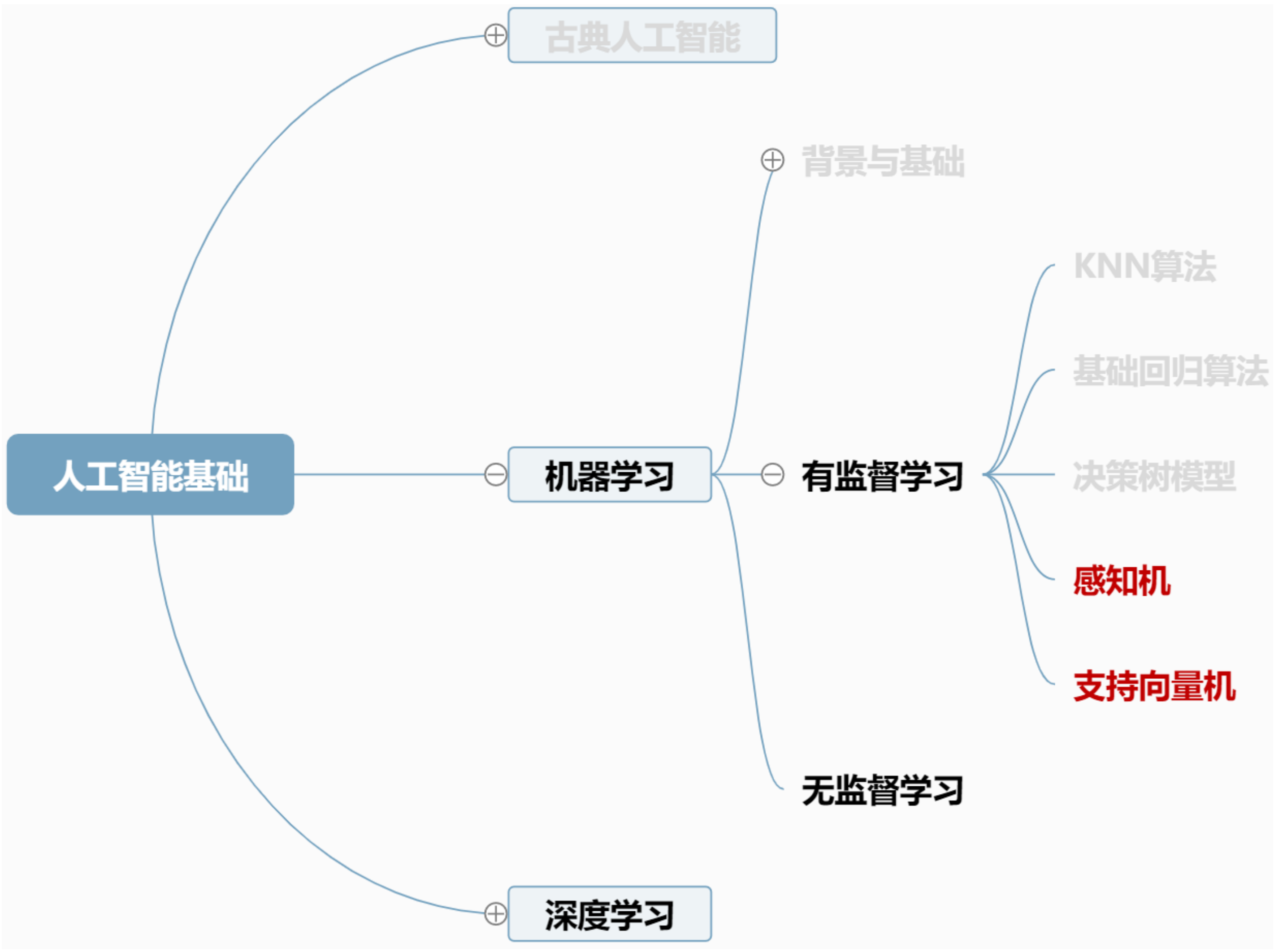
耿阳李敖
2022年3月



内容概要

- **内容回顾**
- K-means 聚类算法
- 其它聚类算法泛讲
- 总结

课程总览

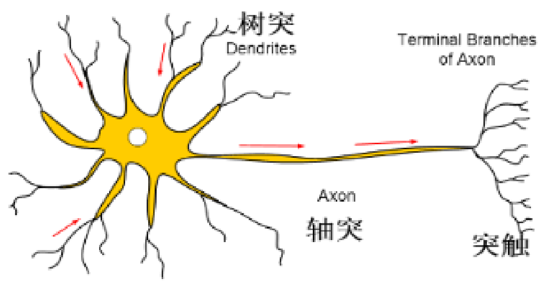


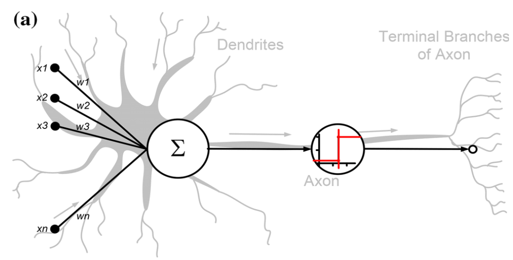
古典人工智能
背景与基础
KNN算法
基础回归算法
人工智能基础
机器学习
有监督学习
决策树模型
感知机
支持向量机
无监督学习
深度学习

感知机

基本思想

- 仿生原理：模拟人类神经细胞的工作原理



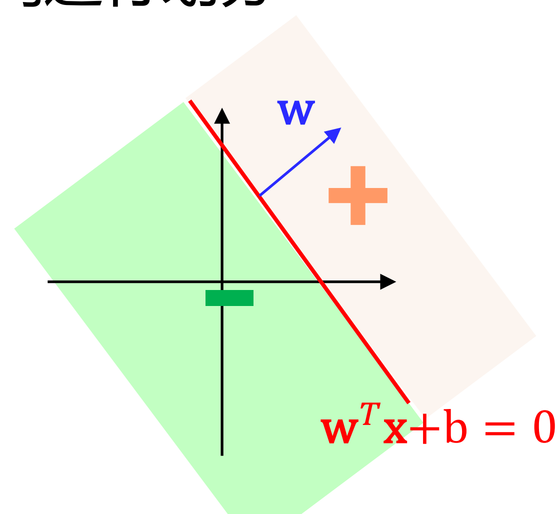


- 几何意义：用一个线性超平面对样本空间进行划分

$$y = \sigma\left(\sum_{i=1}^{d} w_i x_{(i)} + b\right) = \sigma(\mathbf{w}^T\mathbf{x} + b)$$

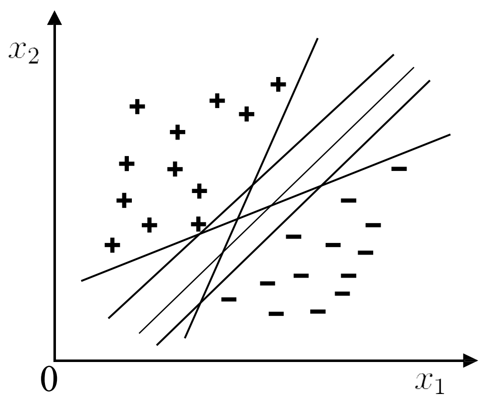
$$\mathbf{w} = (w_1, w_2, \ldots, w_D)^T \in \mathbb{R}^D$$

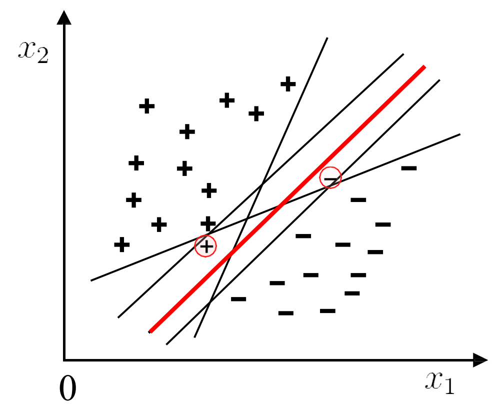
$$\mathbf{x} = \left(x_{(1)}, x_{(2)}, \ldots, x_{(D)}\right)^T \in \mathbb{R}^D$$



支持向量机

对感知机的改进



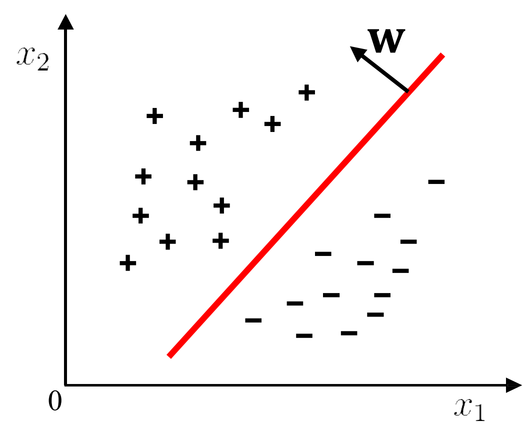


哪个分类超平面效果最好?

