

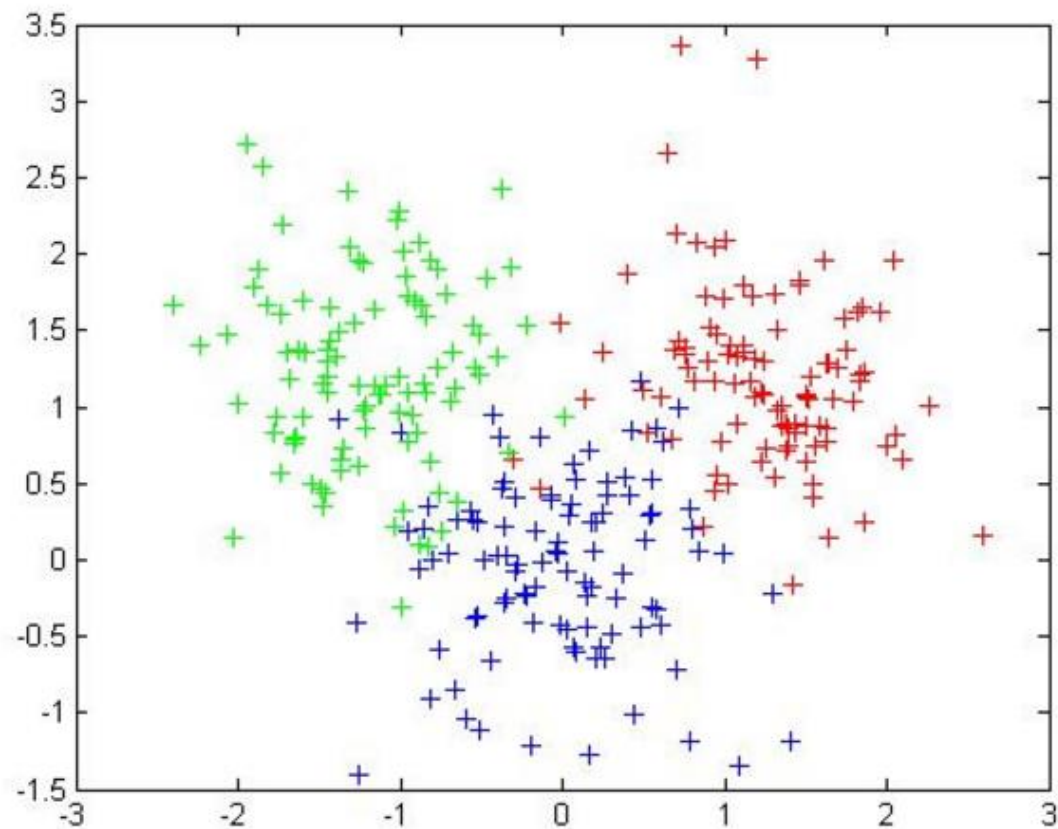
# K-MEANS算法

✓ 聚类概念：

✎ 无监督问题：我们手里没有标签了

✎ 聚类：相似的东西分到一组

✎ 难点：如何评估，如何调参



# K-MEANS算法

✓ 基本概念：

✎ 要得到簇的个数，需要指定K值

✎ 质心：均值，即向量各维取平均即可

✎ 距离的度量：常用欧几里得距离和余弦相似度（先标准化）

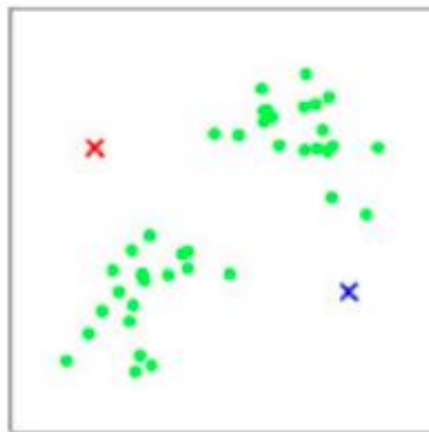
✎ 优化目标：
$$\min \sum_{i=1}^K \sum_{x \in C_i} \text{dist}(c_i, x)^2$$

