大数据导论
Introduction to Big Data

聚类分析与离群点检测

叶允明
计算机科学与技术学院
哈尔滨工业大学（深圳）

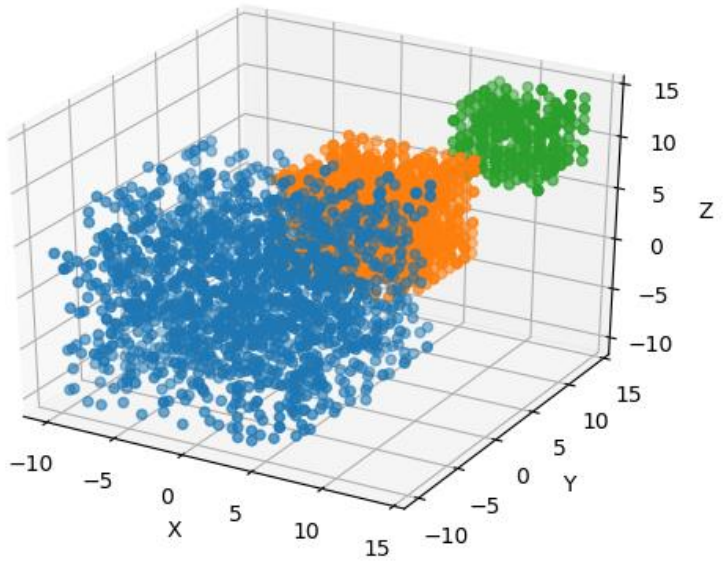
目录

聚类分析简介

参考书：《数据挖掘导论》第7章

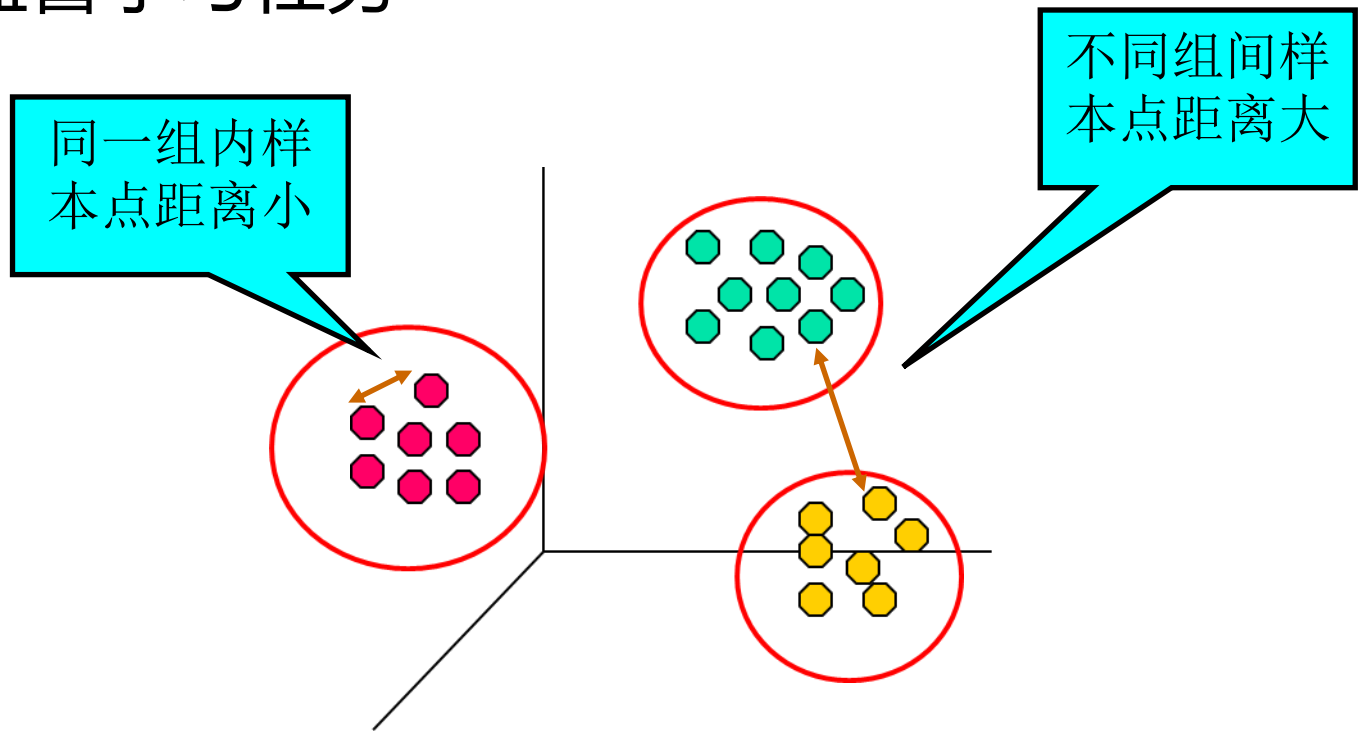
动机

- 对于给定的数据集能否找到一种好的分组方式?
- 如何决定将数据集分成多少组?
- 对于每个组，还可以进一步划分出子组吗?



什么是聚类分析(cluster analysis)

- 在聚类分析任务中，需要将数据集合划分成许多组，使得同一组内的数据具有较高的相似度，而不同组之间的数据相似度较低
- 聚类分析是最常见的无监督学习任务



聚类分析的常见应用

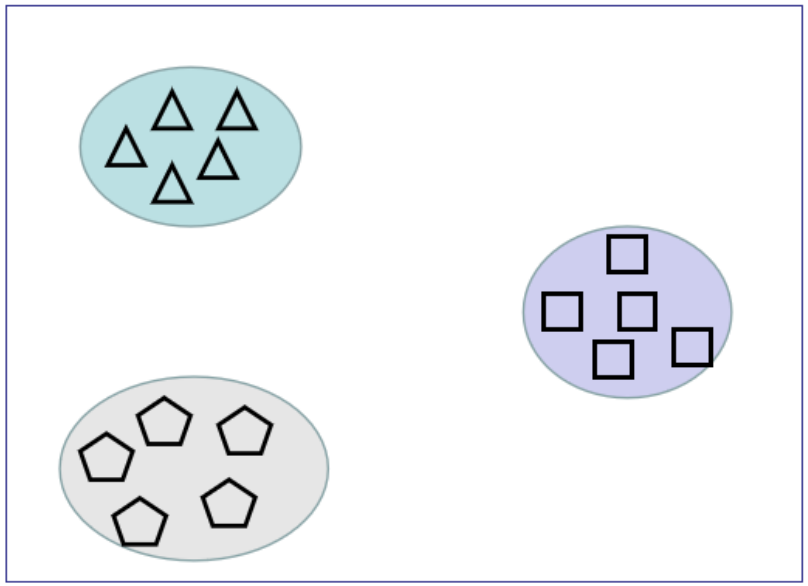
- 商务智能
  - 客户分组
  - 推荐
- WWW应用
  - 文档分组
  - 聚类Weblog数据以发现相似的访问模式组
- 模式识别
- 空间数据分析

聚类分析的功能

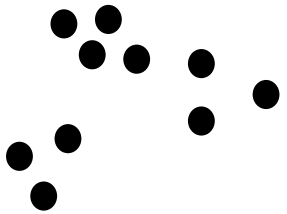
- 理解数据
  - 对浏览过的相关文档进行分组；
  - 对具有相似功能的基因和蛋白质进行分组；
- 汇总
  - 减少大规模数据集的大小
- 预处理
  - 其他数据挖掘算法的预处理步骤

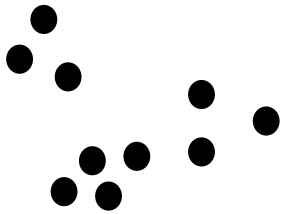
簇（cluster）

- 簇的定义
  - 簇（Cluster）为给定数据集合的子集，满足：
    - 1）同一簇内的数据彼此相似
    - 2）不同簇间的数据彼此相异
- 相关概念
  - 簇中心；
  - 簇大小；
  - 簇密度；

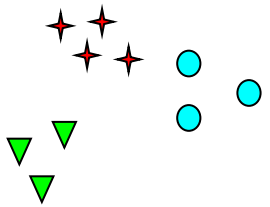


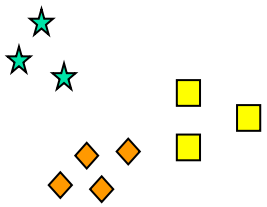
簇的概念可能不明确



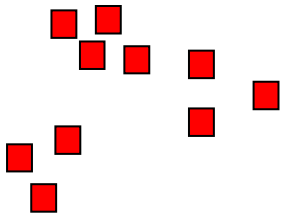


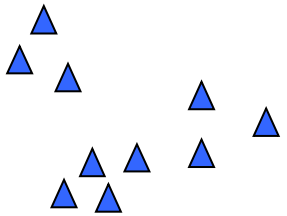
有多少个簇？



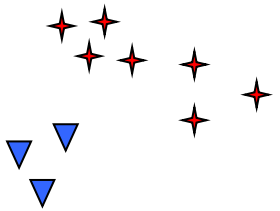


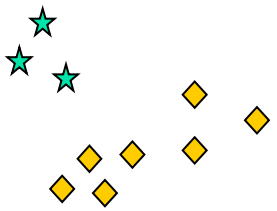
六个簇





两个簇



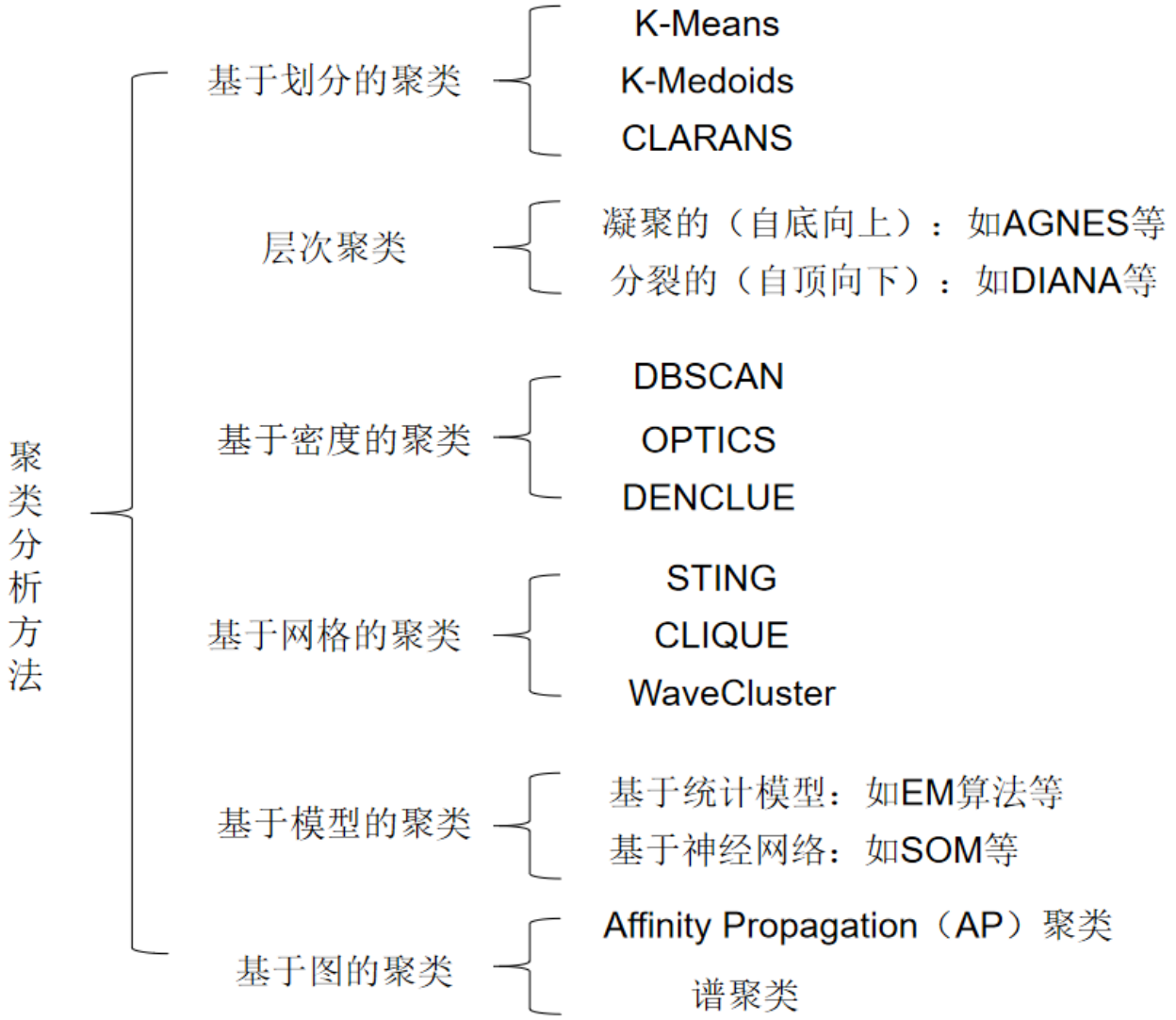


四个簇

什么是好的聚类

- 一个好的聚类方法可以产生高质量的簇：
  - 簇内相似度高（high intra-cluster similarity）
  - 簇间相似度低（low inter-cluster similarity）
- 可伸缩性
- 处理噪声的能力
- 发现任意形状的簇
- 聚类高维数据的能力

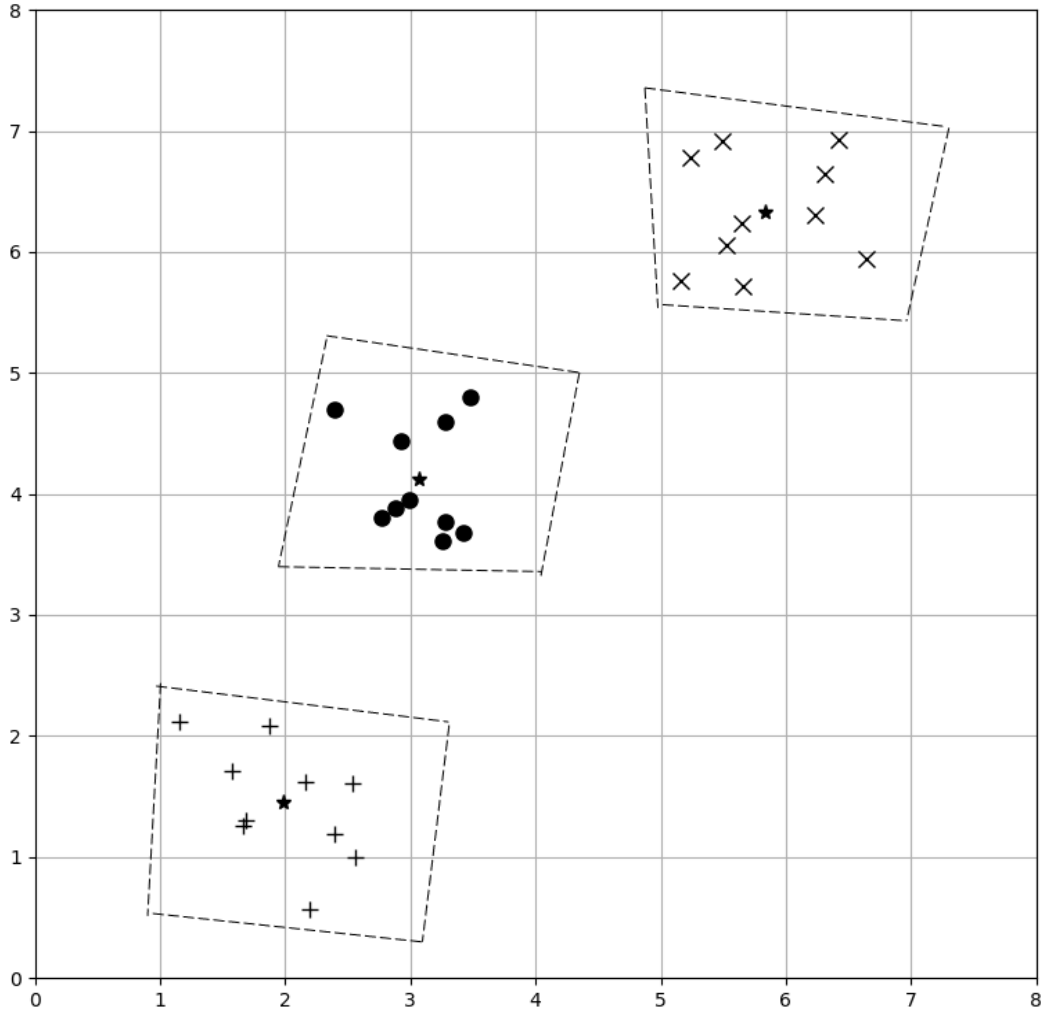
聚类方法概览


聚类分析方法
基于划分的聚类
K-Means
K-Medoids
CLARANS
层次聚类
凝聚的（自底向上）：如AGNES等
分裂的（自顶向下）：如DIANA等
基于密度的聚类
DBSCAN
OPTICS
DENCLUE
基于网格的聚类
STING
CLIQUE
WaveCluster
基于模型的聚类
基于统计模型：如EM算法等
基于神经网络：如SOM等
基于图的聚类
Affinity Propagation（AP）聚类
谱聚类