应选择“正中间”，容忍性好,鲁棒性高

分离间隔最大化原则

$$\max_{\mathbf{w},b}\min_{i} \frac{y_i(\mathbf{w}^T\mathbf{x}_i+b)}{\|\mathbf{w}\|}$$



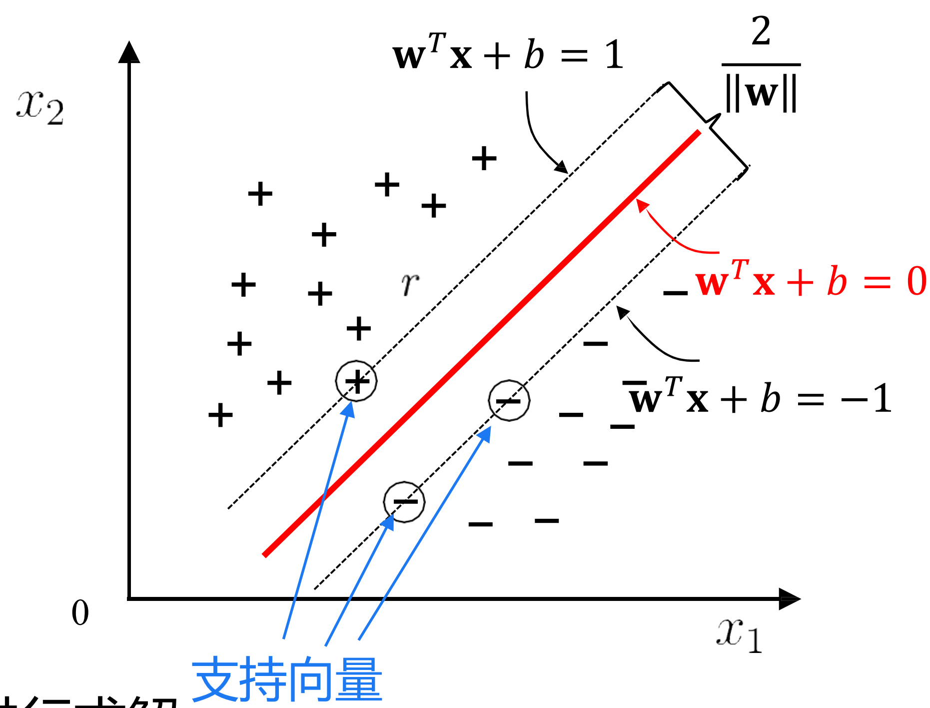
支持向量机

问题转化

$$\max_{\mathbf{w},b}\min_{i}\frac{y_i(\mathbf{w}^T\mathbf{x}_i+b)}{\|\mathbf{w}\|}$$

$$\min_{\mathbf{w},b}\frac{\|\mathbf{w}\|^2}{2}$$

$$\text{s.t.}\quad y_i(\mathbf{w}^T\mathbf{x}_i+b)\geq 1\ \ (i=1,2,\dots,N)$$



- 利用拉格朗日乘子转化为对偶问题进行求解

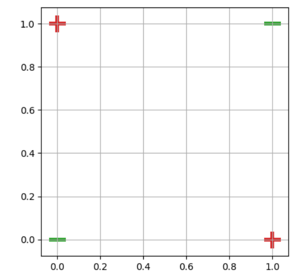
$$\max_{\boldsymbol{\alpha}}\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i y_i\alpha_j y_j\mathbf{x}_i^T\mathbf{x}_j$$

$$\text{s.t.}\quad\sum_{i=1}^{N}\alpha_i y_i=0,\ \alpha_i\geq 0\ (i=1,\dots N)$$

支持向量机

线性不可分挑战

- 异或分类问题

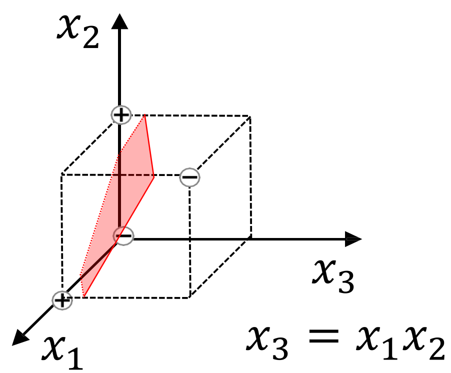


$$\text{正类}(y=+1): \quad \{(0,1),(1,0)\}$$

$$\text{负类}(y=-1): \quad \{(0,0),(1,1)\}$$

- 线性可分的判定：正负类别样本的凸包是否存在交集

核方法



- 基本思想

通过构造特征变到高维空间从而线性可分

- 引入特征映射$\phi(\mathbf{x})$升维

一般维度越高，线性可分性越好，希望能够映射至无穷维

$$\phi(\mathbf{x}) \in \mathbb{R}^{\infty} \qquad \mathbf{w}^T\phi(\mathbf{x}) + b = 0$$

支持向量机

核方法

- 对偶表示

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$$

$$\text{s.t.} \quad y_i(\mathbf{w}^T\phi(\mathbf{x}_i)+b) \geq 1$$

$$\Longleftrightarrow$$

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i y_i \alpha_j y_j \boxed{\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \alpha_i \geq 0 \ (i = 1, \ldots N)$$

- 核函数

$$\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1,\mathbf{x}_1) & \kappa(\mathbf{x}_1,\mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_N,\mathbf{x}_1) \\ \kappa(\mathbf{x}_2,\mathbf{x}_1) & \kappa(\mathbf{x}_2,\mathbf{x}_2) & & \kappa(\mathbf{x}_N,\mathbf{x}_2) \\ \vdots & & \ddots & \vdots \\ \kappa(\mathbf{x}_N,\mathbf{x}_1) & \kappa(\mathbf{x}_N,\mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_N,\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{N\times N}$$

κ被称为核函数，寻找合适的核函数往往比直接寻找ϕ更加方便

支持向量机

核方法

- 常用核函数

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\boldsymbol{x}_i^\top\boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i^\top\boldsymbol{x}_j)^d$	$d\geq 1$为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\lVert\boldsymbol{x}_i-\boldsymbol{x}_j\rVert^2}{2\delta^2}\right)$	$\delta>0$为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\lVert\boldsymbol{x}_i-\boldsymbol{x}_j\rVert}{\delta}\right)$	$\delta>0$
Sigmoid核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\tanh(\beta\boldsymbol{x}_i^\top\boldsymbol{x}_j+\theta)$	tanh为双曲正切函数，$\beta>0$, $\theta<0$

无穷维

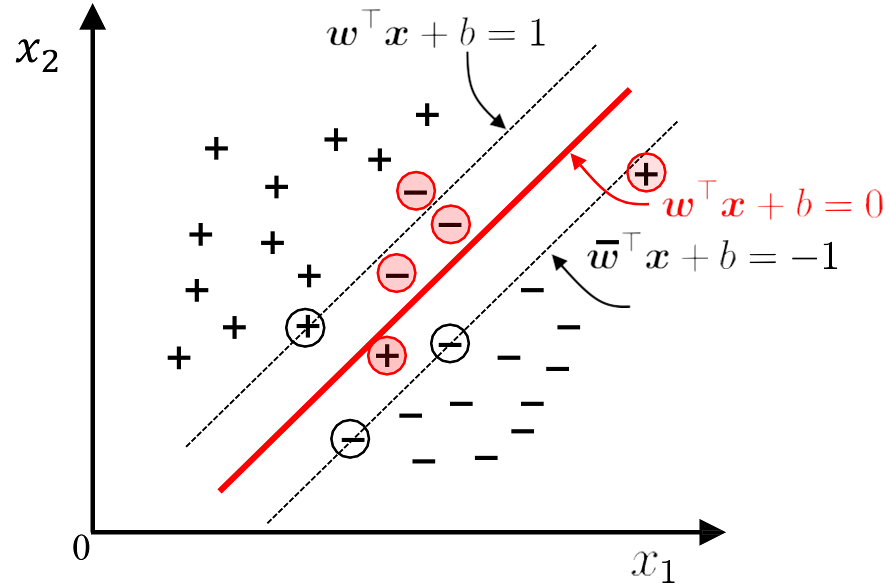
- 选择适合的核函数取决于具体任务，需要用后验方式进行验证

支持向量机

容错策略

- 软间隔法
  - 现实中很难保证数据线性可分; 即便找到某个核函数使其线性可分，也很难断定是否是由过拟合造成的。

红色：不满足约束的样本



  - 引入”软间隔”的概念, 允许支持向量机在一些样本上不满足约束。

支持向量机

容错策略

- 软间隔法

$$\min_{\mathbf{w},b} \ \frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{s.t.} \ \ y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

SVM原始优化问题

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \ \frac{1}{2}\|\mathbf{w}\|^2 + \lambda \sum_{i=1}^{N} \xi_i$$
$$\text{s.t.} \ \ y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0 \quad (i = 1, \ldots, N)$$