# K-MEANS算法

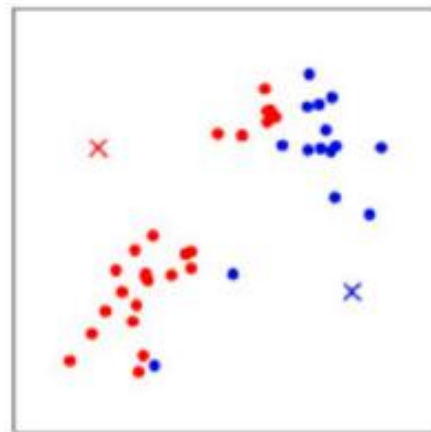
✓ 工作流程：



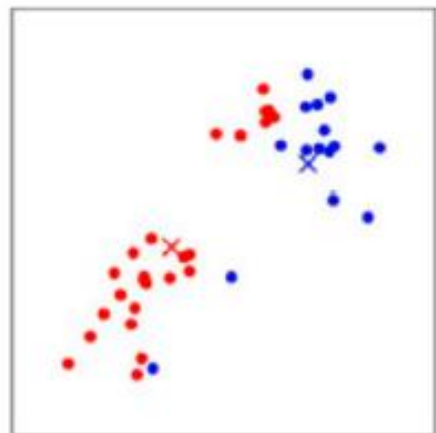
(a)



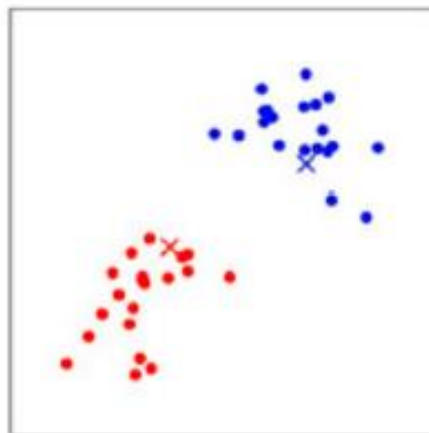
(b)



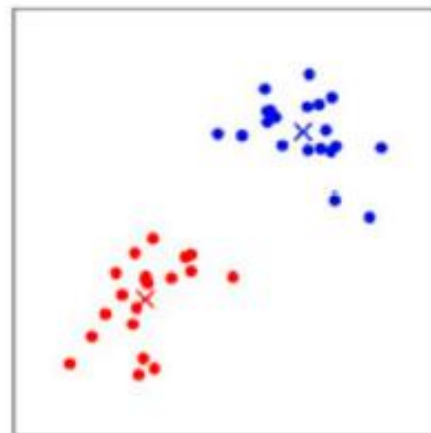
(c)



(d)



(e)



(f)

# K-MEANS算法

✓ 优势：

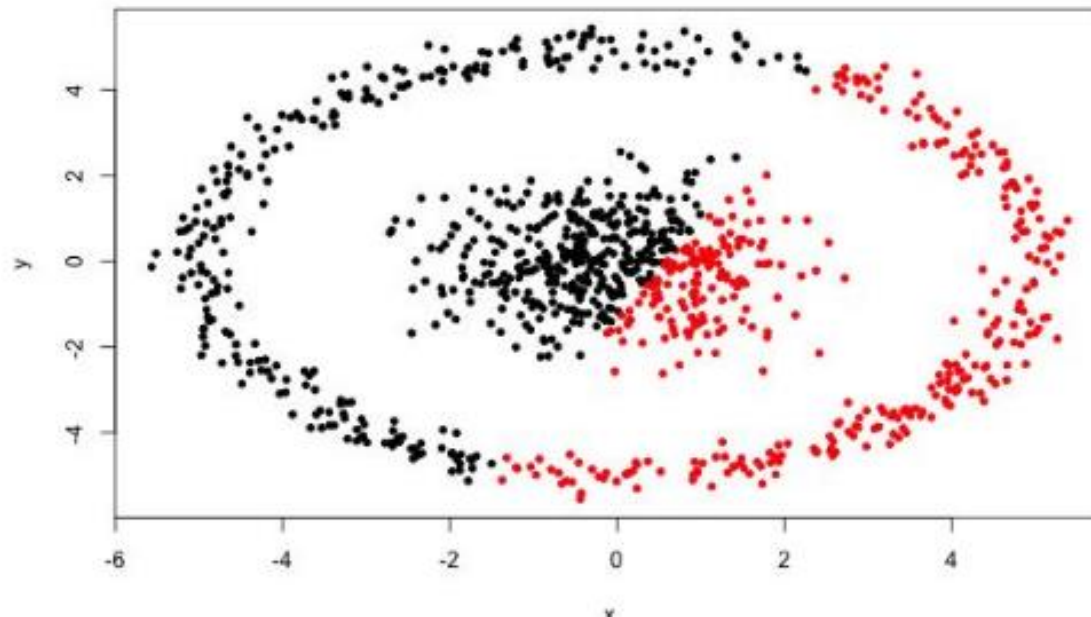
✎ 简单，快速，适合常规数据集

✓ 劣势：

✎ K值难确定

✎ 复杂度与样本呈线性关系

✎ 很难发现任意形状的簇



# DBSCAN算法

- ✓ 基本概念：（ Density-Based Spatial Clustering of Applications with Noise ）
- ✎ 核心对象：若某个点的密度达到算法设定的阈值则其为核心点。  
（ 即  $r$  邻域内点的数量不小于  $\text{minPts}$  ）
- ✎  $\epsilon$ -邻域的距离阈值：设定的半径 $r$
- ✎ 直接密度可达：若某点 $p$ 在点 $q$ 的  $r$  邻域内，且 $q$ 是核心点则 $p$ - $q$ 直接密度可达。
- ✎ 密度可达：若有一个点的序列 $q_0$ 、 $q_1$ 、... $q_k$ ，对任意 $q_i$ - $q_{i-1}$ 是直接密度可达的，则称从 $q_0$ 到 $q_k$ 密度可达，这实际上是直接密度可达的“传播”。