基于划分的聚类算法

基于划分的聚类

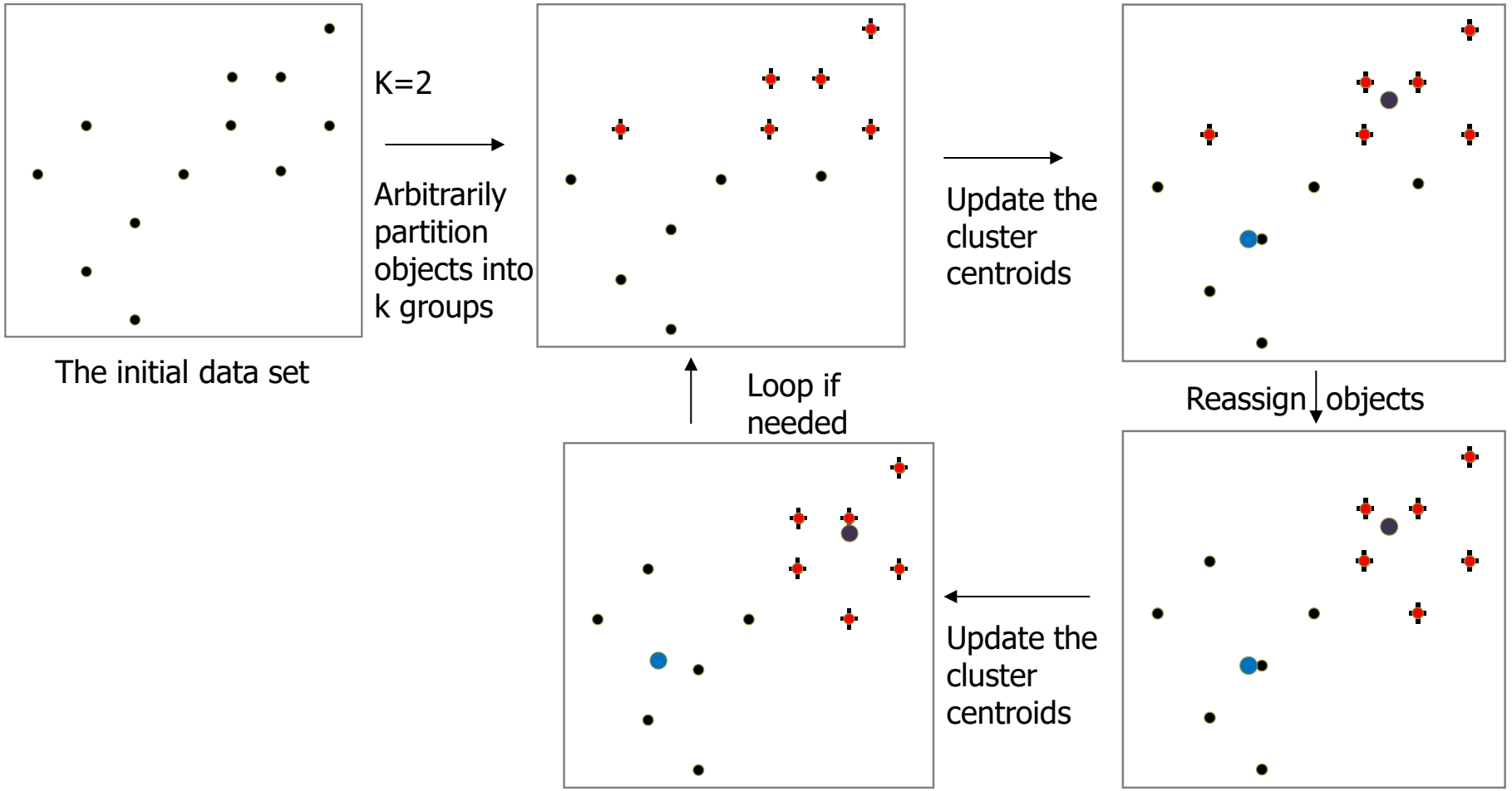
- 将m个数据对象划分成K个簇
  - 优化某一划分指标
- 全局优化
  - 检查所有可能的划分——NP-Hard
- 贪心方法：k-means
  - k-means算法中的簇是基于中心的



k-means算法的基本思想

- 基于划分的聚类方法
- 每个簇和一个中心关联
- 预先指定簇数K
- 主要步骤：
  - 随机选择初始的簇中心（cluster centroids）
  - 每个样本被分配到距离最近的中心所对应的簇中
  - 根据分配的结果重新计算各个簇中心
  - 不断迭代直到簇中心不变（或基本不变）

K-Means算法的简单示例



k-means算法的伪代码

输入： 样本集 $D = \{x_1, x_2, ..., x_m\}$；预定义聚类簇数 K

输出： 聚类结果 $C = \{C_1, C_2, ..., C_K\}$

初始化 K 个簇和簇中心：$C_i = \emptyset, i = 1, 2, ...K$。并从样本集 D 中随机选择 K 个样本作为对应簇的初始中心：$\{\mu_1, \mu_2, ..., \mu_K\}$

repeat

  for j=1,2,...,m **do**

    计算样本点 x_j 到每个簇中心的距离：$Dis(x_j, \mu_k) = ||x_j - \mu_k||$，并得到最近的簇中心 μ_s;

    将样本 x_j 重新划入到距离最近的簇中心 μ_s 对应的簇 C_s 中;

  end for

  for i=1,2,...,K **do**

    计算新的簇中心：$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$

  end for

until 簇中心不再发生变化

k-means算法执行的示例

- 例子：数据集D={(1, 2),(5, 7),(2, 2),(5, 6)}，聚类簇数为2，初始簇中心为：

  $\mu_1 = (1, 2)$和$\mu_2 = (2, 2)$

  - 第一轮迭代：
    - ✓ $\mu_1 = (1, 2)$，C_1={(1, 2)}；$\mu_2 = (2, 2)$，C_2={(2, 2),(5, 6),(5, 7)}
    - ✓ $\mu_1 = (1, 2)$，$\mu_2 = (4, 5)$
  - 第二轮迭代：
    - ✓ $\mu_1 = (1, 2)$，C_1={(1, 2),(2, 2)}；$\mu_2 = (4, 5)$，C_2={(5, 6),(5, 7)}
    - ✓ $\mu_1 = (1.5, 2)$，$\mu_2 = (5, 6.5)$
  - 第三轮迭代：
    - ✓ $\mu_1 = (1.5, 2)$，C_1={(1, 2),(2, 2)}；$\mu_2 = (5, 6.5)$，C_2={(5, 6),(5, 7)}
    - ✓ $\mu_1 = (1.5, 2)$，$\mu_2 = (5, 6.5)$

和上一轮结果一样，迭代终止！

k-means： A Mathematical Programming Problem

- 输入
  - 包含m个样本的数据集$D = \{x_1, x_2, \ldots, x_m\}$； 划分簇数K
- 输出
  - 划分方式$C = \{C_1, C_2, \ldots, C_K\}$
- 目标（损失函数）： $E = \sum_{i=1}^{K} \sum_{x \in C_i} ||x - \mu_i||_2^2$
  - 其中μ_i为第i个簇C_i的簇中心，该损失也称为SSE（Sum of the Squared Errors，误差平方和）

k-means： A Mathematical Programming Problem

- 目标（损失函数）

$$E = \sum_{i=1}^{K} \sum_{x \in C_i} ||x - \mu_i||_2^2$$

- 坐标下降法
  - 固定簇中心，优化分组方式：

$$x \in C_i, \qquad iff: \ i = \underset{j}{\operatorname{argmin}} \, Dis(x, \mu_j)$$

    - ✓ 其中$Dis(\cdot,\cdot)$为距离度量函数，如欧式距离

k-means： A Mathematical Programming Problem

- 坐标下降法
  - 固定分组方式，优化簇中心：

$$\frac{\partial E}{\partial \mu_i} = \frac{\partial(\sum_{i=1}^{K}\sum_{x\in C_i}||x-\mu_i||_2^2)}{\partial \mu_i}$$
$$= \frac{\partial(\sum_{x\in C_i}(||x-\mu_i||_2^2)}{\partial \mu_i}$$
$$= \sum_{x\in C_i}\frac{\partial(||x-\mu_i||_2^2)}{\partial \mu_i}$$
$$= -\sum_{x\in C_i}2(x-\mu_i)$$

  - ✓ 由$\frac{\partial E}{\partial \mu_i}=0$，有$\mu_i = \frac{1}{|C_i|}\sum_{x\in C_i}x$ 簇内样本均值（means）

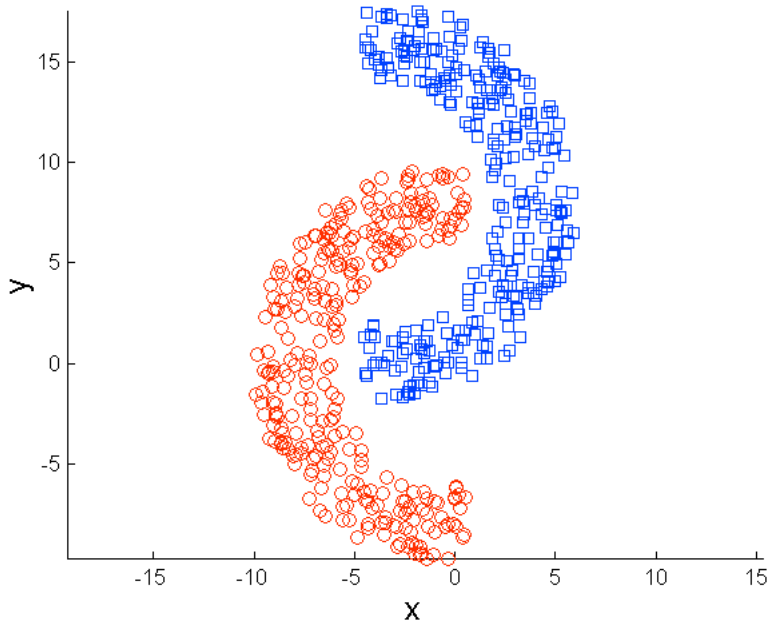
k-means算法的特点

- 复杂度：$O(mKT)$，其中m为样本数，K为簇数，T为迭代次数。
- 优点
  - 简单易实现
  - 相对有效，通常能取得不错的结果
- 多次运行取优

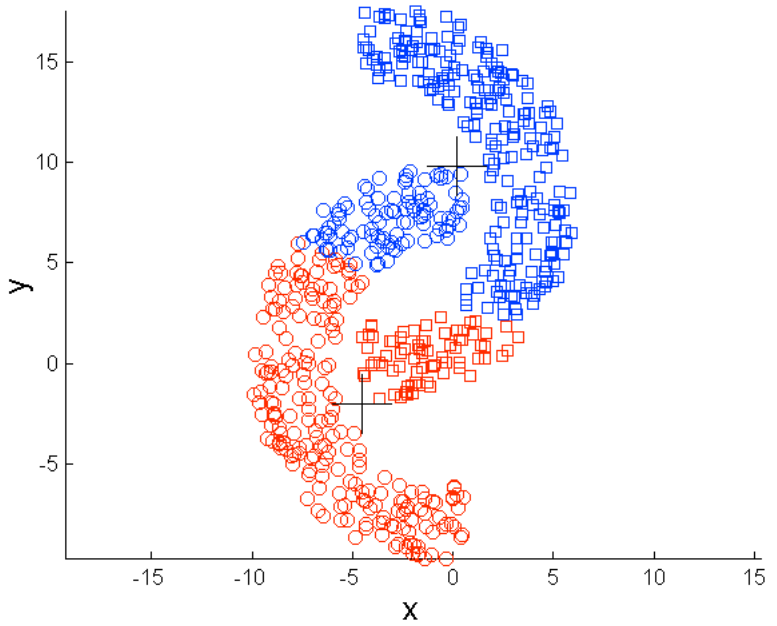
k-means算法的缺点

- 缺点
  - 属性均值有意义方可使用该算法
  - 簇数K需要提前给定，而且算法对K值敏感；
  - 对离群点、噪声敏感；
  - 不能保证收敛于全局最优，受初始簇中心影响；
  - 不适合发现非凸簇，以及大小差别很大的簇；

K-means算法的缺点：非球形簇的问题



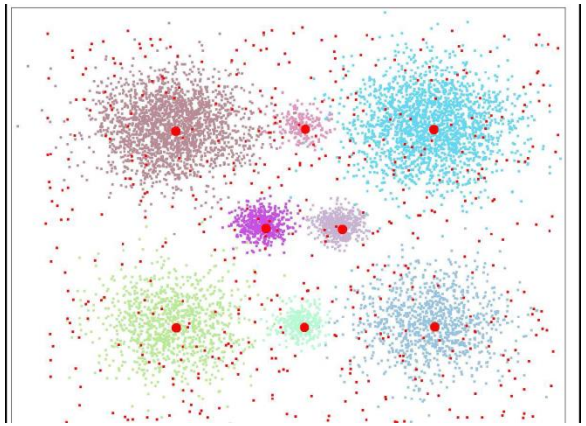
Original Points



K-means (2 Clusters)

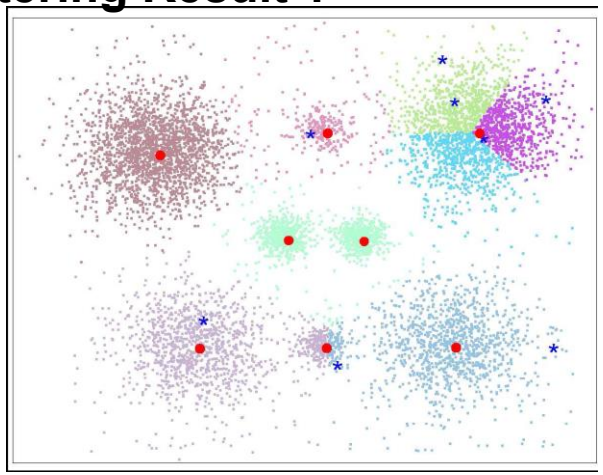
初始中心点选择问题

- *The clustering result of k-means is very sensitive to the selection of initial cluster centers*