ξ_i为第i个训练样本的容错间隔

多分类问题

- 1 vs Others

对于每个类别训练一个SVM，将当前类别当做正类，其他类别当做负类

支持向量机

支持向量回归

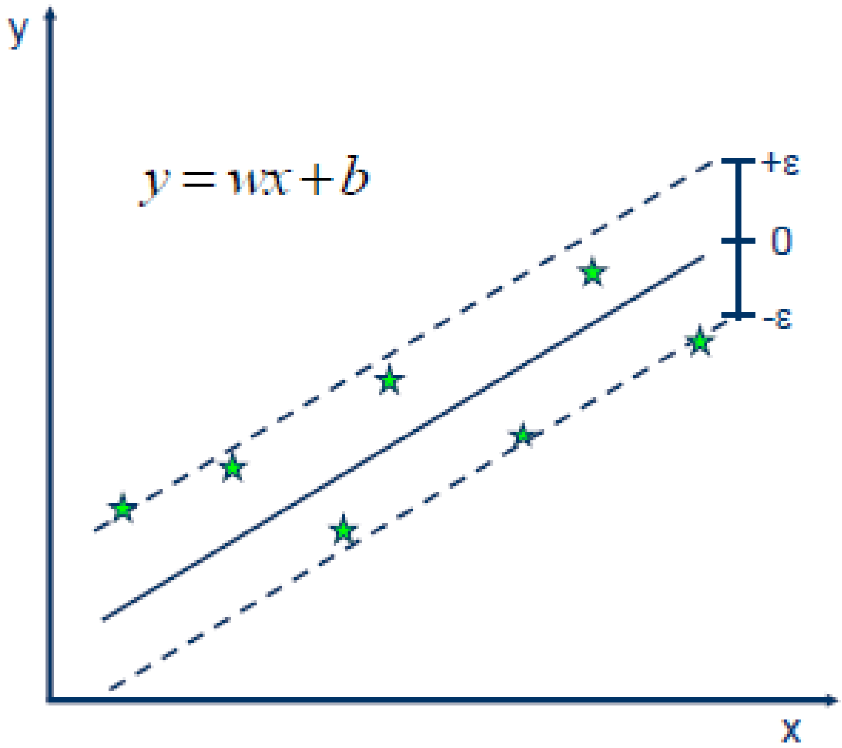
- 距离超平面不超过 ϵ

$$\min_{\mathbf{w},b} \ \frac{1}{2}\|\mathbf{w}\|^2$$

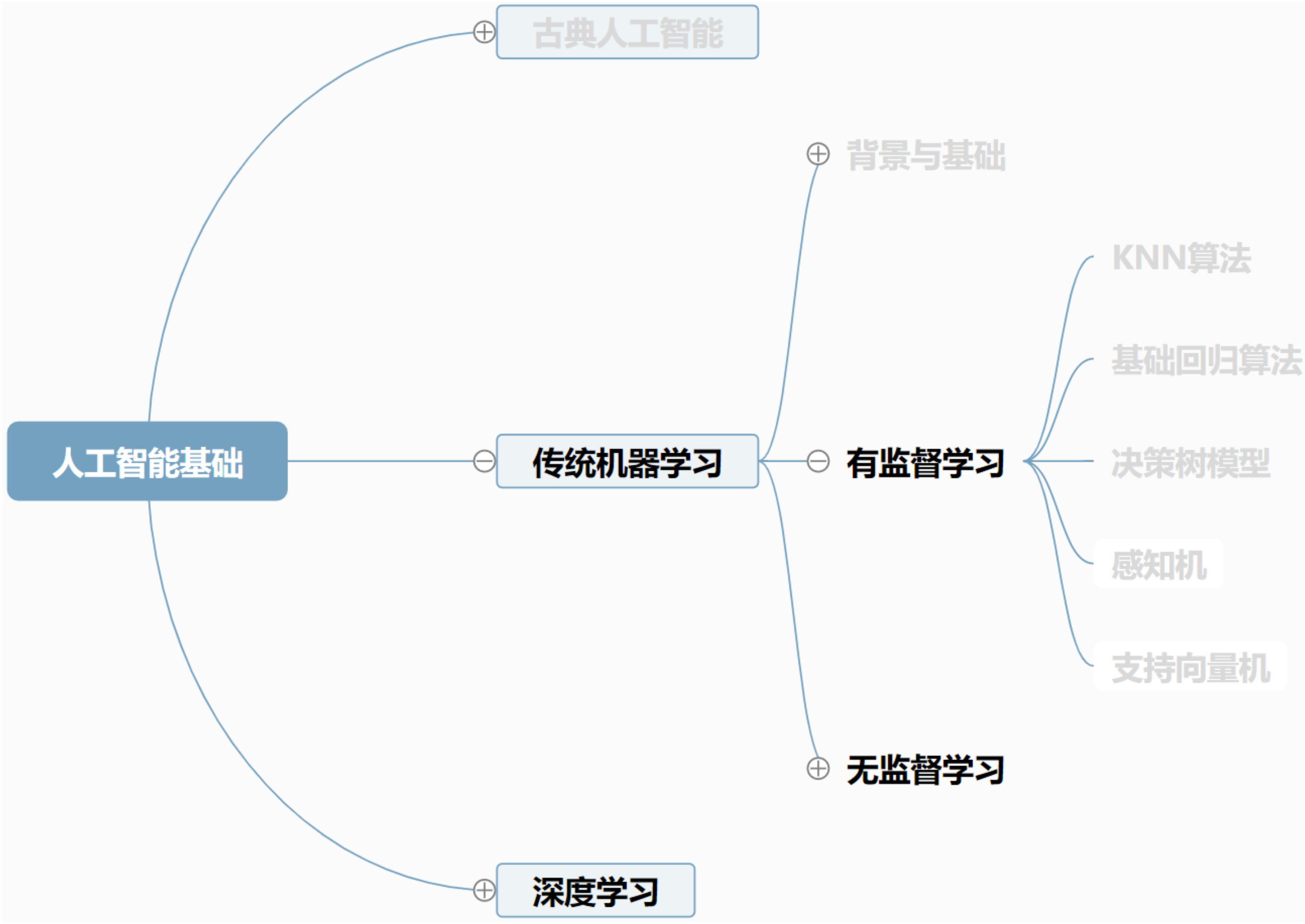
$$\text{s.t.} \ \ |\mathbf{w}^T\mathbf{x}_i + b - y_i| \leq \epsilon$$