# DBSCAN算法

## ✓ 基本概念：

- ✎ 密度相连：若从某核心点 $p$ 出发，点 $q$ 和点 $k$ 都是密度可达的，则称点 $q$ 和点 $k$ 是密度相连的。
- ✎ 边界点：属于某一个类的非核心点，不能发展下线了
- ✎ 直接密度可达：若某点 $p$ 在点 $q$ 的  $r$  邻域内，且 $q$ 是核心点则 $p$ - $q$ 直接密度可达。
- ✎ 噪声点：不属于任何一个类簇的点，从任何一个核心点出发都是密度不可达的

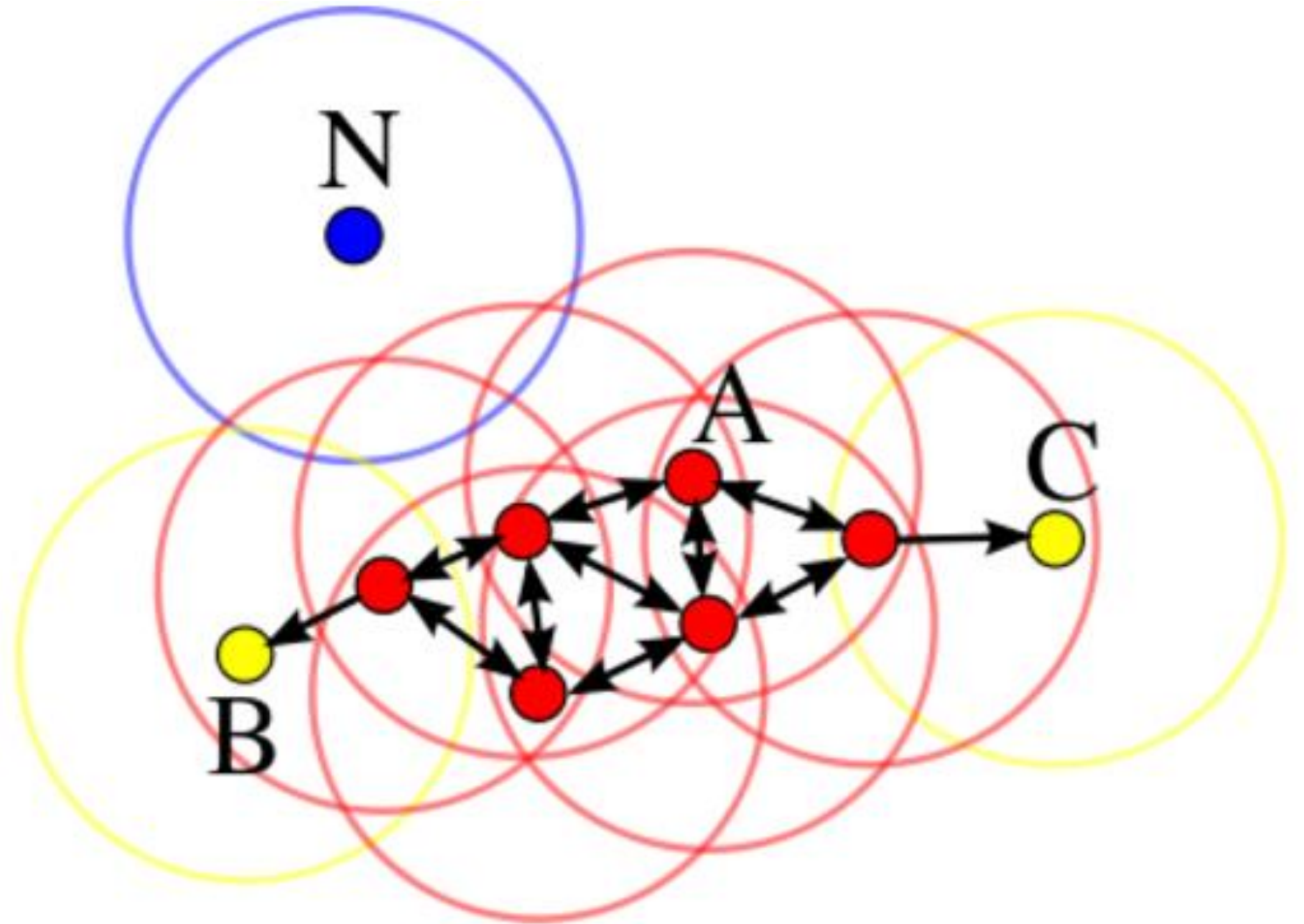
# DBSCAN算法

✓ 基本概念：

✎ A：核心对象

✎ B,C：边界点

✎ N：离群点



# DBSCAN算法

✓ 工作流程：

✎ 参数D：输入数据集

✎ 参数 $\epsilon$ ：指定半径

✎ MinPts：密度阈值

1. 标记所有对象为 unvisited;
2. Do
3. 随机选择一个 unvisited 对象 p;
4. 标记 p 为 visited;
5. If p 的  $\epsilon$ -领域至少有 MinPts 个对象
6.     创建一个新簇 C, 并把 p 添加到 C;
7.     令 N 为 p 的  $\epsilon$ -领域 中的对象集合
8.     For N 中每个点  $p'$
9.         If  $p'$  是 unvisited;
10.             标记  $p'$  为 visited;
11.             If  $p'$  的  $\epsilon$ -领域至少有 MinPts 个对象, 把这些对象添加到 N;