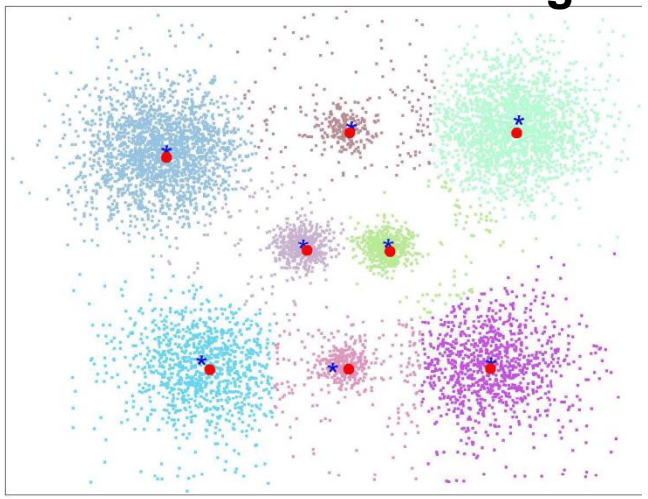


Original Data Set

Clustering Result 1



Clustering Result 2



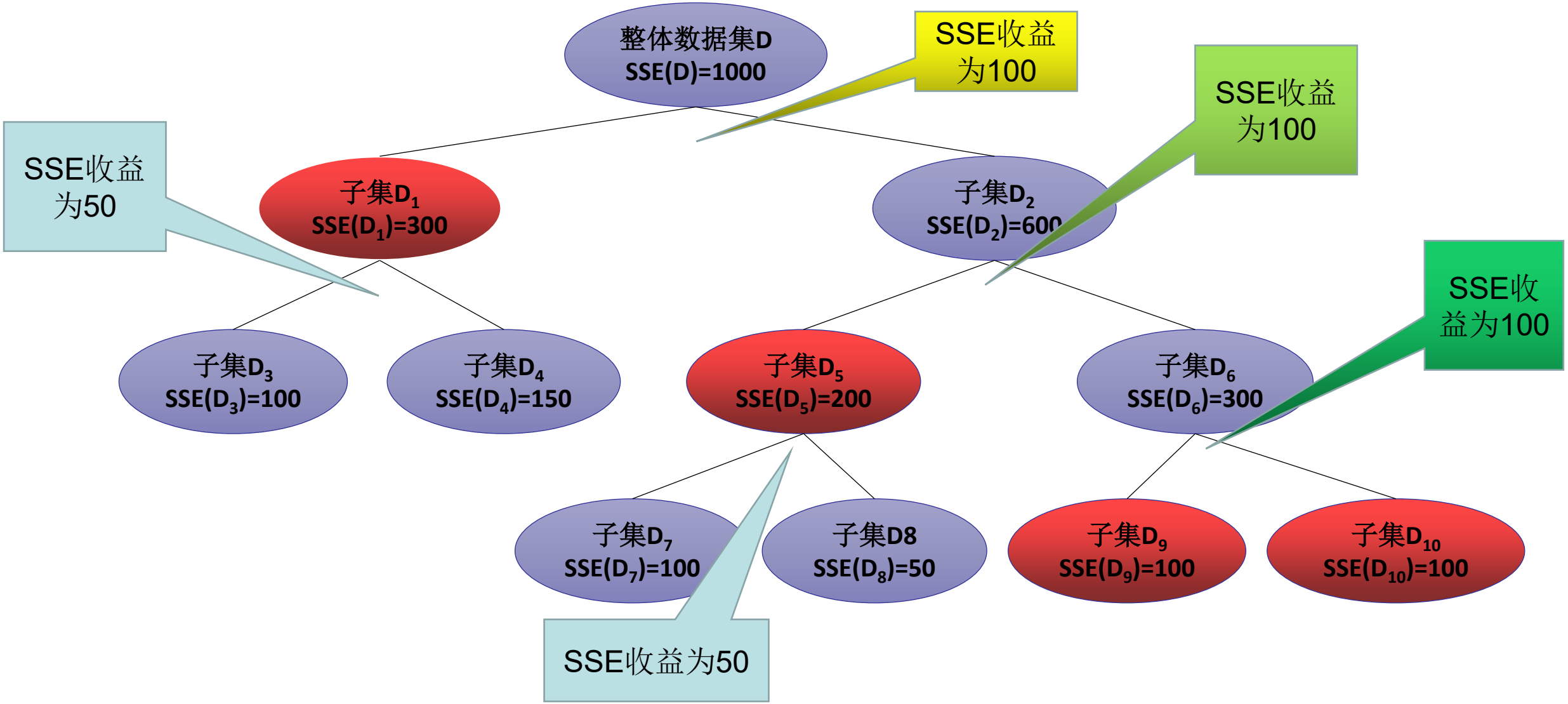
k-means类型算法

- 衍生算法
  - k-modes
  - k-median
  - k-medoids
  - k-means++
  - 二分k-means
  - ……

二分k-means

- 二分k-means：以k-means为基础，但本质思想有差别
  - 也是为了解决k-means算法易受初始簇中心影响，常常收敛到局部最优解的问题
- 一种类似于“决策树”的划分过程
  - 首先将所有样本看成一个簇，然后执行k=2的k-means算法（也即将簇一分为二）；
  - 之后不断选择某个簇做同样的划分；
    - ✓ 选择的标准是划分后能最大限度降低SSE（误差平方和）
  - 直到簇数等于预先设定的数值

二分k-means


整体数据集D
SSE(D)=1000
SSE收益为100
SSE收益为100
SSE收益为50
子集D_1
SSE(D_1)=300
子集D_2
SSE(D_2)=600
SSE收益为100
子集D_3
SSE(D_3)=100
子集D_4
SSE(D_4)=150
子集D_5
SSE(D_5)=200
子集D_6
SSE(D_6)=300
子集D_7
SSE(D_7)=100
子集D8
SSE(D_8)=50
子集D_9
SSE(D_9)=100
子集D_{10}
SSE(D_{10})=100
SSE收益为50

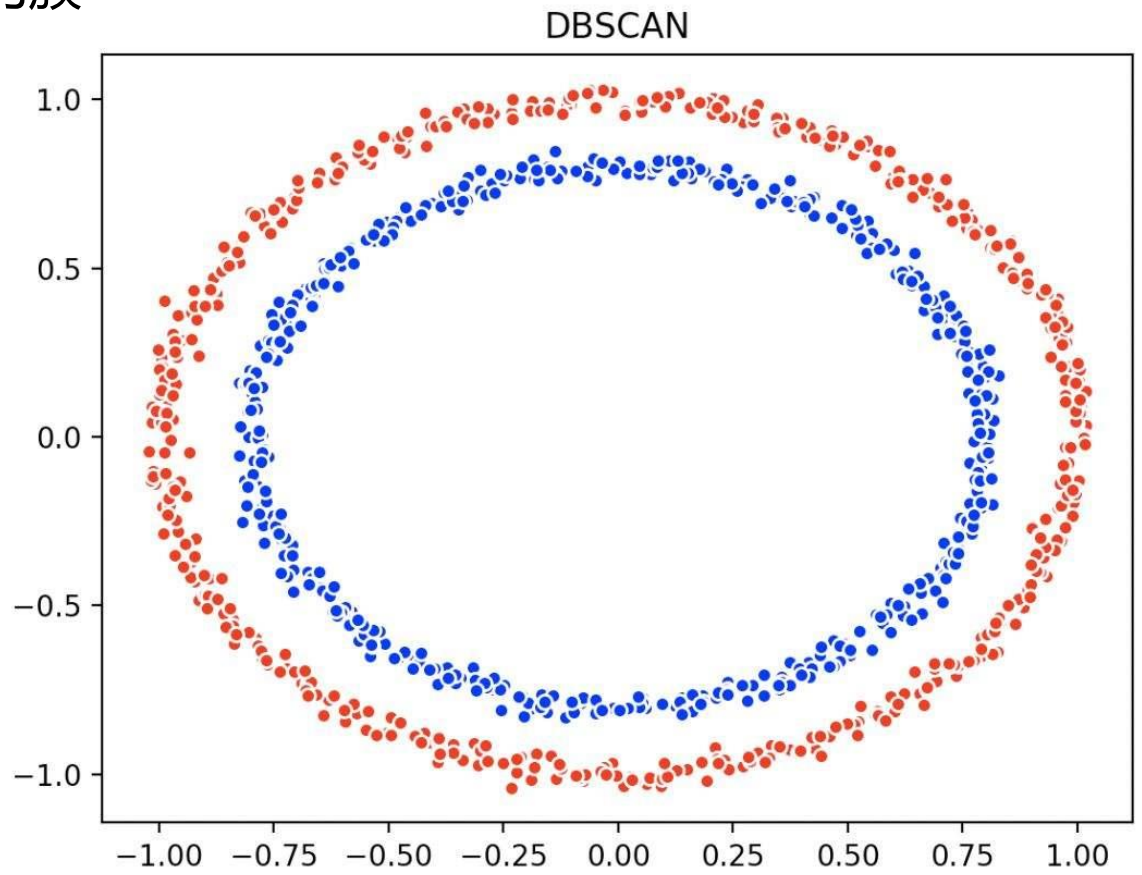
二分k-means

- 几乎不需要随机初始的簇中心
  - 尽管内部k=2的k-means算法仍需随机初始的簇中心，但此时k值仅为2，而且可以执行多次取最优
- 直接以SSE作为分裂标准
- 复杂度
  - 2k-3次k=2的k-means算法（记录下每个簇划分前后的SSE）
  - 每次需要划分的簇都是其父簇的子簇，于是每次执行k=2的k-means的样本点数目会指数级的小于全样本数目

基于密度的聚类算法

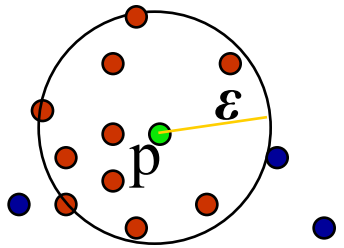
基于密度的聚类

- 簇可以看成特征空间中被稀疏区域分隔的稠密区域
  - 通过连接密度较大的区域，形成不同形状的簇
- 基于密度的聚类方法
  - DBSCAN
  - OPTICS
  - DENCLUE
  - ……



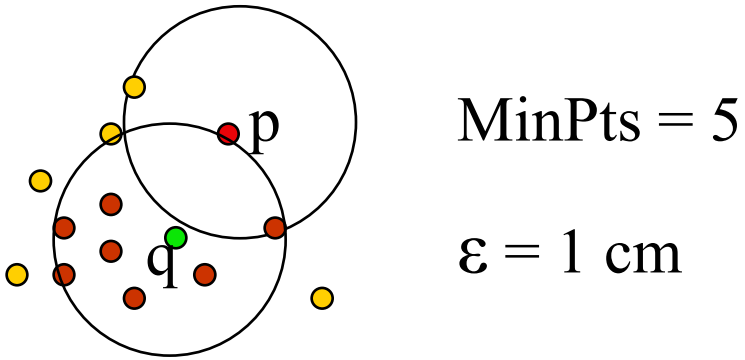
基于密度的聚类：定义

- 两个参数:
  - ε: Maximum radius of the neighborhood
  - *MinPts*: Minimum number of points in an Eps-neighborhood of that point
- $N_{\varepsilon}(p)$:{*q belongs to D* | *dist*(*p*,*q*) <= ε}



基于密度的聚类：定义

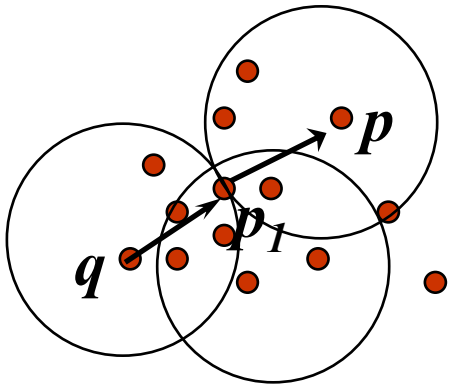
- 直接密度可达（**Directly density-reachable**）: **A point *p* is directly density-reachable from a point *q* wrt. *Eps*, *MinPts* if**
  - 1) *p* belongs to $\boldsymbol{N}_{\varepsilon}$ *q*)
  - 2) core point condition: | $\boldsymbol{N}_{\varepsilon}(q)$| >= *MinPts*



直接密度可达不满足对称性！

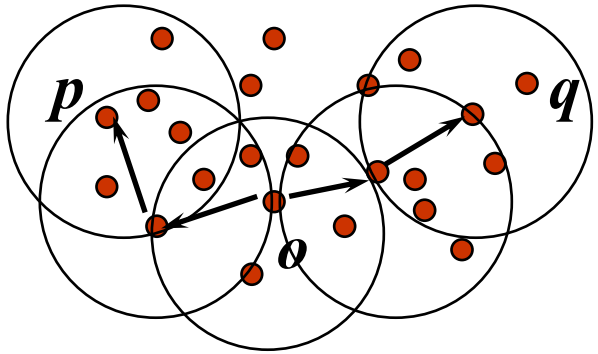
基于密度的聚类：定义

- 密度可达（**Density-reachable**）：
  - A point *p* is density-reachable from a point *q* wrt. *Eps*, *MinPts* if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i
  - 密度可达也不满足对称性
  - 满足传递性



基于密度的聚类：定义

- **密度连接（Density-connected）：**
  - A point *p* is density-connected to a point *q* wrt. ε, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* wrt. *Eps* and *MinPts*.
  - 密度连接满足对称性



簇的定义

- DBSCAN算法中的一个簇C定义为满足如下性质的样本点集合:
  - 对 $\forall x_i, x_j \in C$ ，有x_i和x_j密度相连;
  - 对 $\forall x_i \in C$ ，若x_j由x_i密度可达，则$x_j \in C$
- 结论
  - 如果样本点x_i是核心对象，那么所有由x_i密度可达的点可构成一个簇C_i ，即:

$$C_i = \{x_j | x_j \text{由} x_i \text{密度可达}\}$$

DBSCAN算法的主要步骤

- Step1：DBSCAN算法先计算出所有的核心对象
- Step2：之后随机选择一个核心对象，找到所有可由其密度可达的样本点，形成一个簇
  - 基于核心对象的广度优先搜索
- Step3：重复Step2，直到所有核心对象都被访问过

DBSCAN算法

输入： 样本集 $D = \{x_1, x_2, ..., x_m\}$；邻域超参数：$\epsilon, MinPts$

输出： 聚类结果 $C = \{C_1, C_2, ..., C_K\}$；噪声点集合 NS

计算核心对象集合 O：$O = \{x_i || N_\epsilon(x_i)| \geq MinPts\}$
初始化聚类簇数：$K = 1$
初始化未访问过的样本点集合：$P = D$
while $O \neq \emptyset$ **do**
    随机选取一个核心对象 $o \in O$，
    初始化根据该核心对象 o 考虑的样本点队列 $q = \{o\}$，和从该核心对象生成的簇 $C_K = \{o\}$
    $P = P \setminus \{o\}$
    while $q \neq \emptyset$ **do**
        取出 q 中的队首元素 x_j;
        if x_j 是核心对象 **then**
            令 $\Omega = N_\epsilon(x_j) \cap P$；//属于 x_j 的邻域但未被访问过的点
            $q = q \cup \Omega$；//将这些待考虑的点加入队列
            $P = P \setminus \Omega$；//标记这些点为已访问过
            $C_K = C_K \cup \Omega$；//这些点都是由 o 密度可达的
        end if
        $O = O \setminus C_K$；//从核心对象集合中剔除刚刚生成的簇中的核心对象
        K=K+1;
    end while
end while
$NS = P$；//所有仍未考虑过的点即为噪声点

DBSCAN

- 优点
  - 可以发现任意形状簇
  - 噪声鲁棒
  - 不需要预先指定聚类簇数
- 缺点
  - 对超参数ε和MinPts敏感
  - 不适合发现密度差异较大的簇

数据对象之间的距离

聚类分析的输入数据

- 许多聚类算法在实现时选择如下两种具有代表性的数据结构：
  - 数据矩阵（Data Matrix）：对象-属性结构，每一行代表一个样本，每一列代表一种属性（双模矩阵，two-mode matrix）
  - 相异度矩阵（Dissimilarity Matrix）：对象-对象结构，存储对象两两之间的相异度（单模矩阵，one-mode matrix）

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

聚类分析中的数据类型

- 区间标度变量（Interval-scaled variables）
- 二值变量（Binary variables）
- 标称型、序数型和比例型变量（Nominal, ordinal, and ratio variables）
- 混合类型变量（Variables of mixed types）

区间标度变量

- 大致呈线性比例的连续变量
  - 重量、高度、温度等
- 属性中度量单位的影响
  - 单元越小，变量变化范围越大，对结果的影响越大
  - 标准化+背景知识

相似度

- 通常使用数据对象之间的距离衡量相似度
- 闵可夫斯基距离（Minkowski distance）

$$Dis(x_i, x_j) = \sqrt[q]{|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \cdots + |x_{ip} - x_{jp}|^q}$$

  - q=1：曼哈顿距离（Manhattan distance）
  - q=2：欧氏距离（Euclidean distance）
  - q=∞：切比雪夫距离（Chebyshev distance）

二值变量

- 一个二值变量只有两种状态：0或1
  - 用于二值数据的列联表

Object i \ Object j

	1	0	Sum
1	q	r	q+r
0	s	t	s+t
Sum	q+s	r+t	p

- 对称变量:每个状态具有相同的权值
  - 不变相异度（Invariant dissimilarity）： $Dis(i,j) = \frac{r+s}{q+r+s+t}$
- 非对称变量:正值权重更大
  - 非不变相异度（Noninvariant dissimilarity），Jaccard系数： $Dis(i,j) = \frac{r+s}{q+r+s}$

二值变量相异度

- 例子

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

  - Gender是对称属性,但并不考虑
  - 其余的都不是
  - 假设值Y和P编码为1，值N编码为0，则Jaccard系数为：

$$Dis(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$Dis(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$Dis(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

标称型变量

- 标称型变量是二值变量的一般化，它可以有两种以上的状态，如颜色变量有红、黄、蓝、绿
  - 方法一：简单匹配方法

$$Dis(i,j) = \frac{p-m}{p}$$

    - ✓ m：匹配的数目, p： 全部变量的数目
  - 方法二：使用大量的二值变量
    - ✓ 为每一个状态创建一个新的二值变量

序数型变量

- 序数型变量可以是离散的，也可以是连续的
- 顺序是重要的，例如：排名
- 可以像区间标度变量那样处理
  - 使用顺序替换x_{if}：$r_{if} \in \{1, \ldots, M_f\}$
  - 将范围映射到[0,1]区间：$z_{if} = \frac{r_{if} - 1}{M_f - 1}$
  - 使用区间标度变量的方法计算相异度

比例型变量

- 比例型变量：非线性标度上取正测量值的变量
  - 例如，近似指数比例：Ae^{Bt}
- 把它们当作区间标度的变量?
  - 不是一个好的选择:比例会被扭曲!
- 运用对数变换：$y_{if} = \log(x_{if})$