- 本质上是使用了特殊损失函数的线性回归

- 核方法与软间隔法同样适用

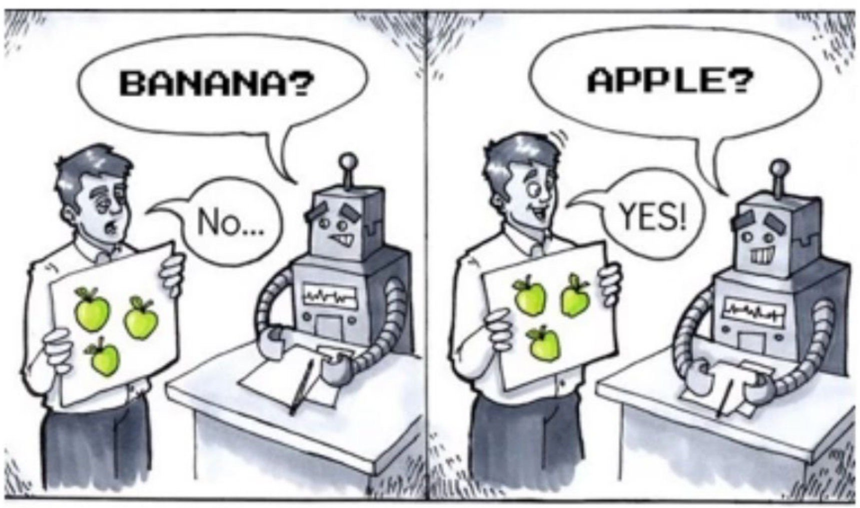


阶段性小节


人工智能基础
古典人工智能
传统机器学习
背景与基础
有监督学习
KNN算法
基础回归算法
决策树模型
感知机
支持向量机
无监督学习
深度学习

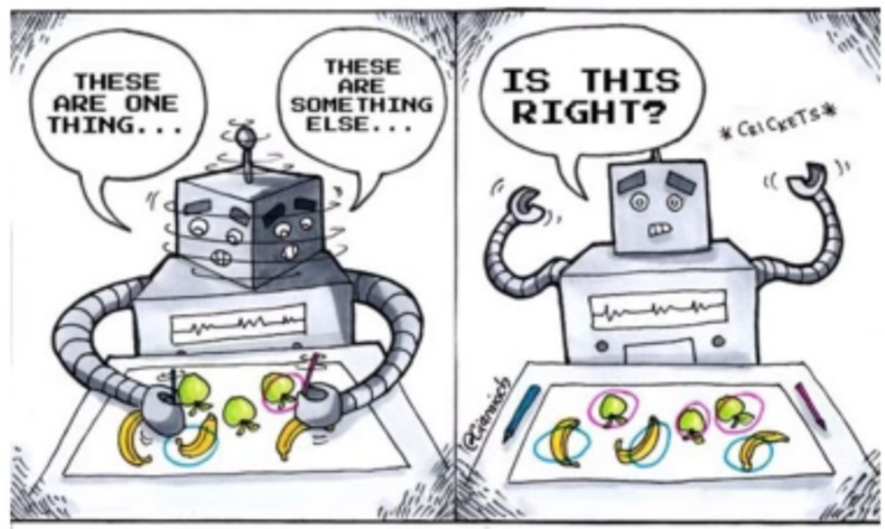
无监督学习概述

有监督学习



Supervised Learning

无监督学习



Unsupervised Learning

无监督学习概述

有监督学习

- 训练集中每个样本的目标或类别事先已知
- 学习映样本对之间的射关系
- 常见任务：分类、回归

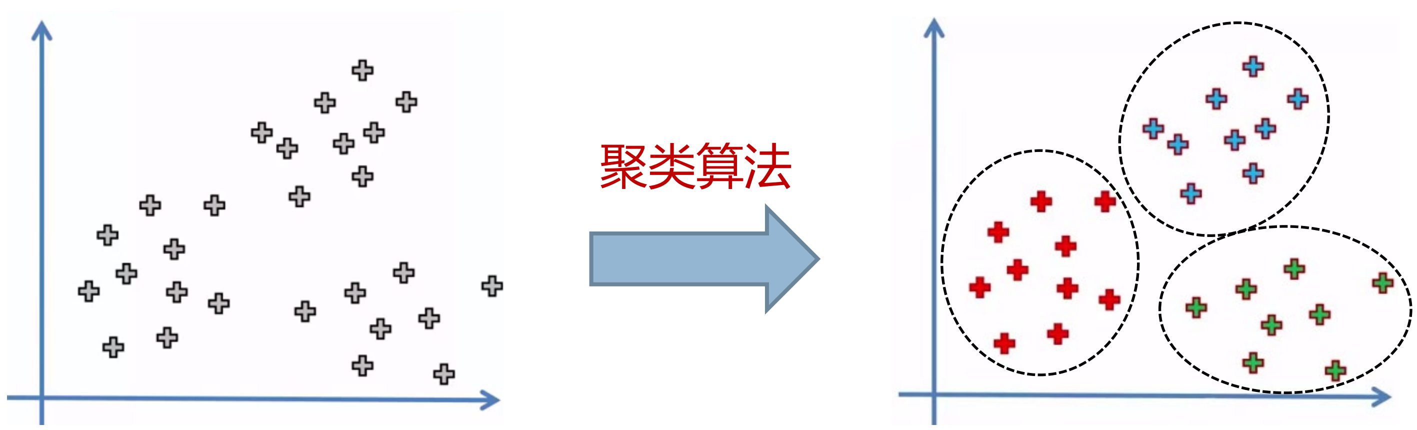
无监督学习

- 训练集中样本的目标及类别未知
- 给定一组样本，发现其内在的属性
- 常见任务：聚类、特征降维、概率分布估计

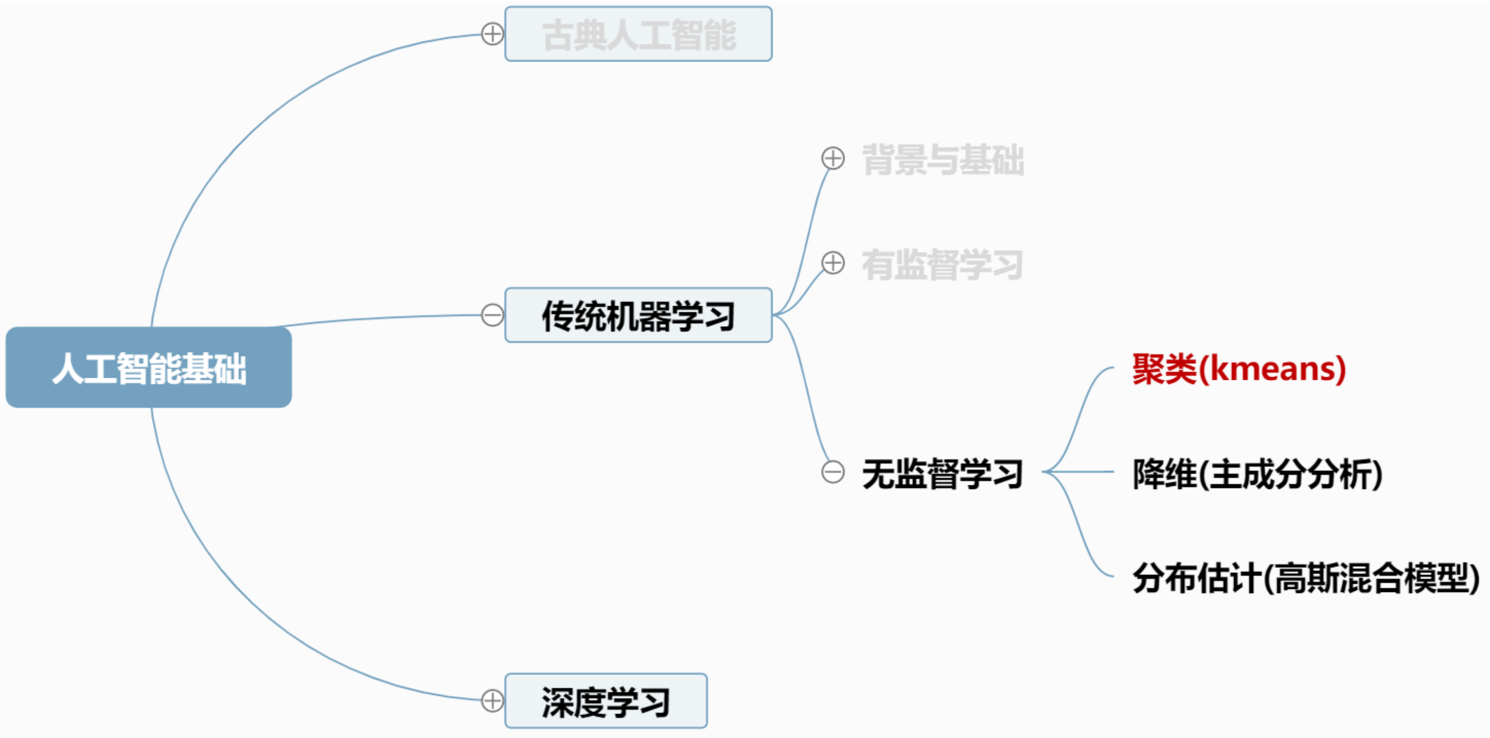
聚类算法

聚类算法

- 在“无监督学习”任务中研究最多、应用最广
- 聚类目标：将数据集中的样本划分为若干个通常不相交的子集（“簇”，cluster）。
- 聚类既可以作为一个单独过程（用于找寻数据内在的分布结构），也可作为分类等其他学习任务的前置过程