12.             如果  $p'$  还不是任何簇的成员, 把  $p'$  添加到 C;
13.     End for;
14.     输出 C;
15. Else 标记 p 为噪声;
16. Until 没有标记为 unvisited 的对象;



# DBSCAN算法

✓ 参数选择：

✎ 半径 $\epsilon$ ，可以根据K距离来设定：找突变点

K距离：给定数据集 $P=\{p(i); i=0,1,...n\}$ ，计算点 $P(i)$ 到集合D的子集S中所有点之间的距离，距离按照从小到大的顺序排序， $d(k)$ 就被称为k-距离。

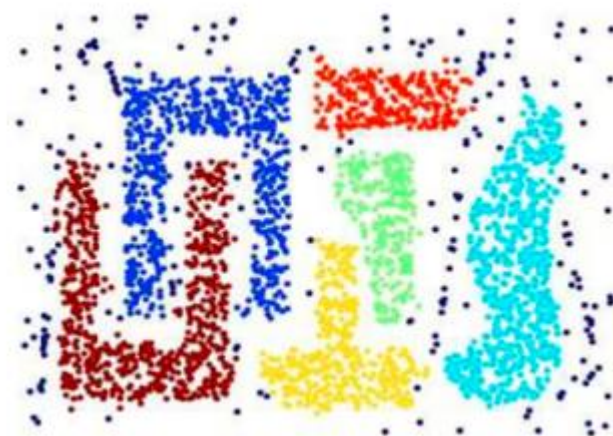
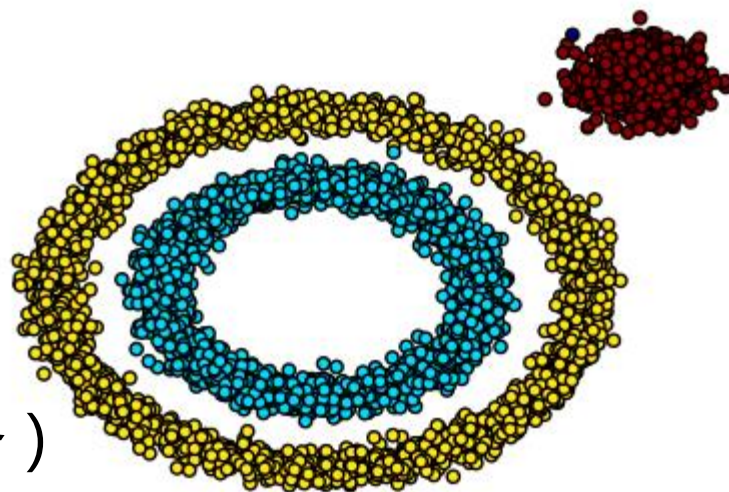
✎ MinPts：k-距离中k的值，一般取的小一些，多次尝试

可视化：<https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/>  
<https://www.naftaliharris.com/blog/visualizing-k-means-clustering/>

# DBSCAN算法

✓ 优势：

- ✎ 不需要指定簇个数
- ✎ 可以发现任意形状的簇
- ✎ 擅长找到离群点（检测任务）
- ✎ 两个参数就够了



✓ 劣势：

<https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/>

- ✎ 高维数据有些困难（可以做降维）
- ✎ 参数难以选择（参数对结果的影响非常大）
- ✎ Sklearn中效率很慢（数据削减策略）