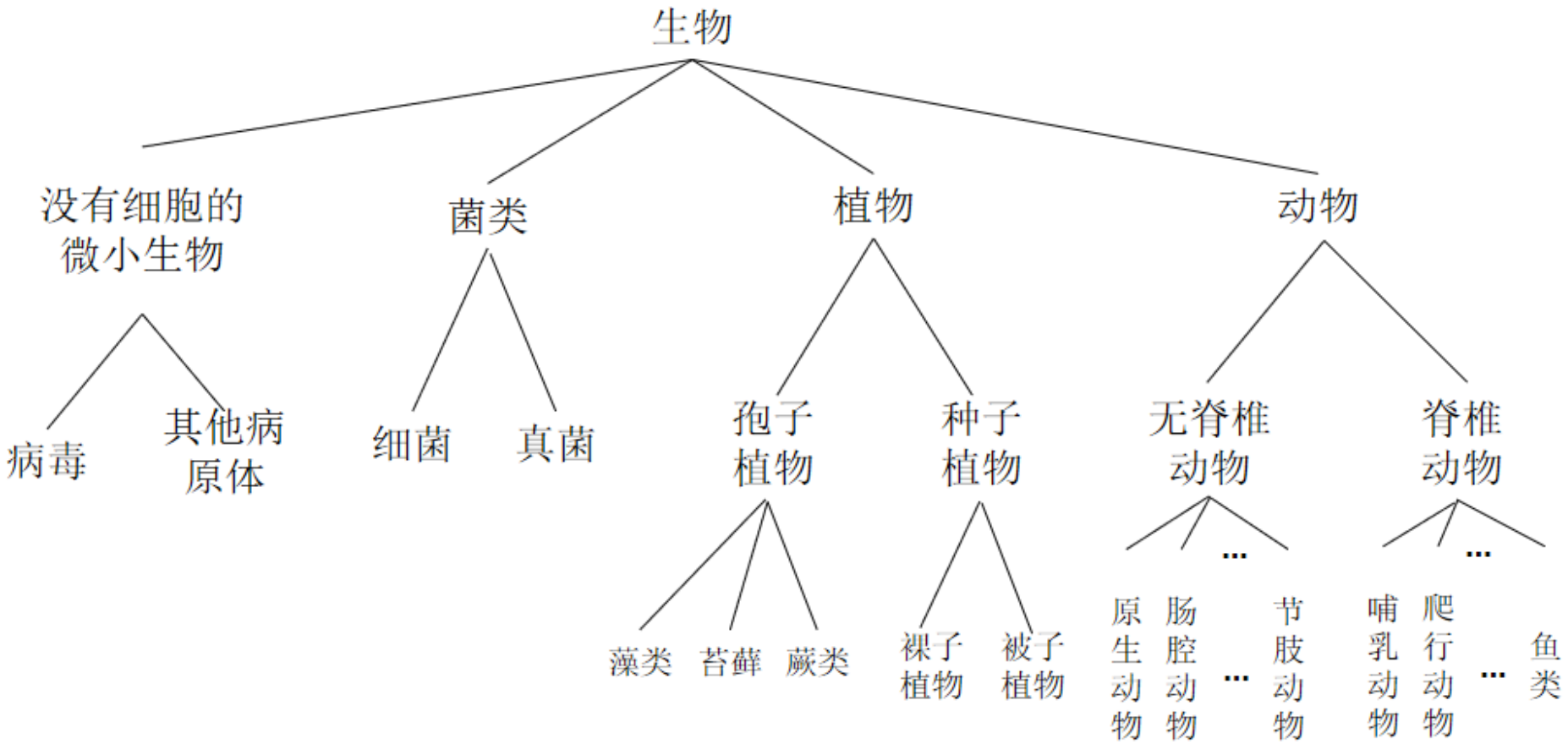
混合类型变量

- 数据库中可以包含各种类型的变量
  - 例如，对称二值型，非对称二值型，标称型，序数型等
- 可以使用加权公式来综合它们的影响

$$Dis(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}(f) Dis_{ij}(f)}{\sum_{f=1}^{p} \delta_{ij}(f)}$$

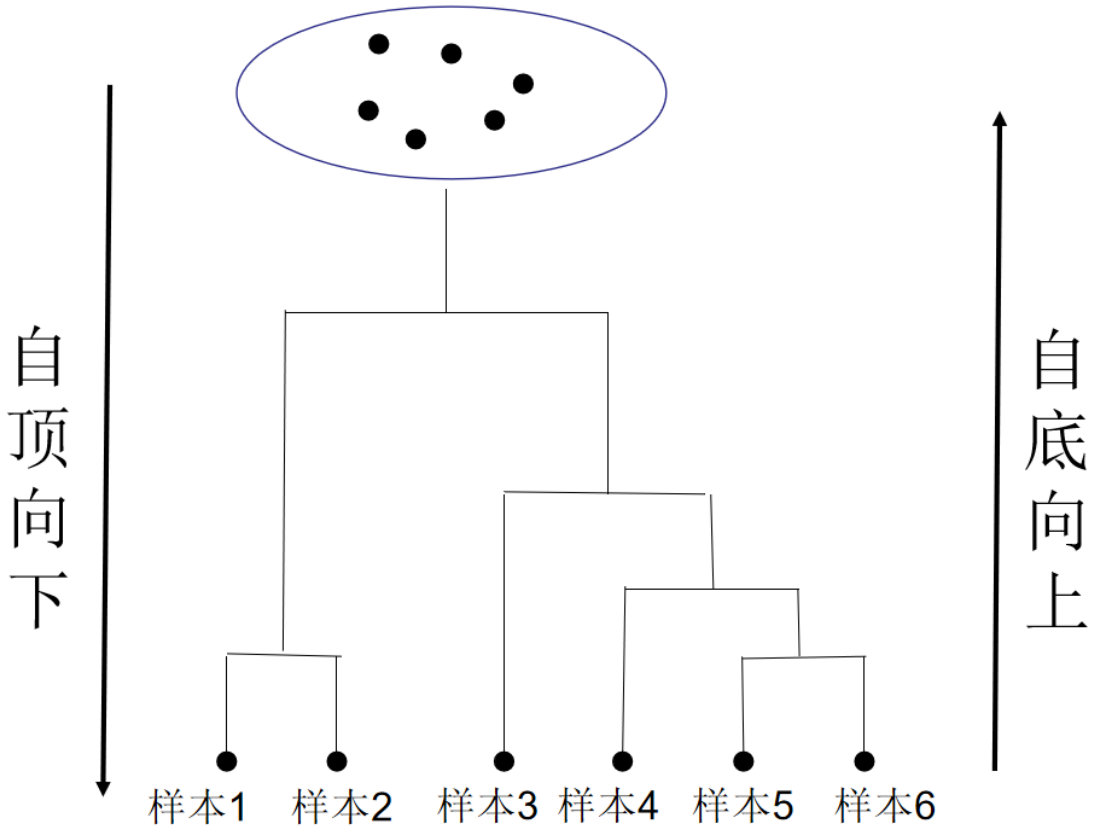
层次聚类算法

层次聚类


生物
没有细胞的微小生物
菌类
植物
动物
病毒
其他病原体
细菌
真菌
孢子植物
种子植物
无脊椎动物
脊椎动物
藻类
苔藓
蕨类
裸子植物
被子植物
...
原生动物
肠腔动物
节肢动物
哺乳动物
爬行动物
鱼类

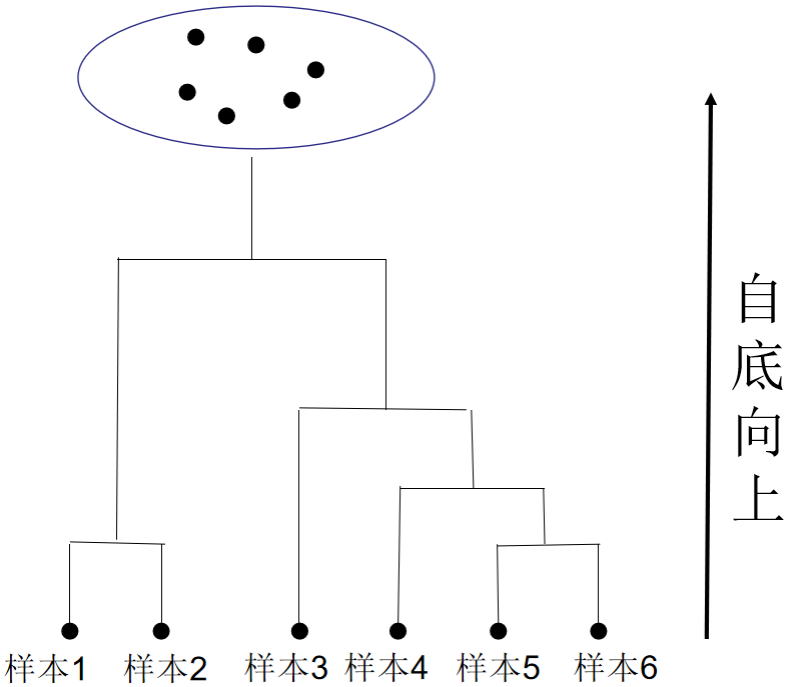
层次聚类

- 凝聚的层次聚类方法（自底向上）
  - AGNES算法
- 分裂的层次聚类方法（自顶向下）
  - DIANA算法
- 树状图（dendrogram）

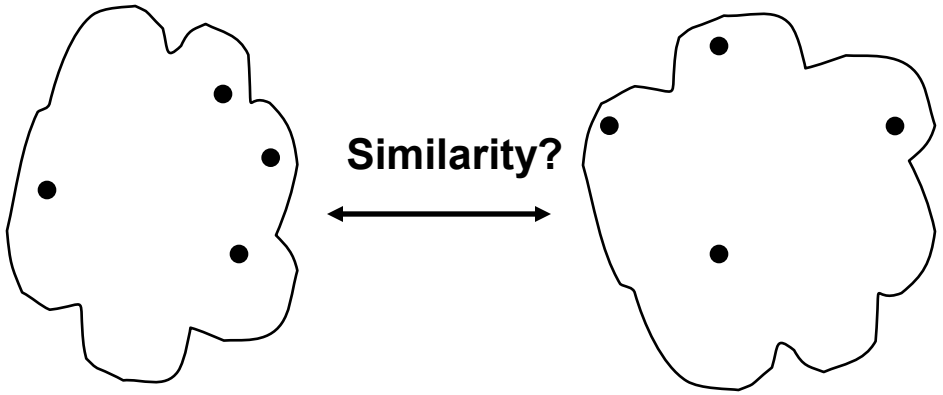


AGNES(Agglomerative Nesting)

- 自底向上
  - 首先将每个样本看成是一个初始簇
  - 之后每次合并距离最近的两个簇，直到簇数目到达预设的值



How to Define Inter-Cluster Similarity

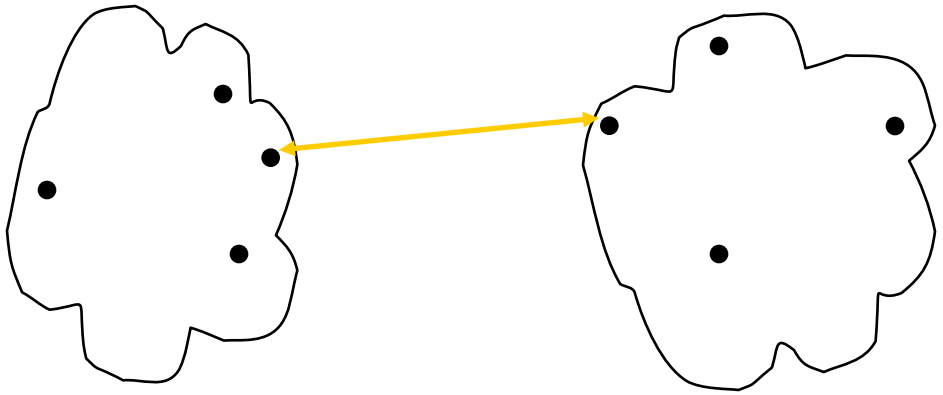


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

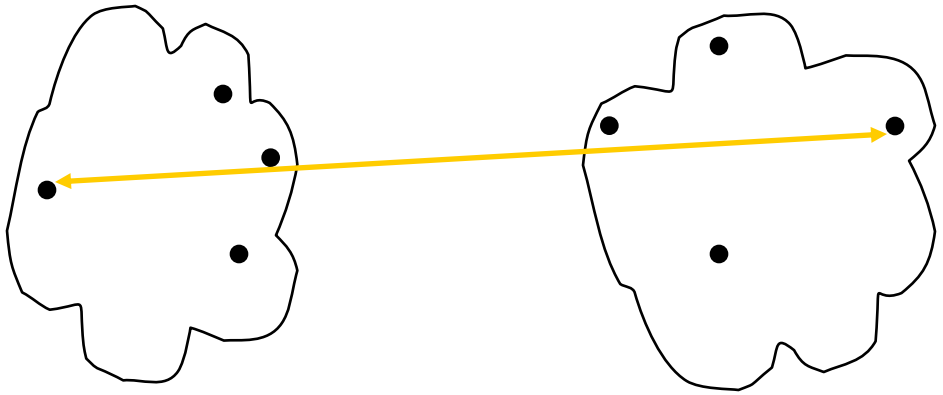


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

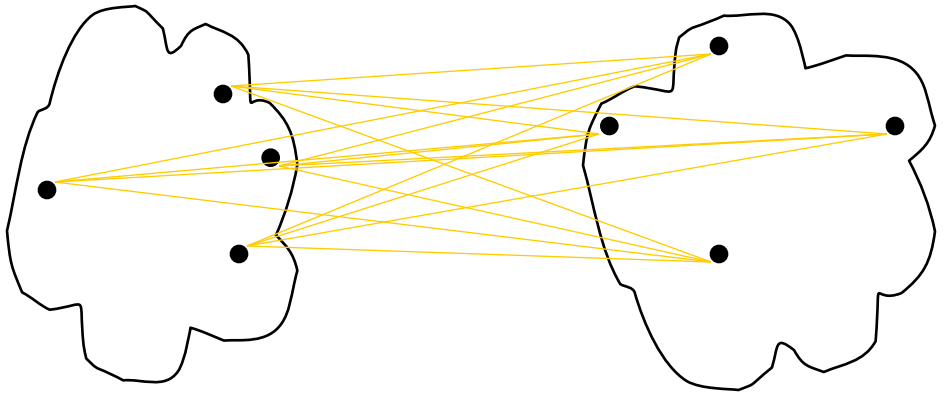


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

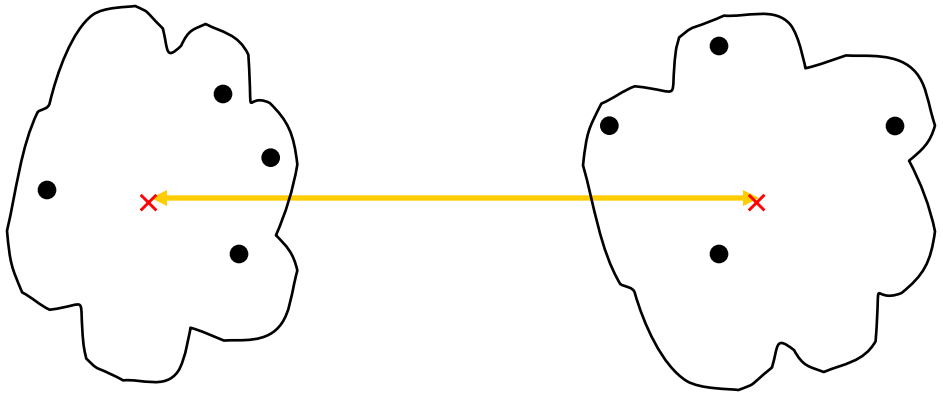


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

Distance Measures

- Minimum distance
- Maximum distance
- Mean distance
- Average distance

$$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$

$$d_{\max}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$

$$d_{mean}(C_i, C_j) = d(m_i, m_j)$$

$$d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{q \in C_j} d(p, q)$$

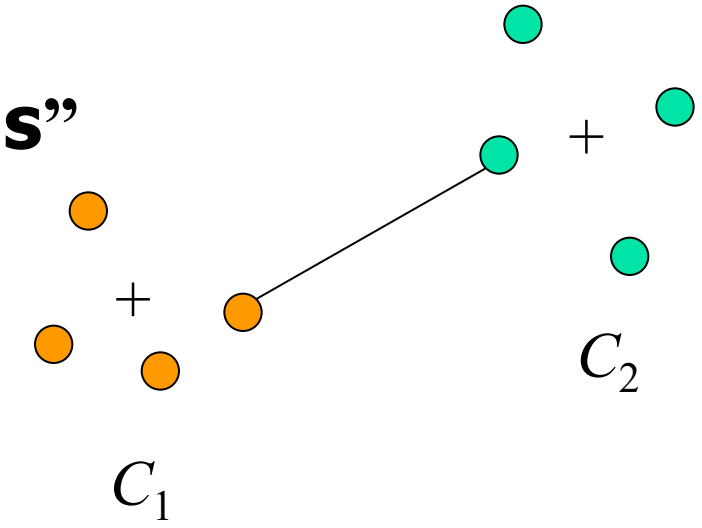
m: mean for a cluster
C: a cluster
n: the number of objects in a cluster