本节概况


人工智能基础
古典人工智能
传统机器学习
背景与基础
有监督学习
无监督学习
聚类(kmeans)
降维(主成分分析)
分布估计(高斯混合模型)
深度学习

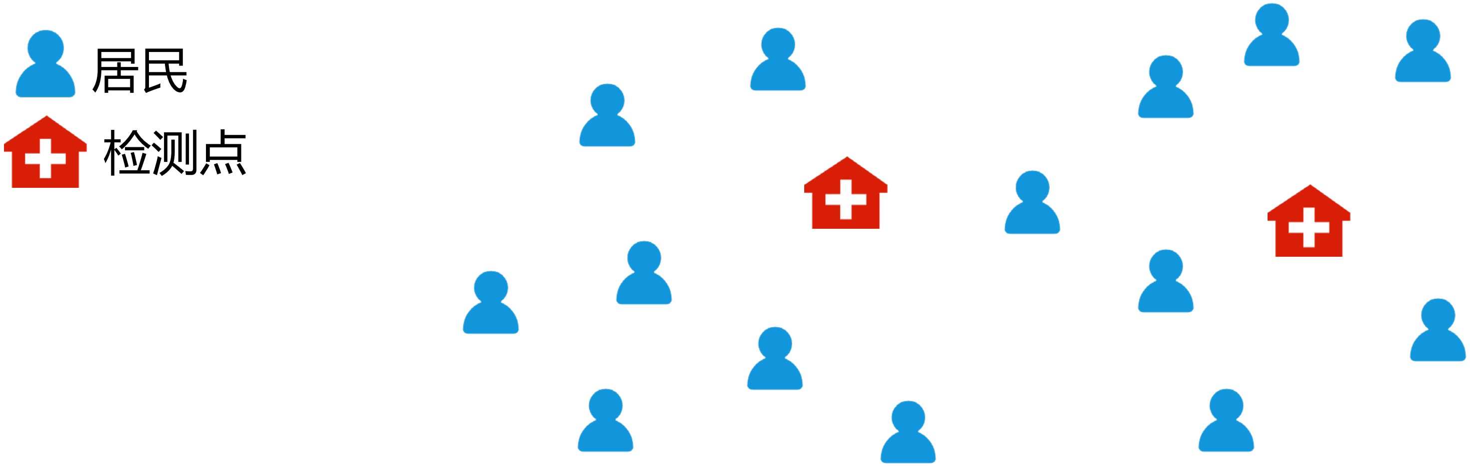
内容概要

- 内容回顾
- **K-means 聚类算法**
- 其它聚类算法泛讲
- 总结

K-means聚类

例子：核酸检测点选址

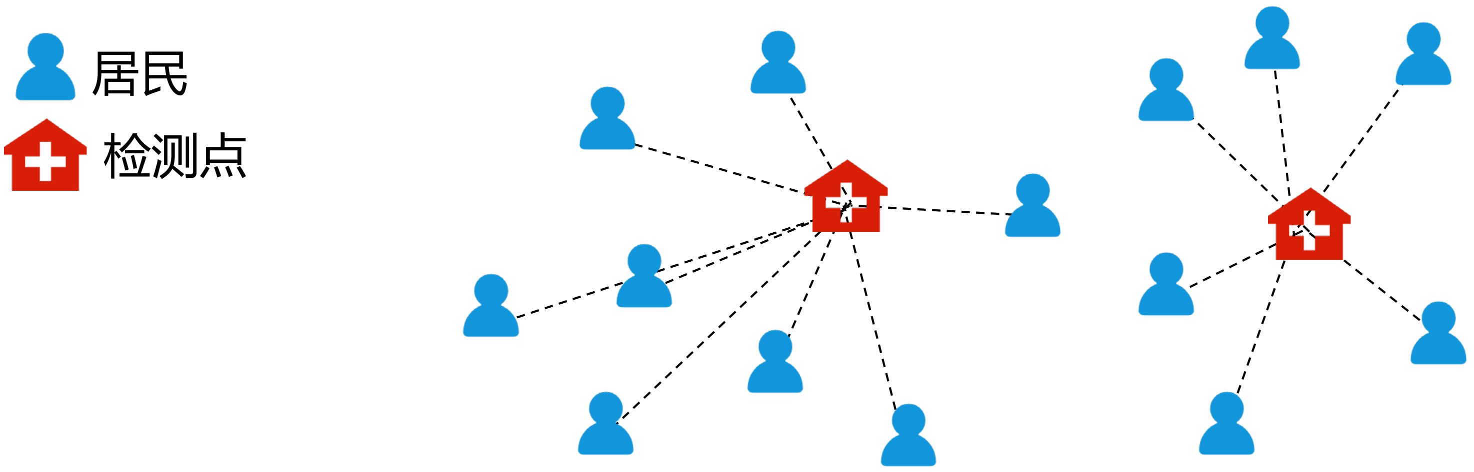
- 随机设置初始检测点



K-means聚类

例子：核酸检测点选址

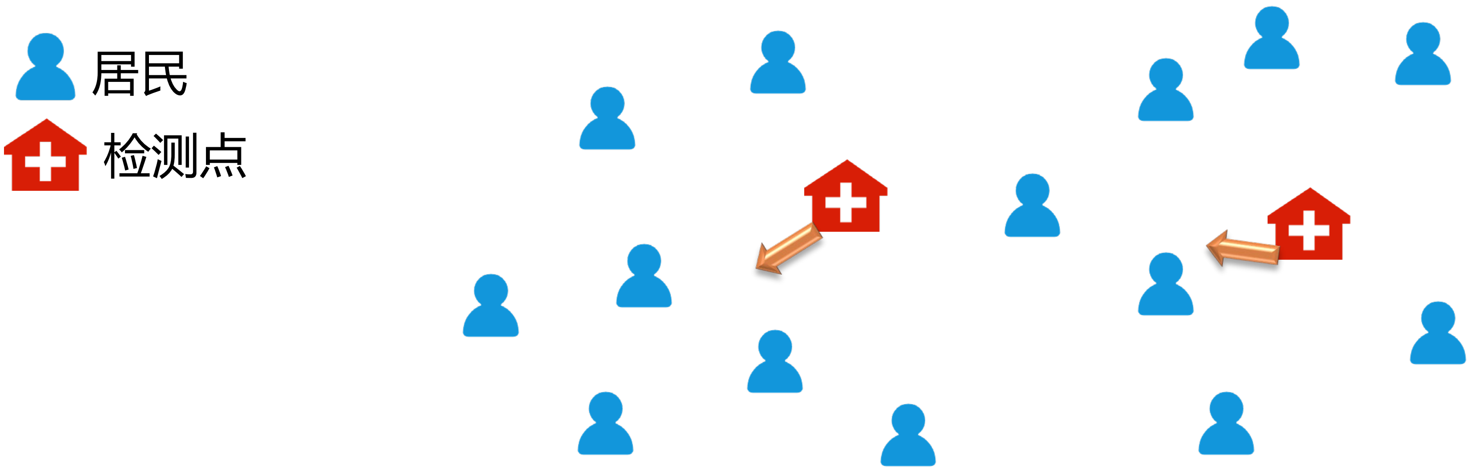
- 随机设置初始检测点
- 待检测人选择距离自己最近的检测点



K-means聚类

例子：核酸检测点选址

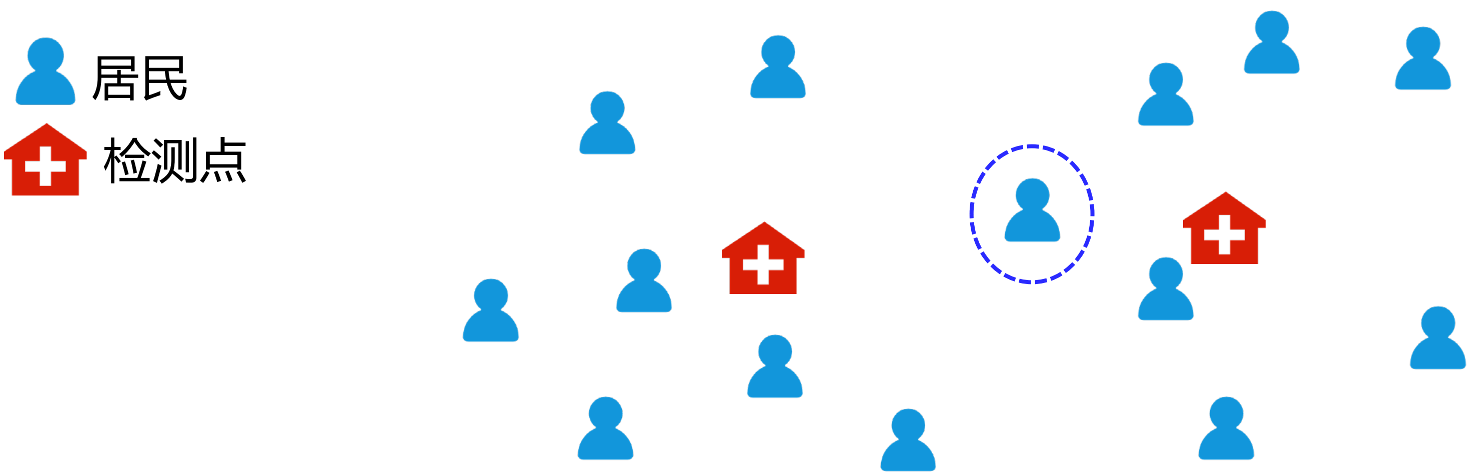
- 随机设置初始检测点
- 待检测人选择距离自己最近的检测点
- 为了方便群众，政府调整检测点的位置



K-means聚类

例子：核酸检测点选址

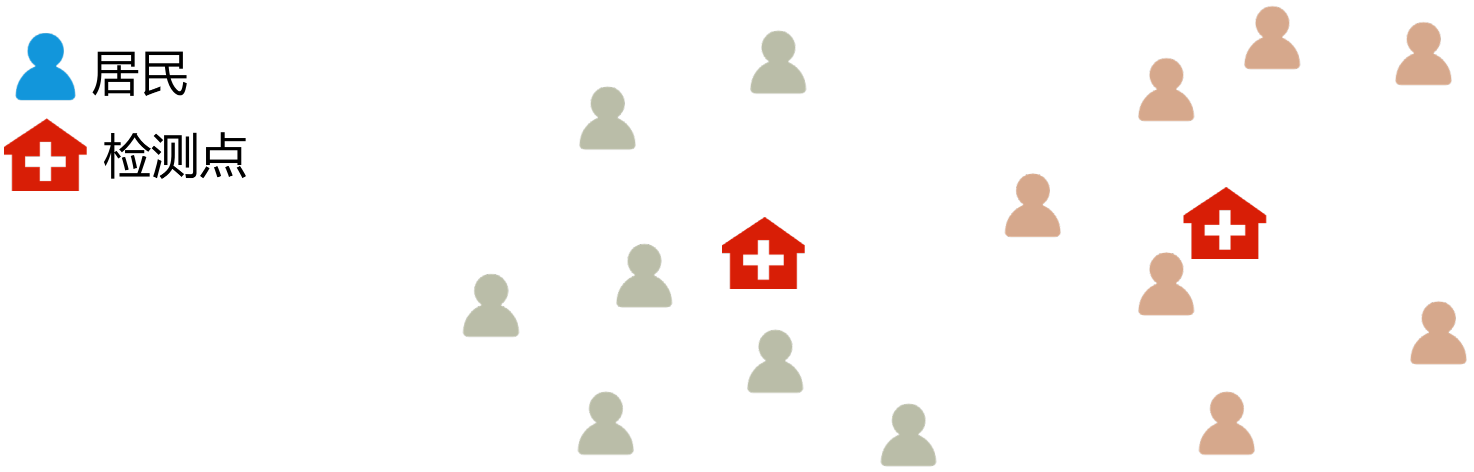
- 随机设置初始检测点
- 待检测人选择距离自己最近的检测点
- 为了方便群众，政府调整检测点的位置



K-means聚类

例子：核酸检测点选址

- 随机设置初始检测点
- 待检测人选择距离自己最近的检测点
- 为了方便群众，政府调整检测点的位置



K-means聚类

主要思想

- 给定N个D维数据点 $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$以及想要划分的簇数 K
- 找到K个中心点$\{\mathbf{c}_1, \dots, \mathbf{c}_K\}$，把$N$个数据点按照距离最小原则分配给每个中心点，使得每个数据点与分配中心点距离**平方**和最小

$$\min_{\{\mathbf{c}_1,\dots,\mathbf{c}_K\}} \sum_{i=1}^{N} \min_{k\in\{1,\dots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

$$\|\mathbf{x}_i - \mathbf{c}_k\|_p = \left(\sum_{j=1}^{D} (x_{ij} - c_{kj})^p\right)^{1/p}$$

- N个数据点中分配给同一个中心点的归属为一个簇

$$y_i = \arg\min_{k\in\{1,\dots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

K-means聚类

问题求解

- 本身是一个NP-难问题，全局最优难以求解

- 退而求其次，找到一个局部最优

- 方法：交替优化

$$\min_{\{\mathbf{c}_1,\ldots,\mathbf{c}_K\}} \sum_{i=1}^{N} \boxed{\min_{k\in\{1,\ldots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2}$$

- 固定$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$，求解内层：Easy

$$y_i = \arg\min_{k\in\{1,\ldots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

K-means聚类

问题求解

- 固定 $\{y_1, y_2, \ldots, y_N\}$，求解外层：

$$\min_{\{\mathbf{c}_1,\ldots,\mathbf{c}_K\}} \sum_{i=1}^{N} \min_{k\in\{1,\ldots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2 \quad \Rightarrow \quad \min_{\{\mathbf{c}_1,\ldots,\mathbf{c}_K\}} \sum_{i=1}^{N} \left\|\mathbf{x}_i - \mathbf{c}_{y_i}\right\|_2^2$$

- 注意到 $\mathbf{c}_k$ 只与 $\{\mathbf{x}_i \mid y_i = k\}$有关，可以对 $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ 分开独立求解

$$\min_{\mathbf{c}_k} \sum_{y_i=k} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

- 无约束凸优化问题：

$$L(\mathbf{c}_k) = \sum_{y_i=k} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

令 $\nabla_{\mathbf{c}_k} L = \mathbf{0}$ 可得最优

K-means聚类

问题求解

- $\nabla_{\mathbf{c}_k} L$ 的表达式

$$\nabla_{\mathbf{c}_k} L = \sum_{y_i=k} (\ ?\ )$$

Tips

$$L(\mathbf{c}_k) = \sum_{y_i=k} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

$$\mathbf{x}_i, \mathbf{c}_k, \nabla_{\mathbf{c}_k} L \in \mathbb{R}^D$$

$$\|\mathbf{x}_i - \mathbf{c}_k\|_2^2 = \mathbf{x}_i{}^T\mathbf{x}_i + \mathbf{c}_k{}^T\mathbf{c}_k - 2\mathbf{x}_i{}^T\mathbf{c}_k$$

A	B	C	D
$2\,\mathbf{x}_i$	$\mathbf{c}_k$	$\mathbf{c}_k - 2\mathbf{x}_i$	$2\mathbf{c}_k - 2\mathbf{x}_i$

K-means聚类

问题求解

- $\nabla_{\mathbf{c}_k} L$ 的表达式

$$\nabla_{\mathbf{c}_k} L = \sum_{y_i=k} \nabla_{\mathbf{c}_k} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

$$= \sum_{y_i=k} \nabla_{\mathbf{c}_k} (\mathbf{x}_i{}^T \mathbf{x}_i + \mathbf{c}_k{}^T \mathbf{c}_k - 2\mathbf{x}_i{}^T \mathbf{c}_k)$$

$$= \sum_{y_i=k} (\mathbf{0} + \nabla_{\mathbf{c}_k} \mathbf{c}_k{}^T \mathbf{c}_k - \nabla_{\mathbf{c}_k} 2\mathbf{x}_i{}^T \mathbf{c}_k)$$

$$= \sum_{y_i=k} (\mathbf{0} + 2\mathbf{c}_k - 2\mathbf{x}_i)$$

$$= \sum_{y_i=k} (2\mathbf{c}_k - 2\mathbf{x}_i)$$

Tips

$$L(\mathbf{c}_k) = \sum_{y_i=k} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$

$$\mathbf{x}_i, \mathbf{c}_k, \nabla_{\mathbf{c}_k} L \in \mathbb{R}^D$$

$$\|\mathbf{x}_i - \mathbf{c}_k\|_2^2 = \mathbf{x}_i{}^T \mathbf{x}_i + \mathbf{c}_k{}^T \mathbf{c}_k - 2\mathbf{x}_i{}^T \mathbf{c}_k$$

$$\mathbf{c}_k{}^T \mathbf{c}_k = \sum_{j=1}^{D} c_{kj}{}^2 \qquad \mathbf{x}_i{}^T \mathbf{c}_k = \sum_{j=1}^{D} x_{ij} c_{kj}$$