Merging Methods

- **Single linkage:** MIN distance
- **Complete linkage:** Maximum distance
- **Average linkage:** Mean distance
- **Centroid linkage:** Average distance

Single Linkage

- **Dissimilarity between two clusters = Minimum dissimilarity between the members of two clusters**
- **Only need dissimilarity matrix**
- **Tend to generate "long chains"**



Example

$$D_1 = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{pmatrix} 0.0 & & & & \\ 2.0 & 0.0 & & & \\ 6.0 & 5.0 & 0.0 & & \\ 10.0 & 9.0 & 4.0 & 0.0 & \\ 9.0 & 8.0 & 5.0 & 3.0 & 0.0 \end{pmatrix}$$

Example

$$D_1 = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{pmatrix} 0.0 & & & & \\ 2.0 & 0.0 & & & \\ 6.0 & 5.0 & 0.0 & & \\ 10.0 & 9.0 & 4.0 & 0.0 & \\ 9.0 & 8.0 & 5.0 & 3.0 & 0.0 \end{pmatrix}$$

$$d_{(12)3} = \min\left[d_{13}, d_{23}\right] = d_{23} = 5.0$$

$$d_{(12)4} = \min\left[d_{14}, d_{24}\right] = d_{24} = 9.0$$

$$d_{(12)5} = \min\left[d_{15}, d_{25}\right] = d_{25} = 8.0$$

Example

$$D_2 = \begin{matrix} (1,2) \\ 3 \\ 4 \\ 5 \end{matrix} \begin{pmatrix} 0.0 & & & \\ 5.0 & 0.0 & & \\ 9.0 & 4.0 & 0.0 & \\ 8.0 & 5.0 & 3.0 & 0.0 \end{pmatrix}$$

Example

$$D_2 = \begin{array}{c} (1,2) \\ 3 \\ 4 \\ 5 \end{array} \begin{pmatrix} 0.0 & & & \\ 5.0 & 0.0 & & \\ 9.0 & 4.0 & 0.0 & \\ 8.0 & 5.0 & 3.0 & 0.0 \end{pmatrix}$$

$$d_{(12)(45)} = \min\left[d_{(12)4}, d_{(12)5}\right] = d_{(12)5} = 8.0$$

$$d_{(45)3} = \min\left[d_{34}, d_{35}\right] = d_{34} = 4.0$$

Example

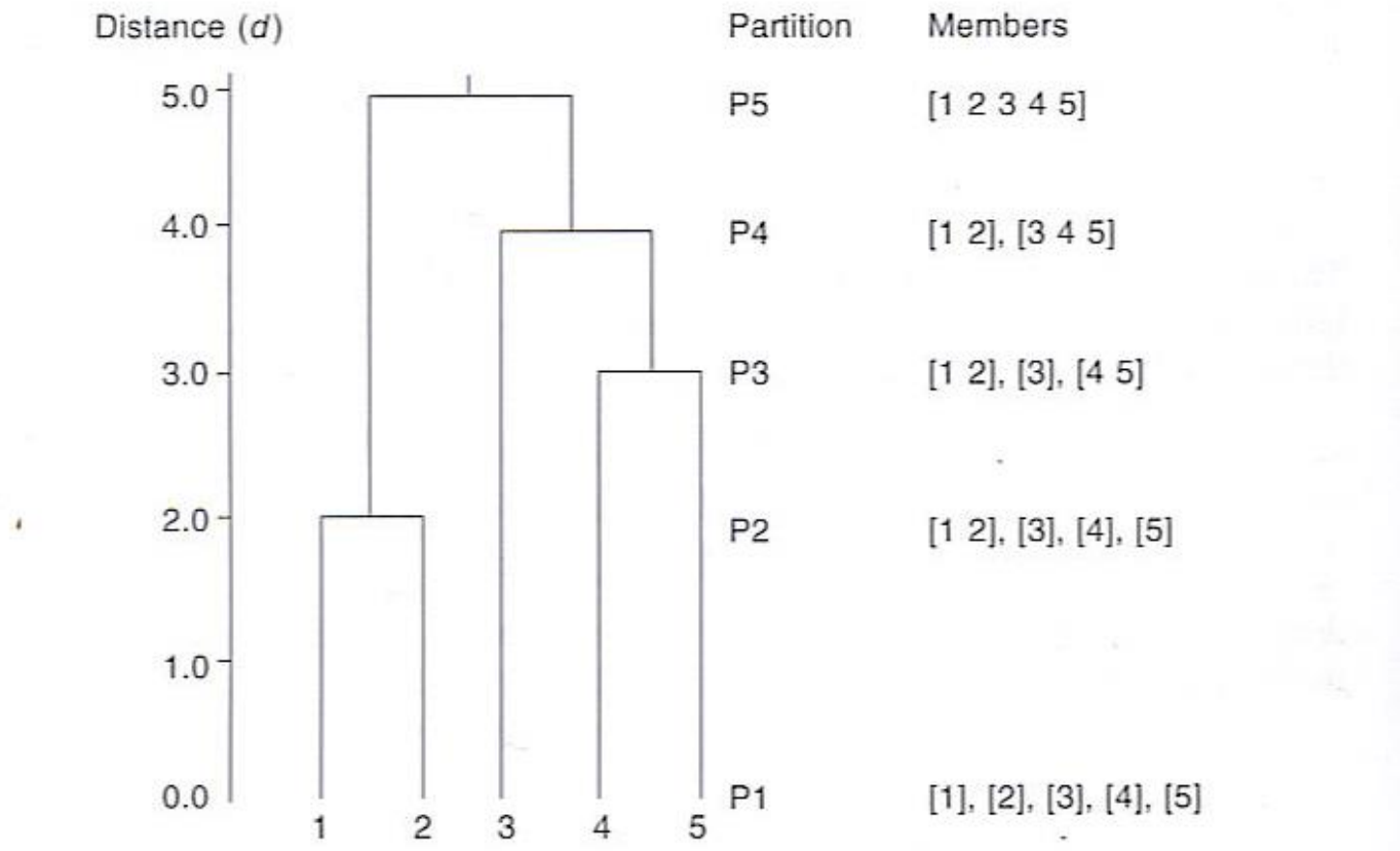
$$D_3 = \begin{matrix} (1,2) \\ 3 \\ (4,5) \end{matrix} \begin{pmatrix} 0.0 & & \\ 5.0 & 0.0 & \\ 8.0 & \textcircled{4.0} & 0.0 \end{pmatrix}$$

Example

$$D_3 = \begin{matrix} (1,2) \\ 3 \\ (4,5) \end{matrix} \begin{pmatrix} 0.0 & & \\ 5.0 & 0.0 & \\ 8.0 & 4.0 & 0.0 \end{pmatrix}$$

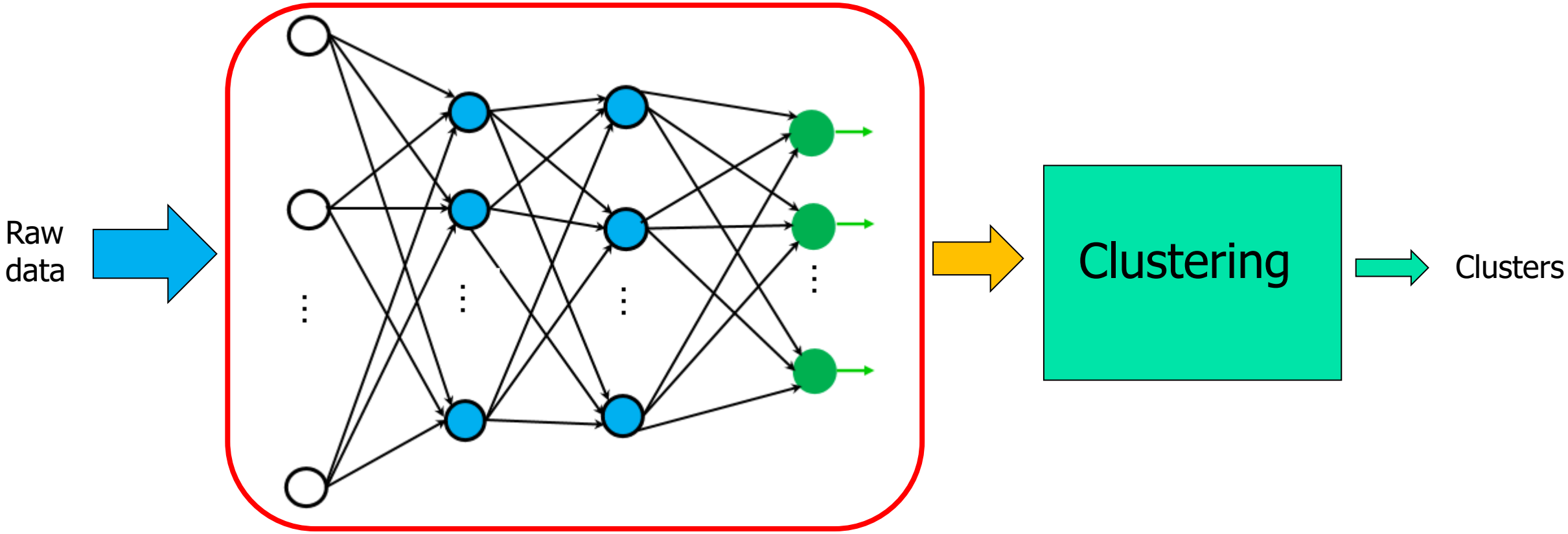
- Add individual 3 to the cluster containing individuals 4 and 5.
- Then merge the groups (1,2) and (3,4,5) into a single cluster.

Example

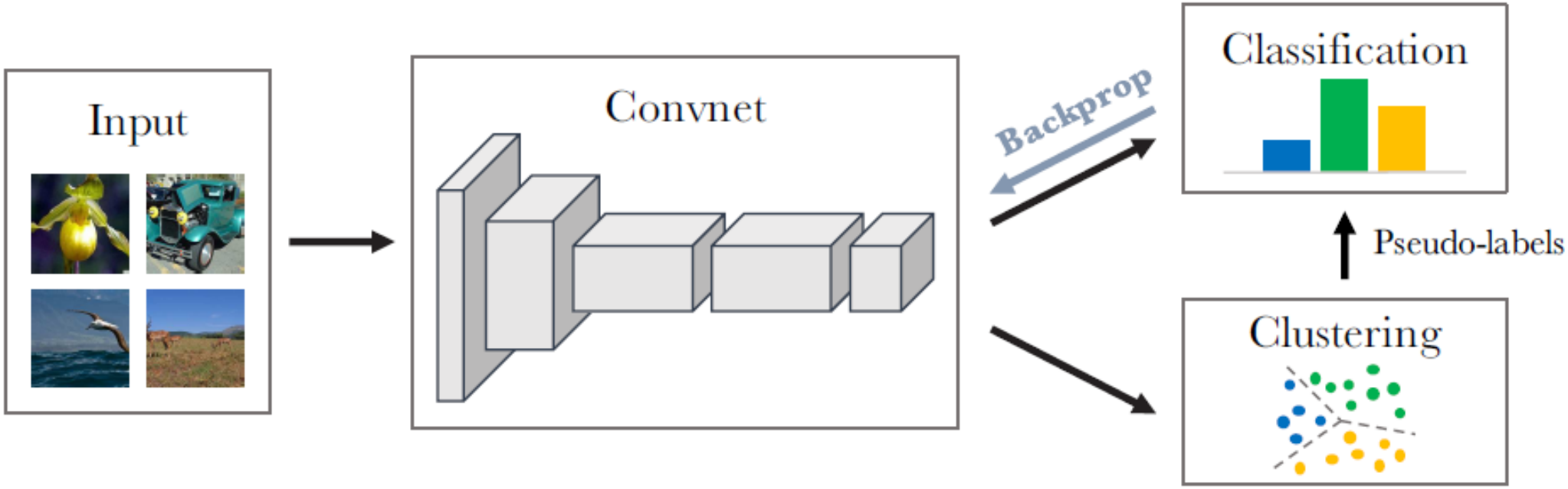

Distance (d)
5.0
4.0
3.0
2.0
1.0
0.0
1
2
3
4
5
Partition
Members
P5
[1 2 3 4 5]
P4
[1 2], [3 4 5]
P3
[1 2], [3], [4 5]
P2
[1 2], [3], [4], [5]
P1
[1], [2], [3], [4], [5]

基于深度学习的聚类方法基础

表征学习+聚类



深度聚类（Deep Clustering）

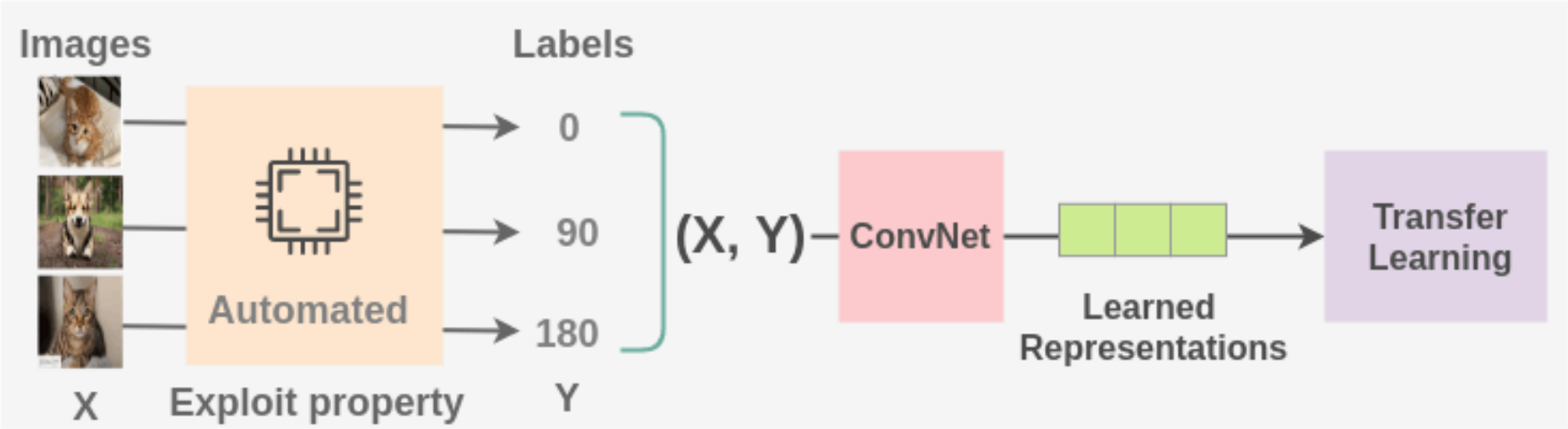


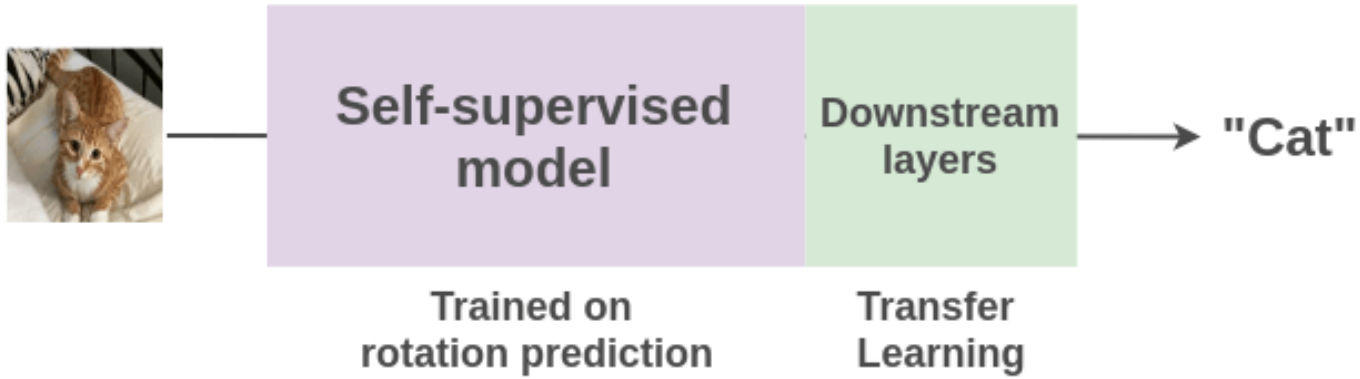
Deep clustering for unsupervised learning of visual features. In Proceedings of the European Conference on Computer Vision (ECCV), 2018.

Self-supervised Learning

- ***Supervised learning*** – learning with labeled data
- ***Unsupervised learning*** – learning with unlabeled data
- ***Self-supervised learning*** – representation learning with unlabeled data
  - Learn useful feature representations from unlabeled data through pretext tasks
  - The term "self-supervised" refers to creating its own supervision (without supervision&labels)
  - Self-supervised learning is one category of unsupervised learning

Self-Supervised Learning: an example





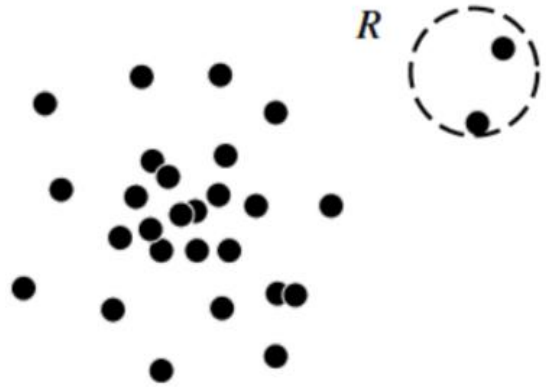
Picture from: Amit Chaudhary – The Illustrated Self-Supervised Learning

离群点检测简介

参考书：《数据挖掘导论》第9章

概述

- 离群点检测（outlier detection）：又称为异常检测（anomaly detection），是找出数据集中行为显著不同于预期对象的过程。
- 离群点：显著不同于数据集中其余数据对象的样本，当假定多数样本由某一随机过程产生时，离群点可认为是由不同的随机过程产生。



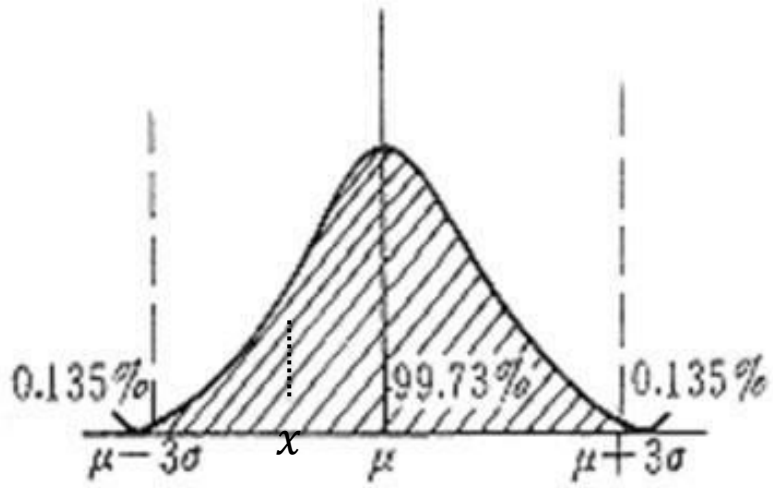
离群点检测的主要方法

- 统计学方法：3sigma准则
- 基于密度的方法：LOF
- 基于聚类的方法：k-means
- 基于基于分类的方法：OCSVM
- 距离的方法：最近邻，k近邻
- 基于重构的方法：PCA, autoencoder

基于统计学的方法

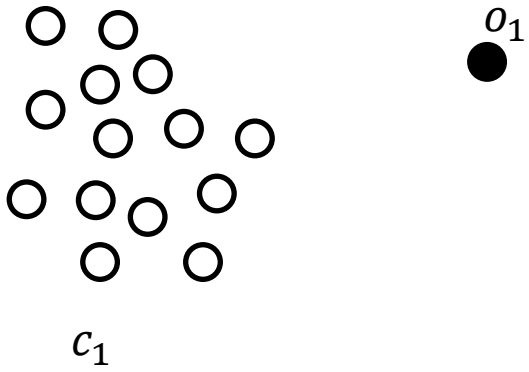
- **基本思想：**
  - **假定正常数据由统计随机模型产生，而离群点不遵从该类模型。**
  - **对于一维变量，当假定数据服从一维正太分布时，对于样本x,可以求得概率密度$p(x)$,若$p(x)$较小，则判为异常。**
  - **异常判断准则：3σ准则**

$[\mu-3\sigma,\mu+3\sigma]$范围内包含99.7%的数据，若数据位于该范围外，则判为异常。



基于距离的方法

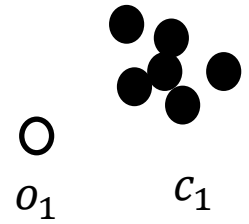
- **基本思想：**
  - **离群点和近邻之间的近邻性显著偏离数据集中其他对象和近邻之间的近邻性。**
  - **对于给定半径的邻域，当该对象邻域内没有足够多的其他样本时，该对象被认为是离群点。**

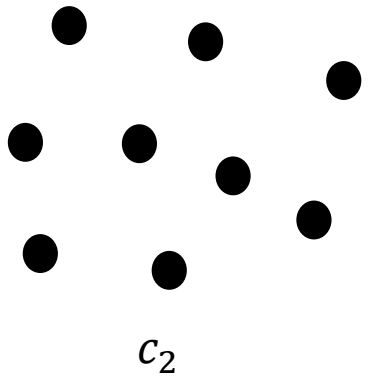


o_1和邻居的距离显著偏离c_1中对象和其邻居的距离，认为o_1是离群点

基于密度的方法

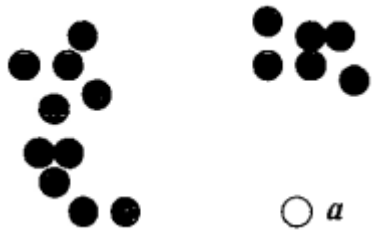
- **基本思想：**
  - **正常对象周围的密度与其邻域周围的密度类似，而离群点周围的密度显著不同于邻域周围的密度。**
  - **如图，o_1的局部密度显著偏离其最近邻的局部密度，而稀疏簇c_2中每一点的局部密度和其最近邻的局部密度一致，故o_1是异常点，c_2中每一点是正常点。**



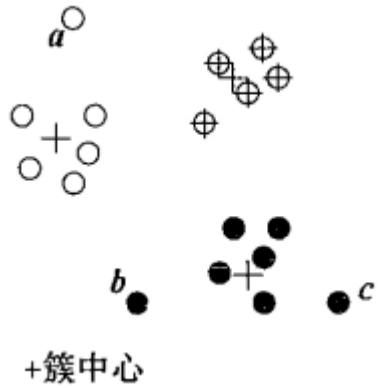


基于聚类的方法

- **基本思想：**
  - **离群点不属于某个簇，或与最近簇之间的距离很远。**



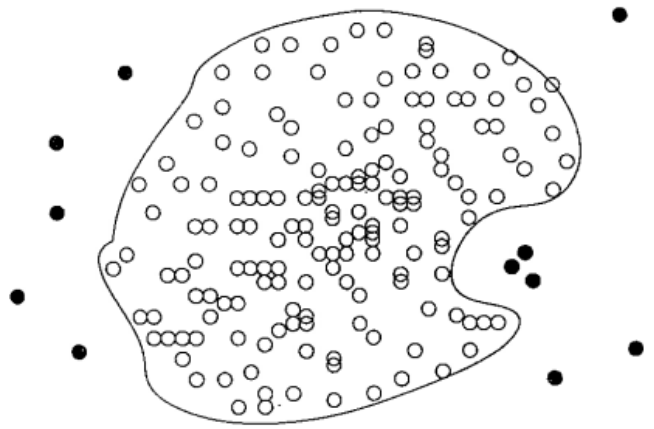
点a是离群点，因为不属于任何簇。



a b c是离群点，因为远离和其最近的簇

基于分类的方法

- **基本思想：**
  - **训练集高度有偏，大多数为正常点，只含有少数异常点**
  - **构建描述正常类边界的分类器**
  - **当样本落在正常类的决策边界之外时，判为异常。**



黑色样本落在正常类决策边界之外，故判为异常。

最大似然估计法: 问题定义

- Given a data set $\mathbb{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}\}$
- Assume the probability density function of $\mathbb{X}$ is known to be $f_\theta(x)$
  - θ is the parameters, and to be learned from data
- Likelihood of $\mathbb{X}$:

$$L(\theta) = \prod_{i=1}^{m} f_\theta(\boldsymbol{x}^{(i)}) = f_\theta(\boldsymbol{x}^{(1)}) f_\theta(\boldsymbol{x}^{(2)}) \ldots f_\theta(\boldsymbol{x}^{(m)})$$

- Maximize:

$$\theta^*: \ \operatorname*{argmax}_\theta L(\theta) = \operatorname*{argmax}_\theta \left(\prod_{i=1}^{m} f_\theta(\boldsymbol{x}^{(i)}) \right)$$

- Given a data set $\mathbb{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(m)}\}$ $\quad\quad$ $\boldsymbol{x}^{(i)} \in \boldsymbol{R}^d$

- Assume the distribution of $\mathbb{X}$ be Multivariate Gaussian:

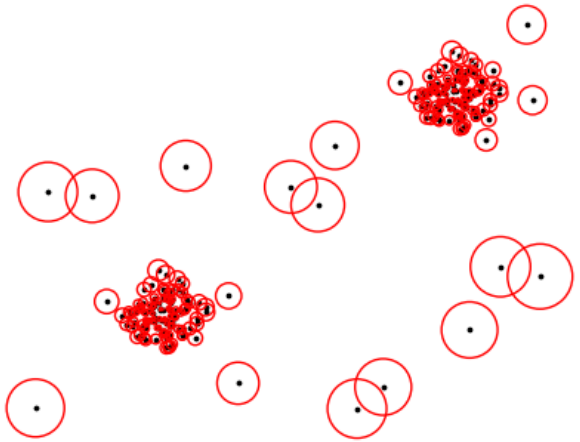
$$\theta\colon\ \boldsymbol{\mu}, \boldsymbol{\sigma} \qquad f_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}} \frac{1}{|\boldsymbol{\sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$$

- Maximize:

$$\theta^*\colon\ \underset{\theta\colon \boldsymbol{\mu},\boldsymbol{\sigma}}{\operatorname{argmax}} L(\theta) = \underset{\boldsymbol{\mu},\boldsymbol{\sigma}}{\operatorname{argmax}} \left(\prod_{i=1}^{m} f_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{x}^{(i)}) \right)$$

$$\boldsymbol{\mu}^* = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}^{(i)} \qquad \boldsymbol{\sigma}^* = \frac{1}{m}\sum_{i=1}^{m} \left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}^*\right)\left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}^*\right)^T$$

检测阶段



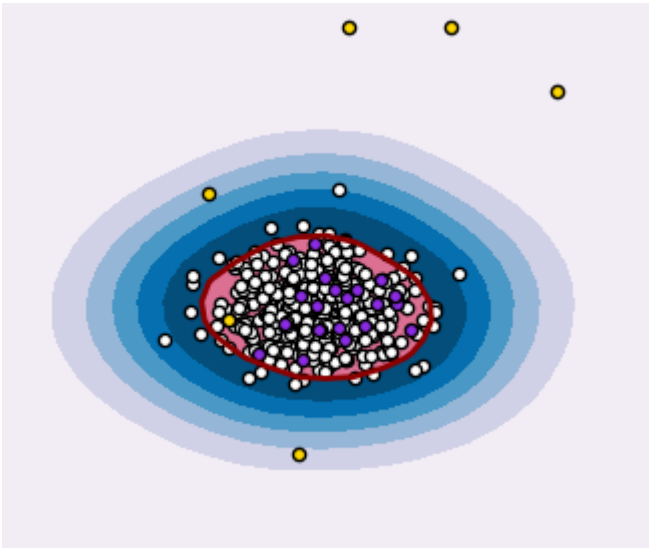
- Given a threshold value $\boldsymbol{\delta}$
- Decide whether a given data object $\boldsymbol{x}$ is an outlier

$$f_{\boldsymbol{\mu}^*,\boldsymbol{\sigma}^*}(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}} \frac{1}{|\boldsymbol{\sigma}^*|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}^*)^T \boldsymbol{\sigma}^{*-1}(\boldsymbol{x}-\boldsymbol{\mu}^*)\right\}$$

$f_{\boldsymbol{\mu}^*,\boldsymbol{\sigma}^*}(\boldsymbol{x}) \geq \boldsymbol{\delta}$ Normal data object

$f_{\boldsymbol{\mu}^*,\boldsymbol{\sigma}^*}(\boldsymbol{x}) < \boldsymbol{\delta}$ Outlier

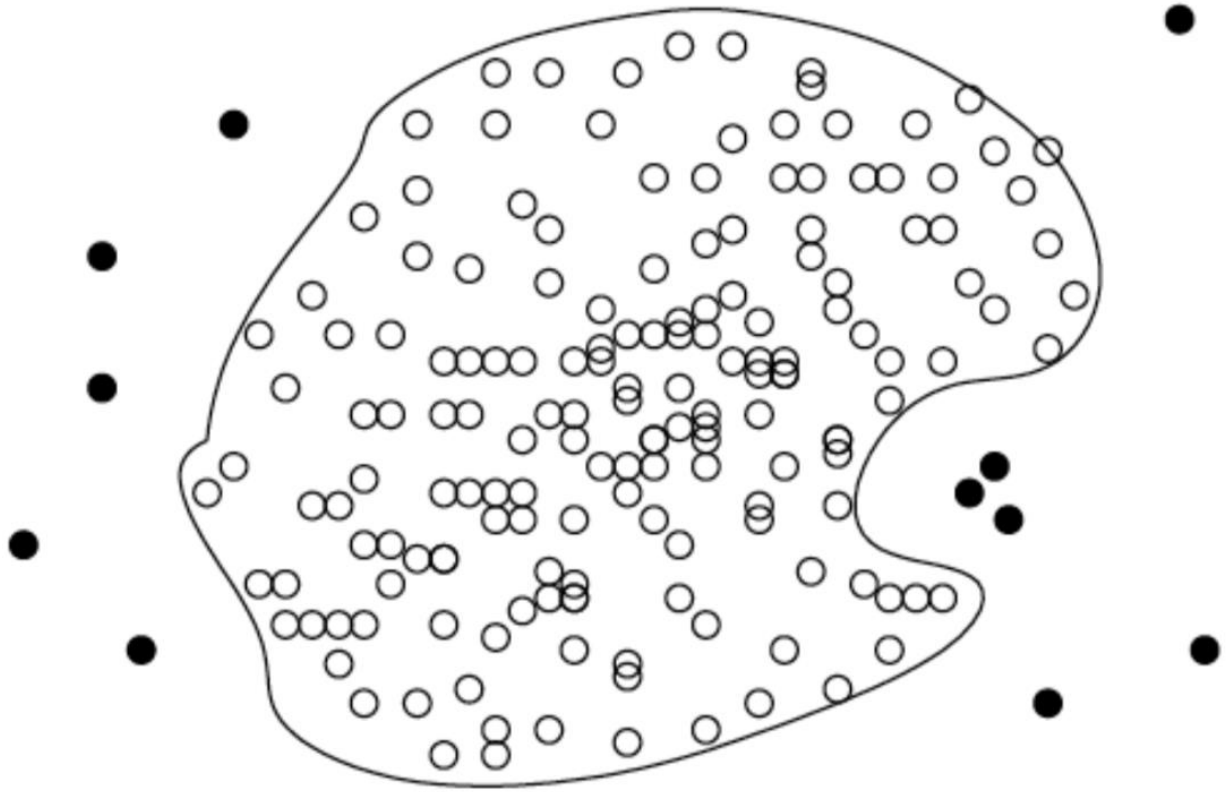
$\boldsymbol{\delta}$ determine the results



One-class SVM and Isolation Forest

单类支持向量机（One-class SVM）：基本思想

- **Learning the Boundary of the input data (i.e. normal class with large amount of obje**

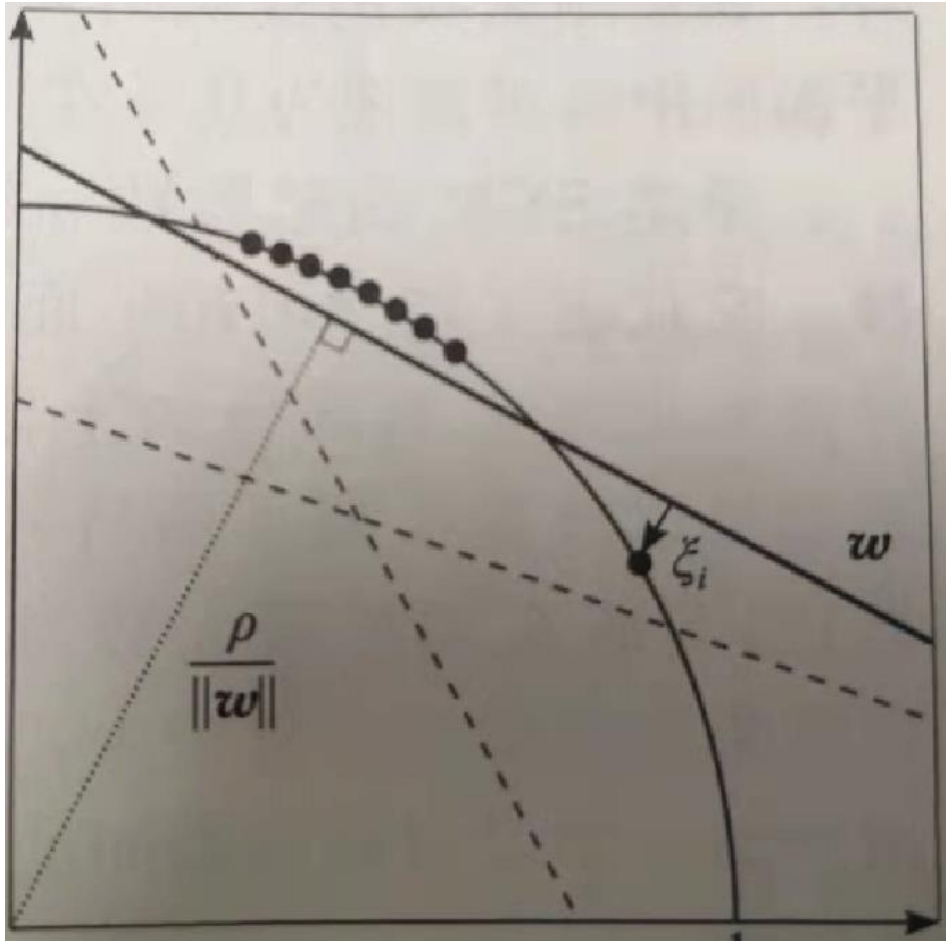


One-class SVM: *v-SVM*

- 如何定义第二个类?
  - $\emptyset(x)$: Projection to high-dimensional feature space
  - With Gaussian kernel:

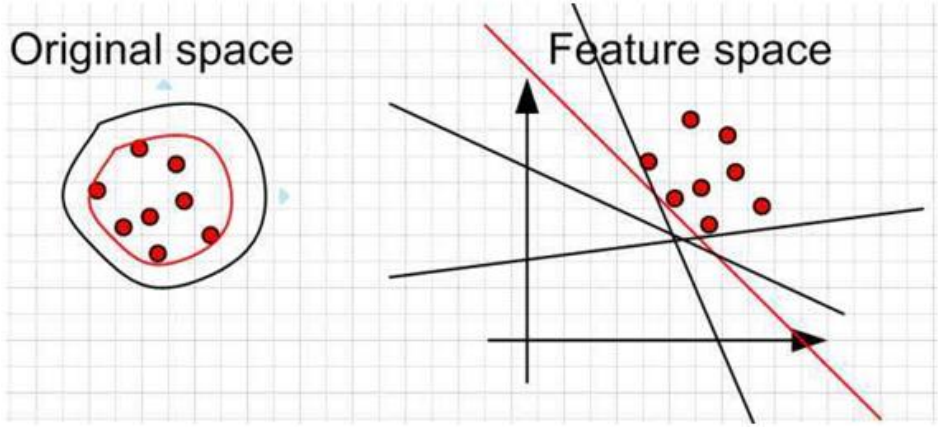
$$K(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right)$$

$$K(\boldsymbol{x}, \boldsymbol{x}) = <\emptyset(x), \emptyset(x)> = \|\emptyset(x)\|^2 = 1$$



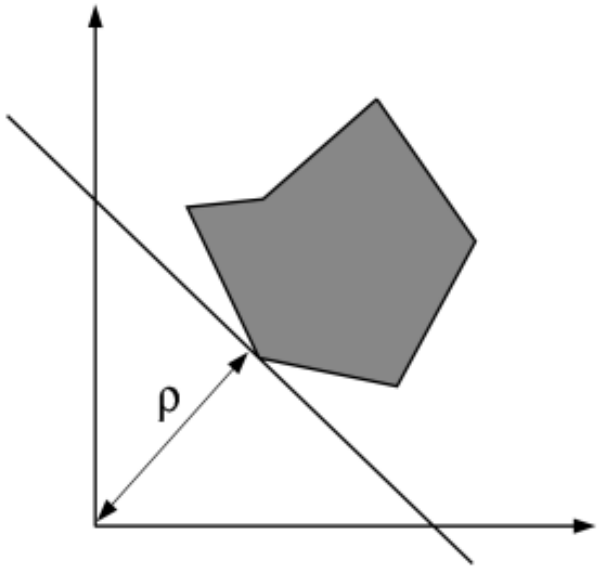
- Define the separating hyperplane:

$$w \cdot \emptyset(x) = \rho$$



- Outlier detection function:

$$f(x) = sgn(< w, \emptyset(x) > -\rho)$$



- 训练学习过程：

  - Learned model: $w \cdot \emptyset(x) = \rho$

  - Objective function:

$$\min_{\boldsymbol{w},\boldsymbol{\xi},\rho} \frac{\mathbf{1}}{\mathbf{2}}\|\boldsymbol{w}\|^2 + \frac{1}{\nu m}\sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} \xi_i - \rho$$

$$\text{subject to: } \langle \boldsymbol{w}, \emptyset(x_i) \rangle \geq \rho - \xi_i, \xi_i \geq 0$$

$$0 < \nu \leq 1$$

$$K(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right)$$

$$\boldsymbol{w}^* = \sum_{i=1}^{m} \lambda_i^* \phi(\boldsymbol{x}_i)$$

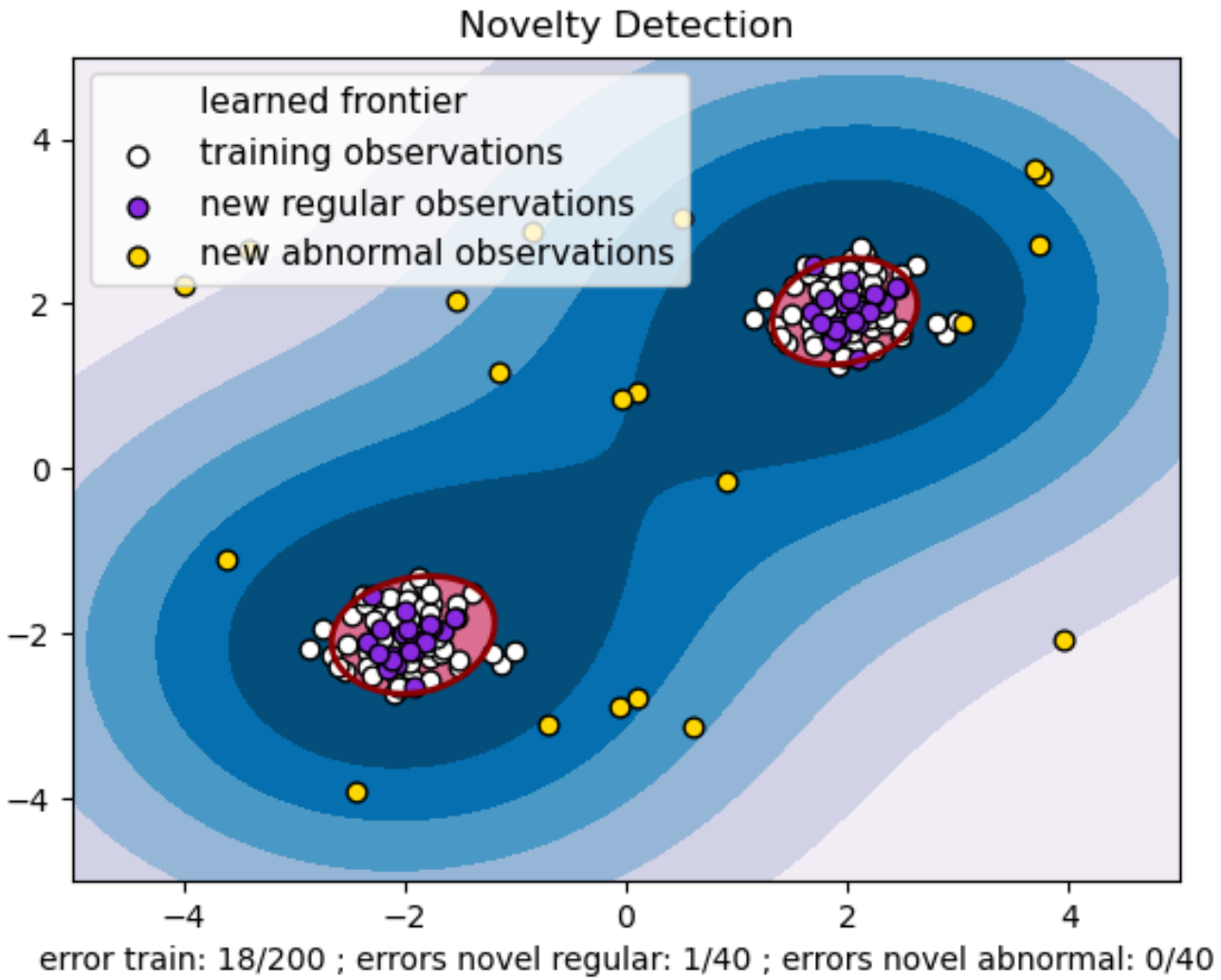
$$\rho^* = \sum_{i=1}^{m} \lambda_i^* \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_s)$$

$$= \sum_{i=1}^{m} \lambda_i^* K(\boldsymbol{x}_i, \boldsymbol{x}_s)$$

- 离群点检测过程：

$$f(x) = sgn(< w, \emptyset(x) > -\rho)$$

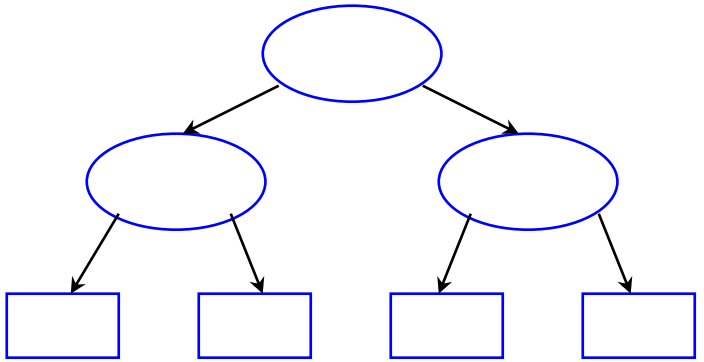
$$= sgn(\sum_{i=1}^{m} \lambda_i^* \phi(\boldsymbol{x}_i)\, \emptyset(x) - \rho) \quad = sgn(\sum_{i=1}^{m} \lambda_i^* K(\boldsymbol{x}_i, x) - \rho)$$

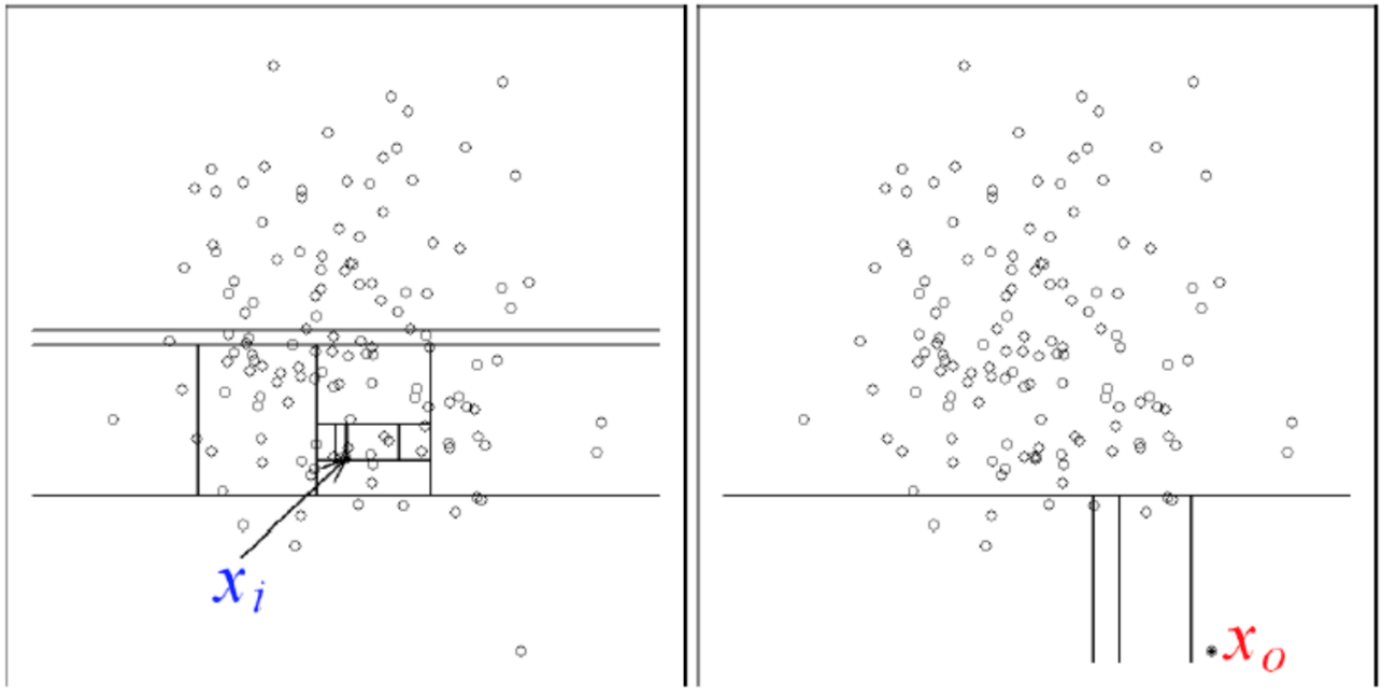


https://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html

基于孤立森林的离群点检测
(Isolation-based outlier detection)

- Outliers are few and different
- when randomly split the space into small region, an outlier is more likely to be *ISOLATED*

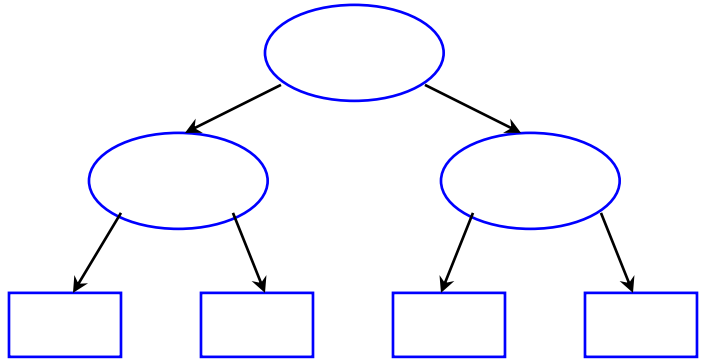




(a) Isolating x_i (b) Isolating x_o

- F. T. Liu, K. M. Ting and Z. H. Zhou, ***Isolation-based Anomaly Detection. ACM*** TKDD, 2011

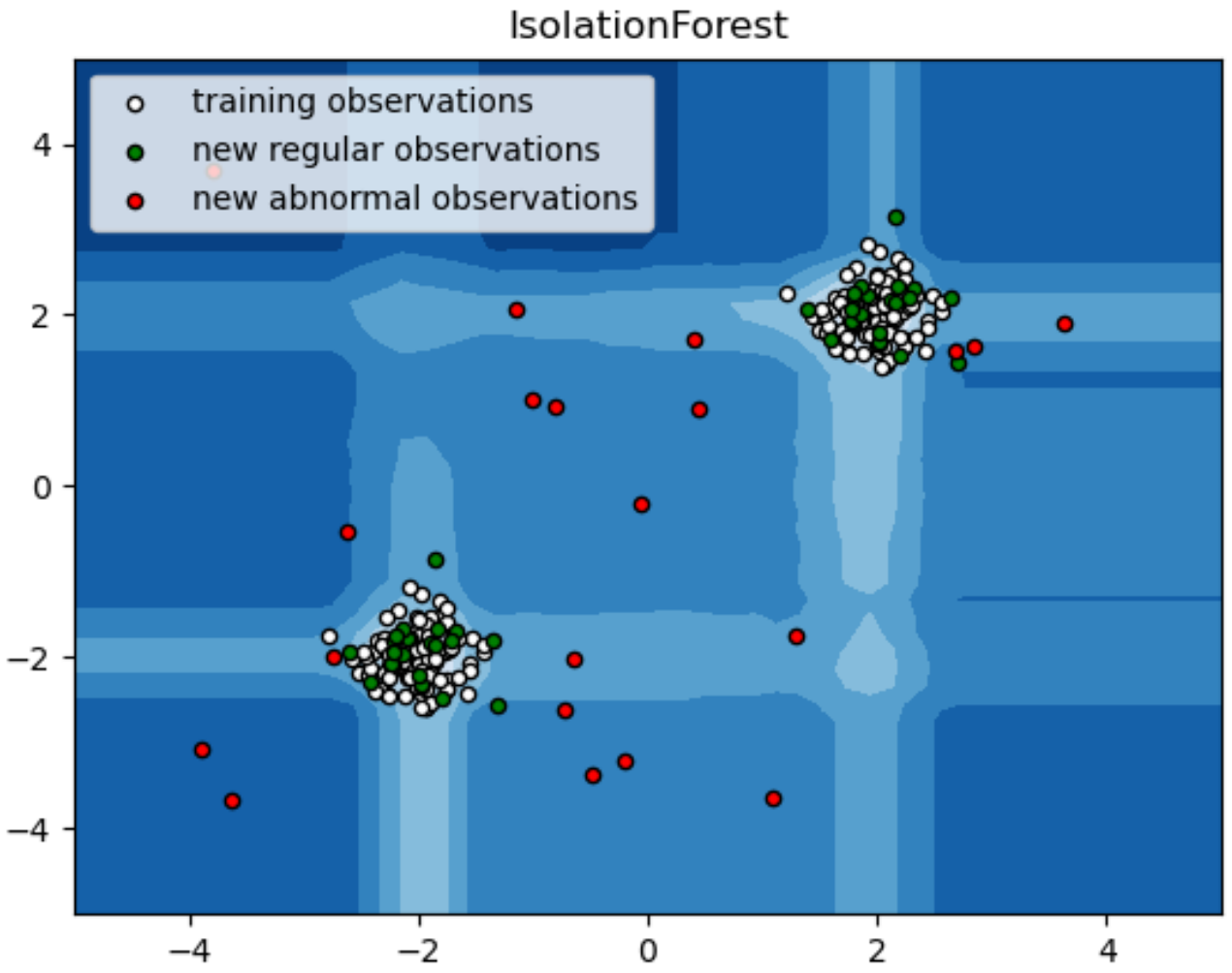
Key idea: Modeling "Isolation" using tree structure, and characterize the outlier suspiciousness with the path length from the root to the isolated object