K-means聚类

问题求解

- $\nabla_{\mathbf{c}_k} L$ 的表达式

$$\nabla_{\mathbf{c}_k} L = \sum_{y_i=k} 2\mathbf{c}_k - 2\mathbf{x}_i = 0 \quad \Rightarrow \quad \sum_{y_i=k} \mathbf{c}_k = \sum_{y_i=k} \mathbf{x}_i$$

$$\mathbf{c}_k = \frac{1}{N_k} \sum_{y_i=k} \mathbf{x}_i$$

N_k：$\{\mathbf{x}_i \mid y_i = k\}$中数据点个数

$\{\mathbf{x}_i \mid y_i = k\}$中数据点的均值（mean）

K-means聚类

算法流程

- 输入：N 个 K 维数据点 $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$，聚类个数 K
- 随机设定K个中心点$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$
- 循环开始
  - 固定 $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ 对于所有数据点找到距离最近的中心点

$$y_i = \arg \min_{k \in \{1,\ldots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2 \qquad i = 1, \ldots, N$$

  - 固定 $\{y_1, \ldots, y_N\}$ 分别求 K 个簇的均值进行更新

$$\mathbf{c}_k \leftarrow \frac{1}{N_k} \sum_{y_i = k} \mathbf{x}_i \qquad k = 1, \ldots, K$$

  - 判断$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ 是否没有发生变化，如果是则跳出循环
- 返回$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$以及$\{y_1, \ldots, y_N\}$

K-means聚类

算法流程

- 输入：N 个 K 维数据点 $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$，聚类个数 K
- 随机设定K个中心点$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$
- 循环开始
  - 固定 $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ 对于所有数据点找到距离最近的中心点

$$y_i = \arg\min_{k\in\{1,\ldots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2 \quad i = 1, \ldots, N$$

$\mathcal{O}(NKD)$

  - 固定 $\{y_1, \ldots, y_N\}$ 分别求 K 个簇的均值进行更新

$$\mathbf{c}_k \leftarrow \frac{1}{N_k}\sum_{y_i=k} \mathbf{x}_i \quad k = 1, \ldots, K$$

$\mathcal{O}(ND)$

  - 判断$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ 是否没有发生变化，如果是则跳出循环
- 返回$\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$以及$\{y_1, \ldots, y_N\}$

总时间复杂度：$\mathcal{O}(LNKD)$

L 为迭代次数，通常比较小

K-means聚类

可视化演示

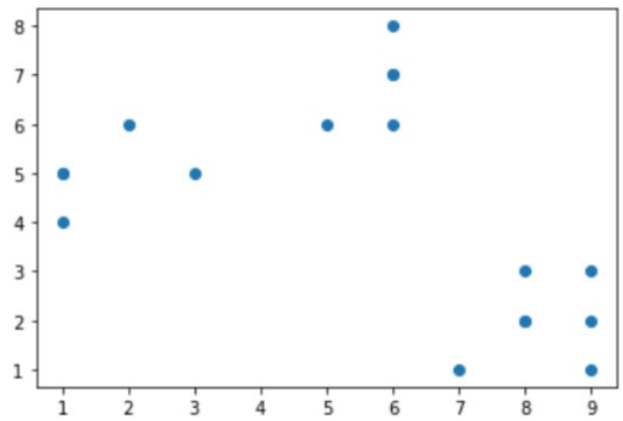
K-means聚类

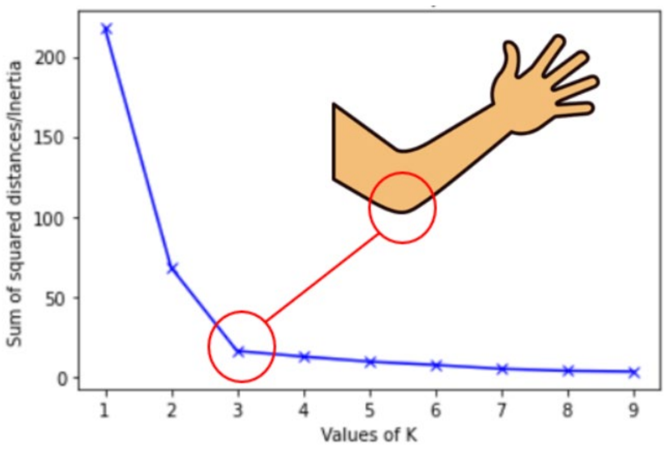
如何选择聚类簇数 K ？

- 肘方法（Elbow Method）

$$\min_{\{\mathbf{c}_1,\ldots,\mathbf{c}_K\}} \sum_{i=1}^{N} \min_{k\in\{1,\ldots,K\}} \|\mathbf{x}_i - \mathbf{c}_k\|_2^2$$ （K-means目标函数）

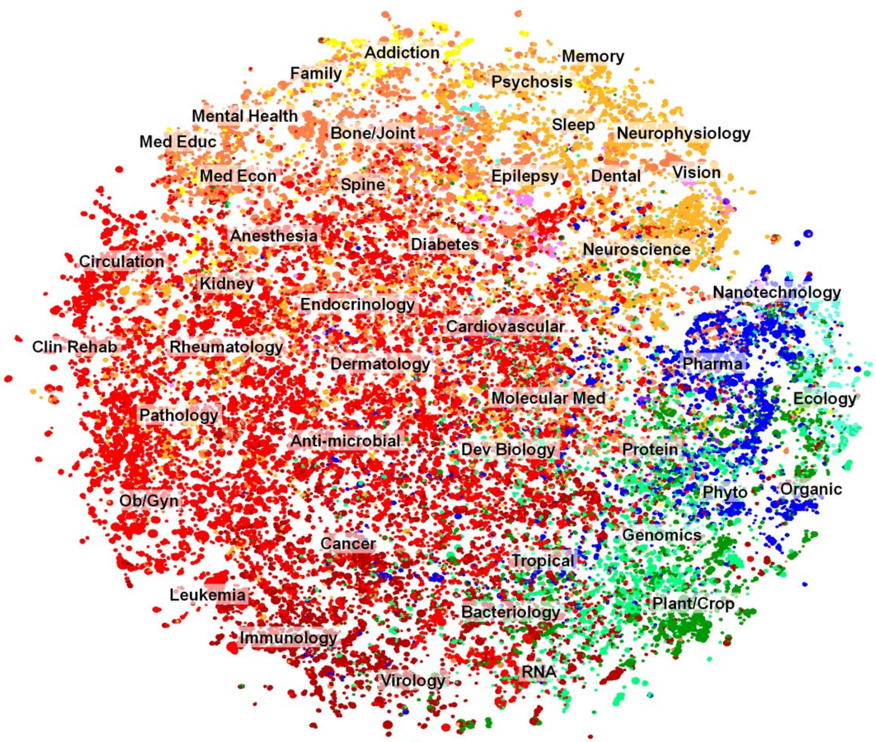
- 以K-means目标函数为纵坐标，以 K 为横坐标画一条曲线
- 曲线拐点（肘点）位置左右对应的 K 比较适合





K-means聚类

应用

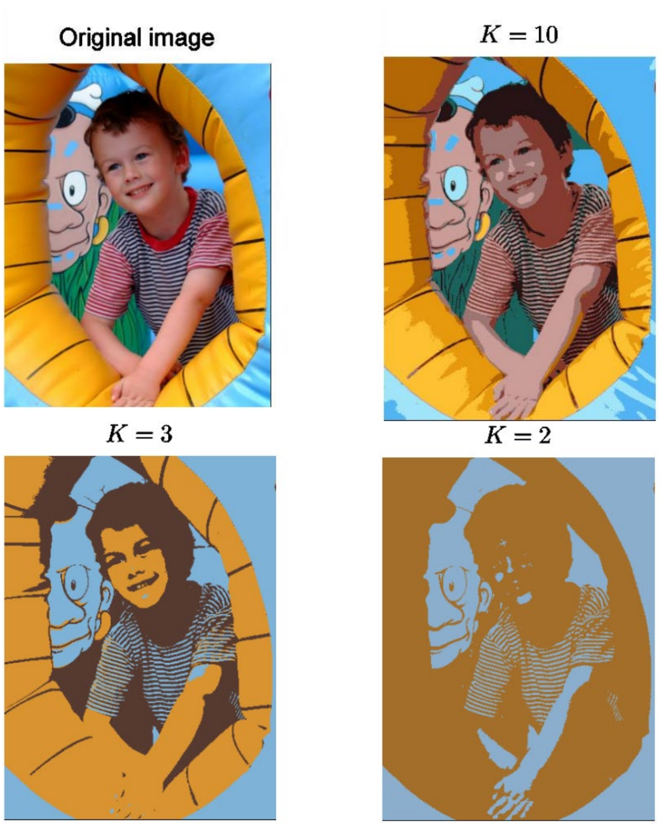


文本分类：根据论文内容判断其所属领域

社群发现：根据社交网络与用户画像发现可能的兴趣主题社群

K-means聚类

应用



像素量化：在颜色空间对像素值进行聚类

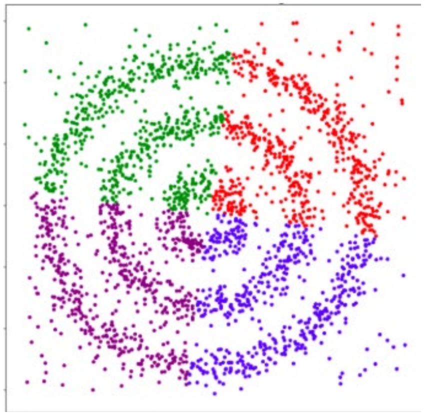
图像分割：生成超像素对图像进行预处理

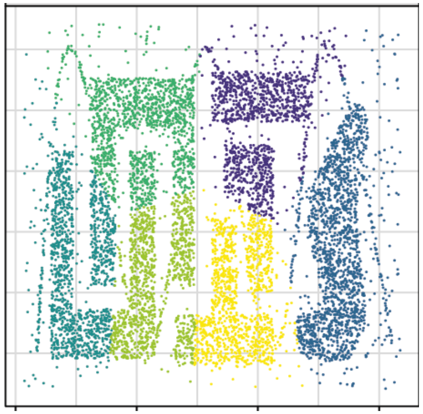
内容概要

- 内容回顾
- K-means 聚类算法
- **其它聚类算法泛讲**
- 总结

密度聚类

K-means 结果:

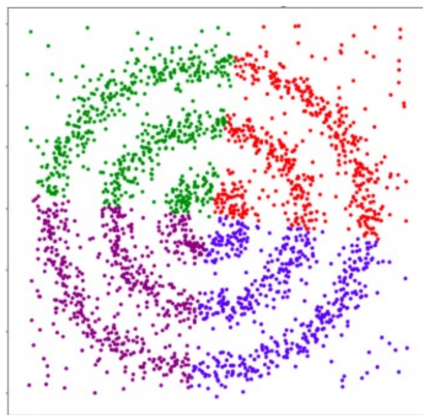


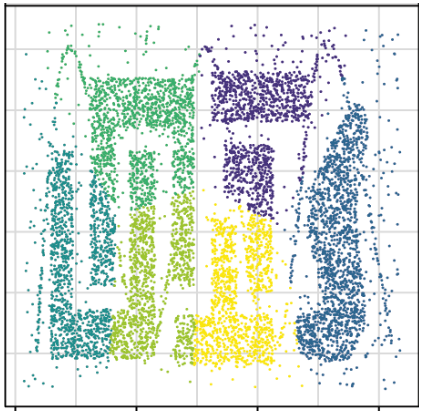


聚类泛讲

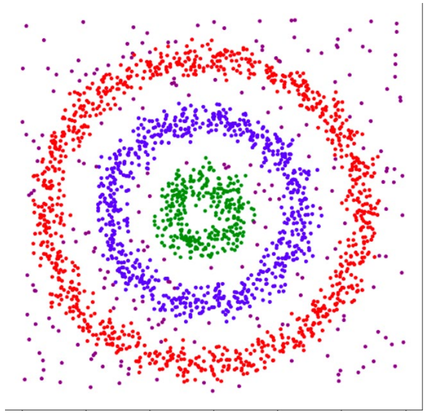
密度聚类

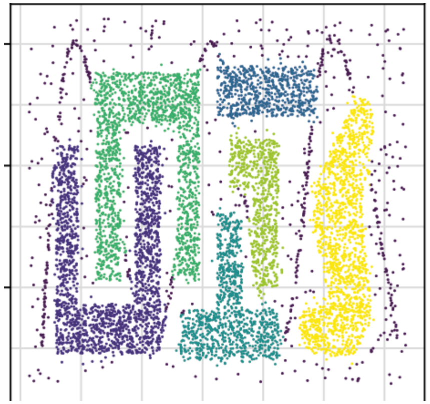
K-means 结果:





密度聚类结果:





密度聚类

- 核心思想：找出那些被低密度区域分割开的高密度区域作为簇
- DBSCAN算法流程
  - 密度估计：ε 邻域内点个数

$$\rho(\mathbf{x}) = |N_\varepsilon(\mathbf{x})| \qquad N_\varepsilon(\mathbf{x}) = \{\mathbf{x}_i | \mathbf{x}_i \in X, \mathrm{dist}(\mathbf{x}, \mathbf{x}_i) \le \varepsilon\}$$

  - 核心点发现：密度大于 MinPts

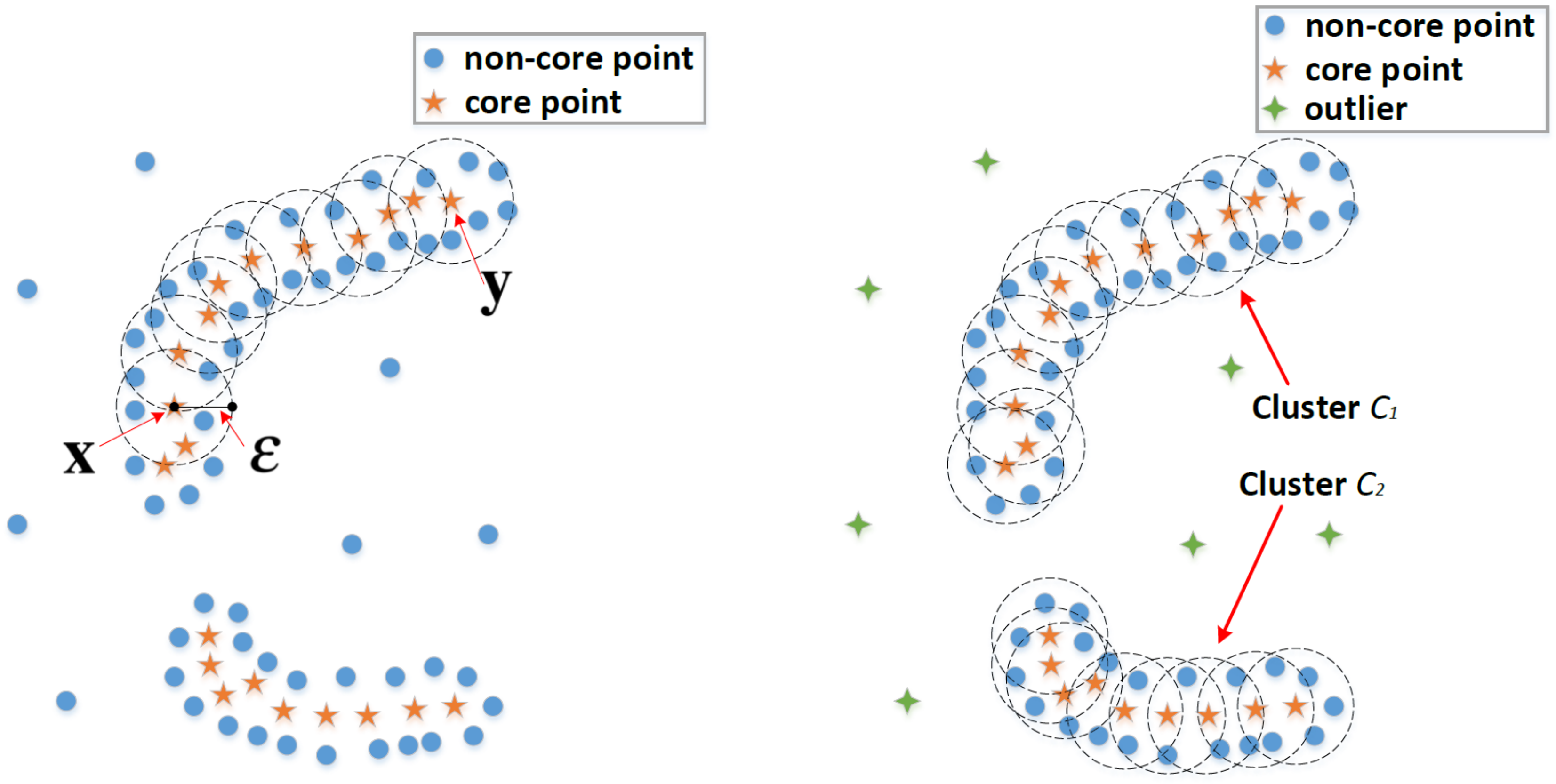
$$O \triangleq \{\mathbf{x} \in X | \rho(\mathbf{x}) \ge MinPts\}$$

  - 以核心点集为“骨架”，搜索向外生长，形成簇结构

$$C_i = N_\varepsilon(O_i) \triangleq \bigcup_{\mathbf{x} \in O_i} N_\varepsilon(\mathbf{x}) \qquad (i = 1, \dots, K)$$

聚类泛讲

密度聚类


non-core point
core point
x
ε
y
non-core point
core point
outlier
Cluster C_1
Cluster C_2

(a) (b)

密度聚类

Algorithm 1.7: DBSCAN

Input: dataset $X = \{\mathbf{x}_i\}_{i=1}^{n}$, ε, $MinPts$

Output: clusters $C_1, \ldots, C_k$, outliers set A

for $i = 1$ **to** n **do**

  find $N_\varepsilon(\mathbf{x}_i)$;

  $\rho(\mathbf{x}_i) = |N_\varepsilon(\mathbf{x}_i)|$;

define $O \triangleq \{\mathbf{x} \in X | \rho(\mathbf{x}) \geq MinPts\}$;

$k = 0$;

repeat

  $k = k + 1$;

  randomly select a core point $\mathbf{o}$ from O;

  use the depth-first-search algorithm to find the set $O_k \triangleq \{\mathbf{x} \in O | \mathbf{x}$ is connected to $\mathbf{o}\}$;

  define $C_k \triangleq N_\varepsilon(O_k)$;

  $O = O \backslash O_k$;

until $O = \emptyset$;

define $A \triangleq X \backslash \left(\bigcup_{i=1}^{k} C_i\right)$;

- 与 K-means 对比
  - 能够发现不同形状的簇结构
  - 能够自动确定簇个数
  - 密度估计在高维空间会失效
  - 时间复杂度为平方阶

聚类泛讲

层次聚类

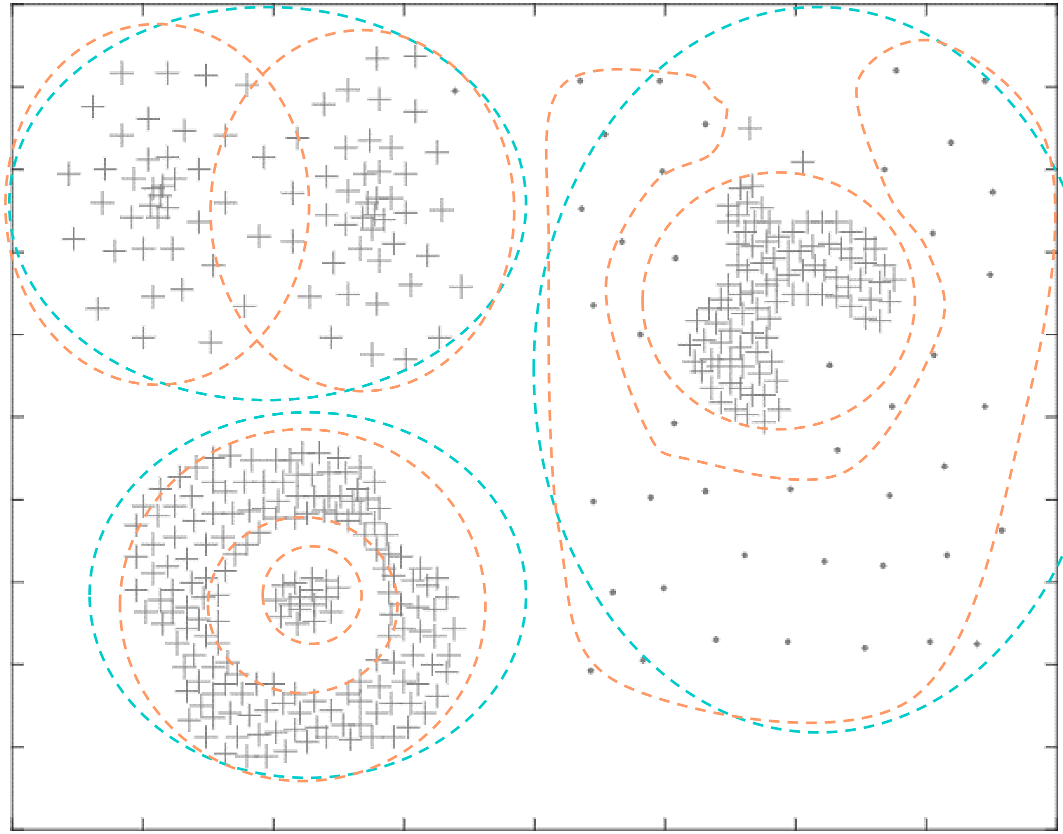
- 有几个簇?