- General steps:
  - Randomly subsample the original data
  - On each subsampled data, grow an *iTree* by
    - ✓ Randomly pick an attribute and a split value between min and max
    - ✓ Split the data into two subtrees
    - ✓ This process iterates until "isolation" is reached (no more points to be split or instances share the same value)
  - Compute outlier score by consulting the average path length from root to the isolated objects

Output of outlier detection

- Label
  - Each test instance is given a normal or anomaly label
- Score
  - Each test instance is assigned an anomaly score
    - ✓ Allows the output to be ranked
    - ✓ Requires an additional threshold parameter



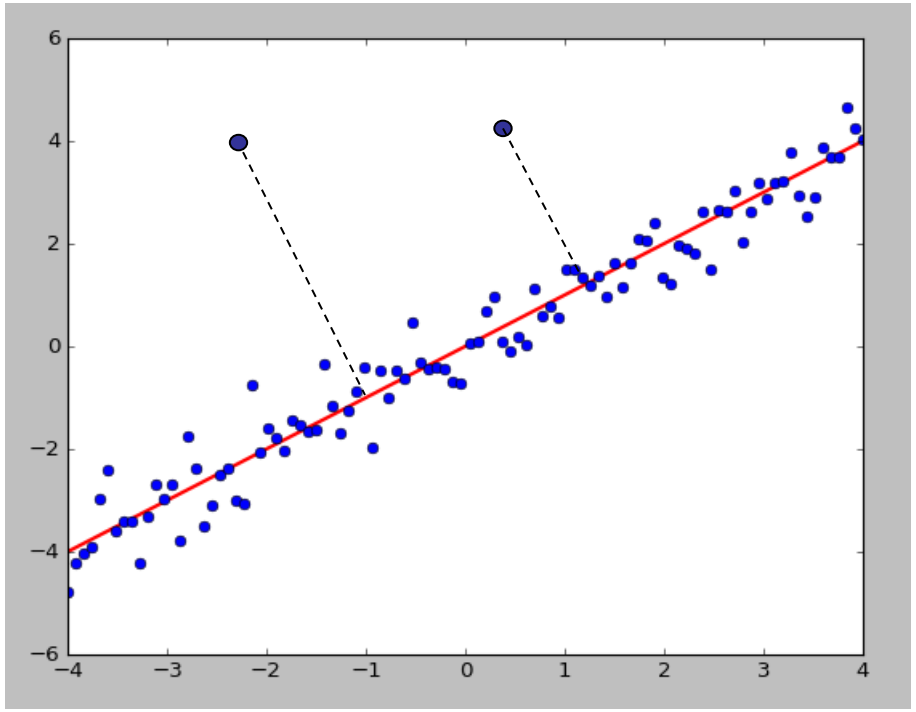
https://scikit-learn.org/stable/auto_examples/ensemble/plot_isolation_forest.html#sphx-glr-auto-examples-ensemble-plot-isolation-forest-py

基于重构的方法

- **基本思想：**
  - **假设正常类中存在隐藏结构，隐藏结构可以用低维特征空间进行表示。**
  - **正常样本在低维特征空间中可得到良好的近似。**
  - **通过将数据x_i投影到低维空间，得到低维表示y_i，再将y_i重新投影到原始空间，对样本进行重构，得到重构样本$\hat{x}_i$**

$$x_i \longrightarrow z_i \longrightarrow \hat{x}_i$$

$$x_i,\ \hat{x}_i \in R^n, z_i \in R^k,\ k \ll n$$

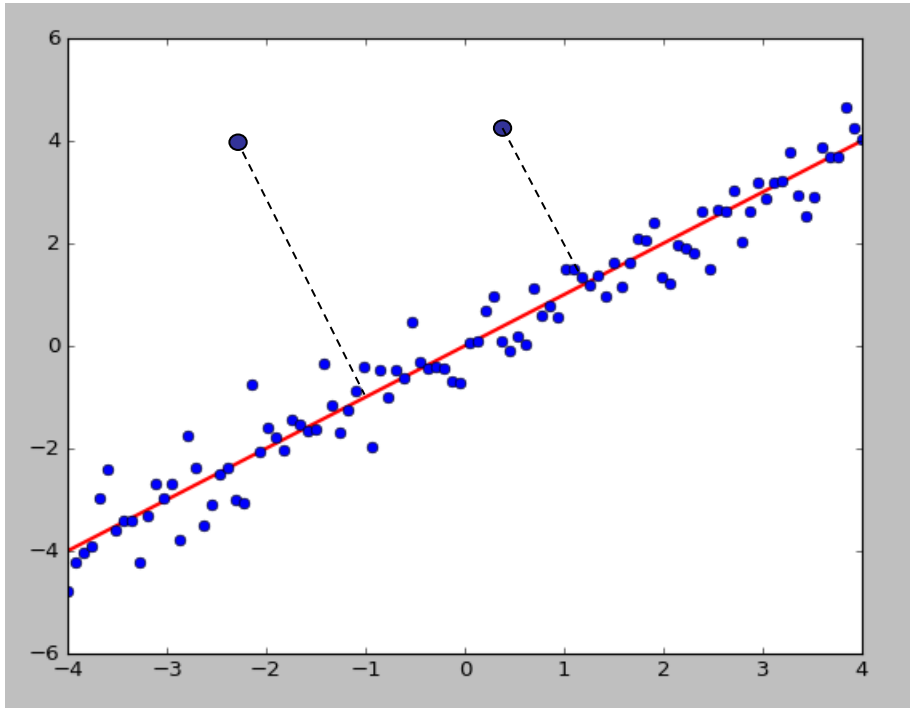


基于重构的方法

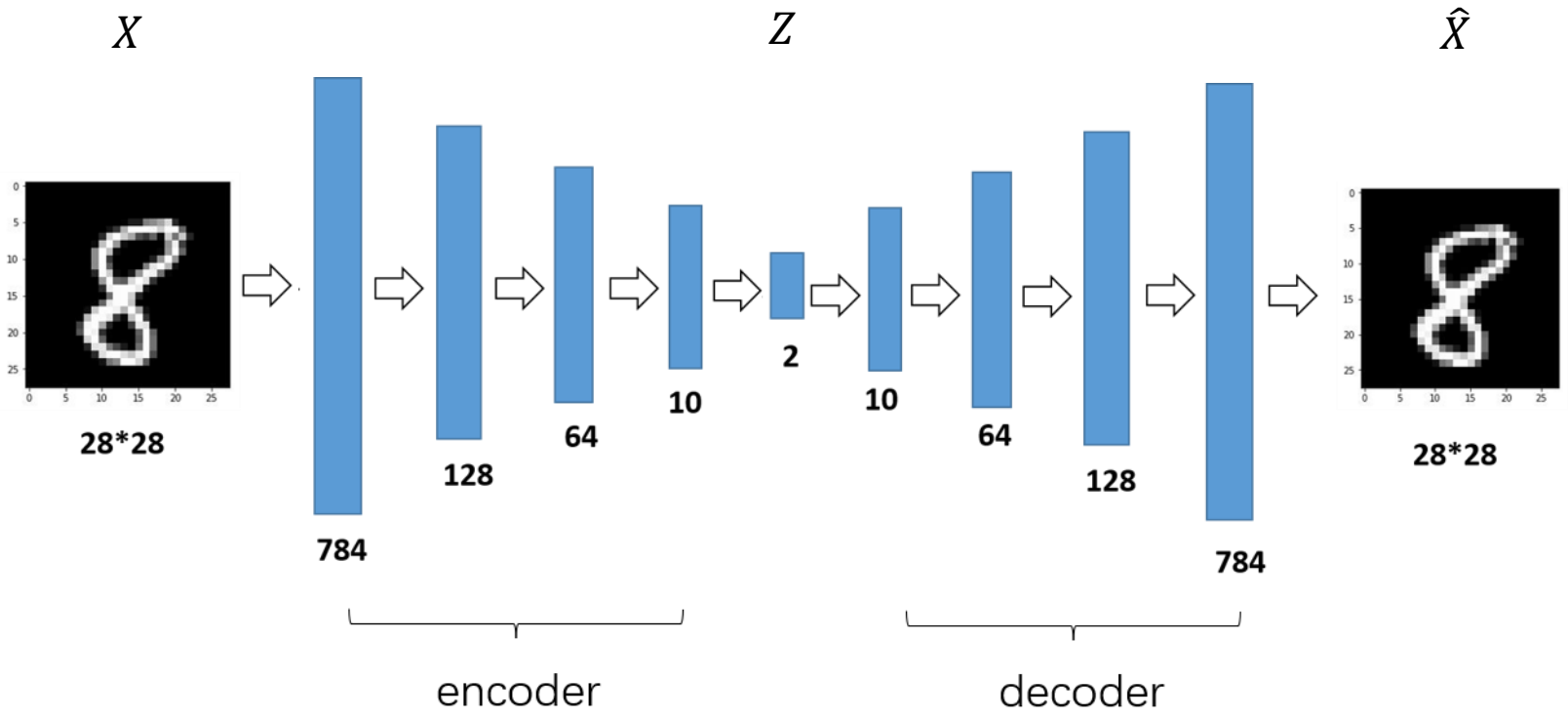
- **基本思想：**
  - **重构误差$e = \|x_i - \hat{x}_i\|^2$，根据重构误差判别异常。**
  - **对于正常样本，低维空间可以捕获其主要特征，y_i是x_i的良好近似，重构得到的$\hat{x}_i$和x_i接近，重构误差较低。**
  - **对于异常样本，不符合正常类的隐藏结构，低维空间无法获得其良好近似，即y_i无法近似表示x_i，重构得到$\hat{x}_i$和原样本x_i偏差较大，重构误差较大。**

$$x_i \longrightarrow z_i \longrightarrow \hat{x}_i$$

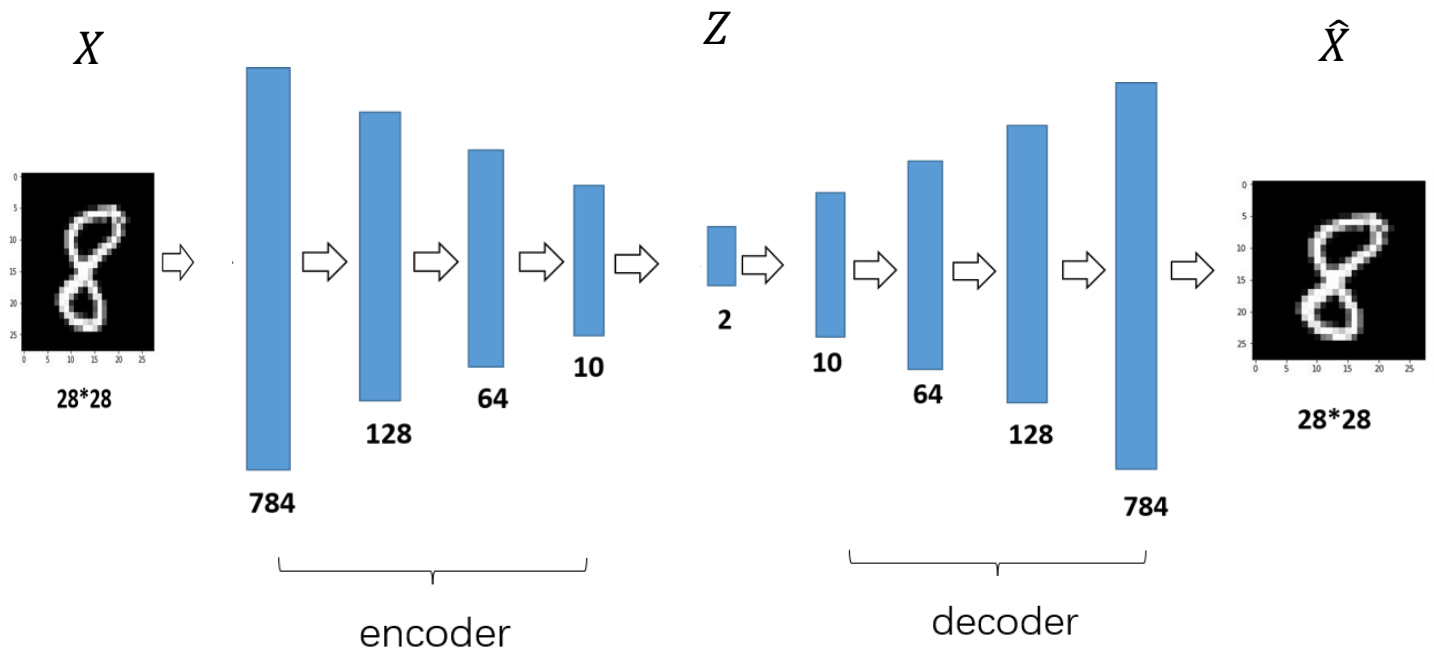
$$x_i,\ \hat{x}_i \in R^n, z_i \in R^k,\ k \ll n$$



基于自编码器的重构方法



训练目标



- encoder：map x to low dimensional z.

$$z = f_\varphi(x)$$

- decoder：reconstruct x based on z.

$$\hat{x} = g_\theta(z)$$

- Objective function：minimize reconstruction error.

$$L = \sum_{i=1}^{N} \|x_i - \tilde{x}_i\|_2^2$$

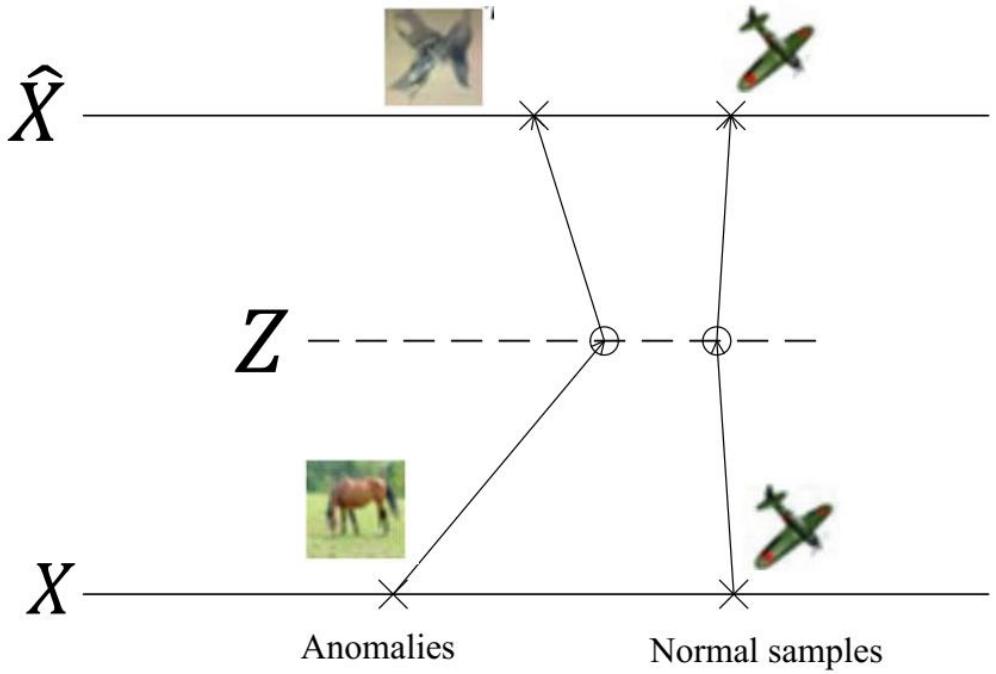
$$= \sum_{i=1}^{N} \left\|x_i - g_\theta(f_\varphi(x_i))\right\|_2^2$$

离群点检测

- Core idea： compare reconstruction errors with given threshold to detect anomalies.

error> threshold **anomaly**

error< threshold **normal**



致谢

- 部分图表、文字来自教材、互联网等，仅供公益性的学习参考，在此表示感谢！如有版权要求请联系：yym@hit.edu.cn，谢谢！