A 3个

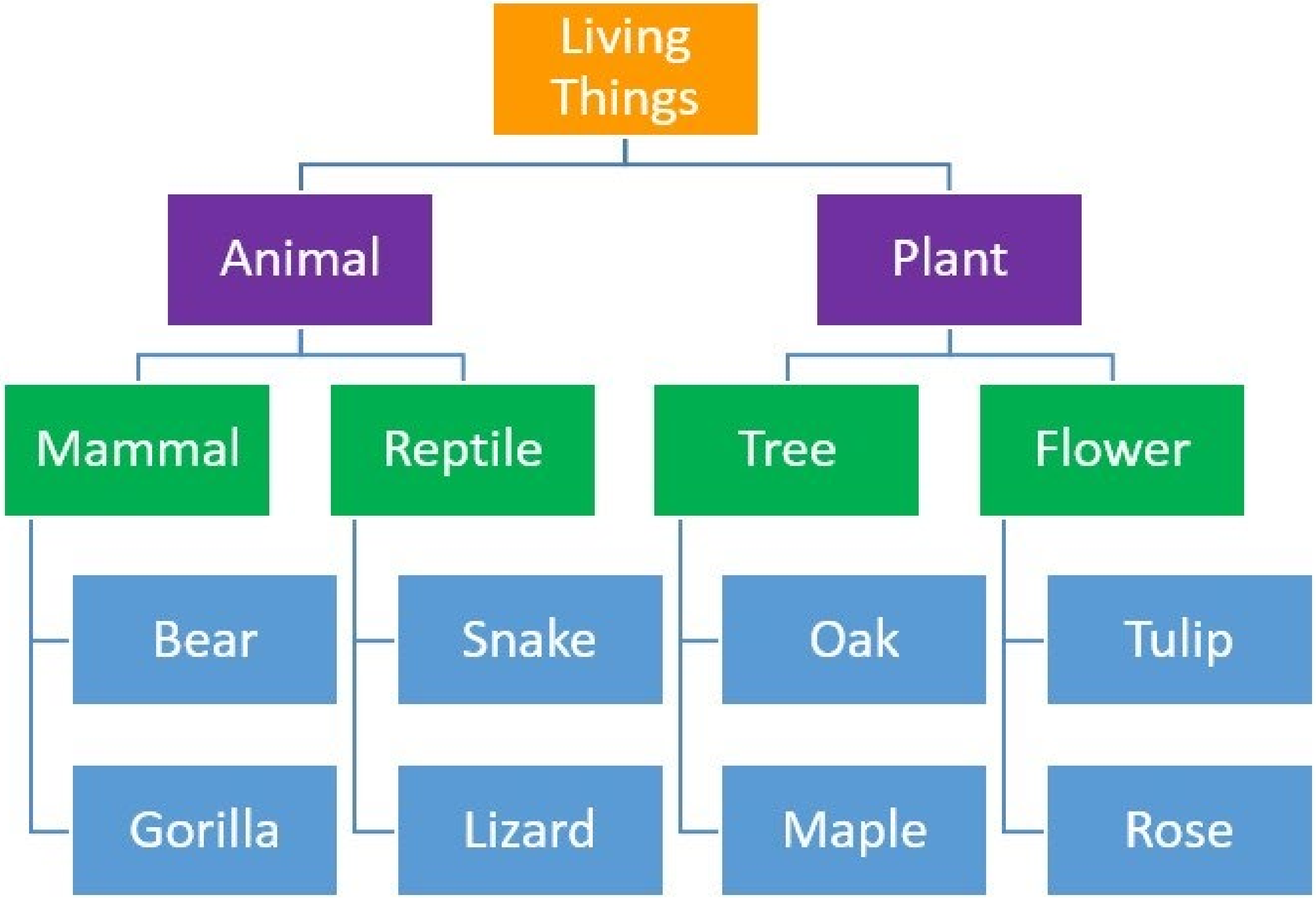
B 4个

C 5个

D 6个



层次聚类


Living Things
Animal
Plant
Mammal
Reptile
Tree
Flower
Bear
Gorilla
Snake
Lizard
Oak
Maple
Tulip
Rose

层次聚类

- 对数据集中的簇结构进行层级分析
- 包括两类
  - **自底向上**

    把数据集中的每个对象最为一个簇开始，迭代地把簇合并成为更大的簇，直到最终形成一个大簇，或者满足某个终止条件。基于自底向上算法有AGNES算法、BIRCH算法等。
  - **自顶向下**

    把所有数据点放至一个簇中开始，该簇是层次结构的根。然后，它把根上的簇划分为多个较小的子簇，并且递归地把这次簇划分成更小的簇，直到满足终止条件。常见的自顶向下的算法有DIANA算法。

层次聚类

- AGNES算法伪代码

AGNES算法的描述如右边所示，在第1-9行，算法先对仅含一个样本的初始聚类和相应的距离矩阵进行初始化；然后在第11-23行，AGNES不断合并距离最近的聚类簇，并对合并得到的聚类簇的距离矩阵进行更新；上述过程不断重复，直到达到预设的聚类簇数。

输入： 样本集$D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
聚类簇距离度量函数$d\in\{d_{\min},d_{\max},d_{\text{avg}}\}$;
聚类簇数k.

过程:

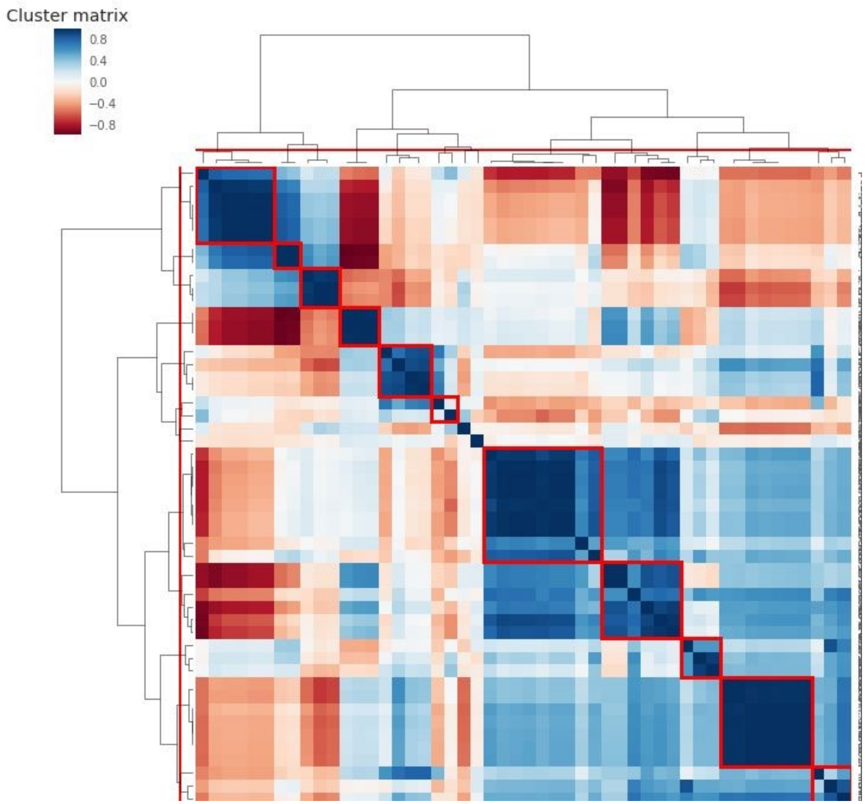
1: **for** $j=1,\ldots,m$ **do**
2: $\quad C_j=\{\boldsymbol{x}_j\}$
3: **end for**
4: **for** $i=1,\ldots,m$ **do**
5: $\quad$ **for** $j=i,\ldots,m$ **do**
6: $\quad\quad M(i,j)=d(C_i,C_j)$;
7: $\quad\quad M(j,i)=M(i,j)$
8: $\quad$ **end for**
9: **end for**
10: 设置当前聚类簇个数：$q=m$
11: **while** $q>k$ **do**
12: $\quad$ 找出距离最近的两个聚类簇(C_{i^*},C_{j^*});
13: $\quad$ 合并(C_{i^*},C_{j^*})：$C_{i^*}=C_{i^*}\bigcup C_{j^*}$;
14: $\quad$ **for** $j=j^*+1,\ldots,q$ **do**
15: $\quad\quad$ 将聚类簇C_j重编号为C_{j-1}
16: $\quad$ **end for**
17: $\quad$ 删除距离矩阵M的第j^*行与第j^*列;
18: $\quad$ **for** $j=1,\ldots,q-1$ **do**
19: $\quad\quad M(i^*,j)=d(C_{i^*},C_j)$;
20: $\quad\quad M(j,i^*)=M(i^*,j)$
21: $\quad$ **end for**
22: $\quad q=q-1$
23: **end while**
24: **return** 簇划分结果

输出： 簇划分$\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

聚类泛讲

层次聚类

- AGNES算法矩阵展示



内容概要

- 内容回顾
- K-means 聚类算法
- 其它聚类算法泛讲
- **总结**

总结

- 支持向量机
  - 核方法解决线性不可分问题
  - 容错机制、多分类问题、支持向量回归
- K-means聚类
  - 主要思想
  - 优化原理
  - 参数选择
  - 经典应用
- 其它聚类算法
  - 密度聚类
  - 层次聚类

北京交通大學
BEIJING JIAOTONG UNIVERSITY