

16

Density-Based Clustering

密度聚类

利用数据分布紧密程度聚类



实验是科学向自然提出的问题，测量是对自然回答的记录。

An experiment is a question which science poses to Nature, and a measurement is the recording of Nature's answer.

—— 马克斯·普朗克 (Max Planck) | 德国物理学家，量子力学的创始人 | 1858 ~ 1947



- ◀ `itertools.cycle()` 把一组数据循环取出
- ◀ `itertools.islice()` 返回一个迭代器
- ◀ `numpy.random.seed()` 设置随机数种子可以使每一次生成随机数据的时候结果相同
- ◀ `sklearn.cluster.DBSCAN()` DBSCAN 聚类函数
- ◀ `sklearn.cluster.OPTICS()` OPTICS 聚类函数
- ◀ `sklearn.datasets.make_circles()` 创建环形样本数据
- ◀ `sklearn.preprocessing.StandardScaler().fit_transform()` 标准化数据；通过减去均值然后除以标准差，处理后数据符合标准正态分布

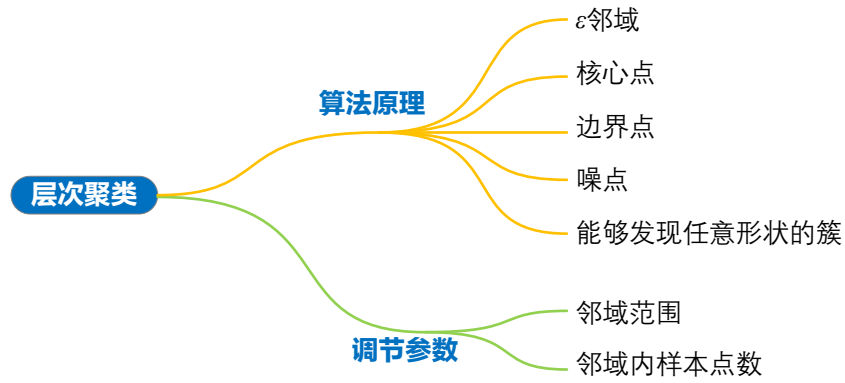
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com



16.1 DBSCAN 聚类

密度聚类是一种基于数据点密度的聚类方法，其核心思想是将高密度区域作为聚类中心，并将低密度区域作为聚类边界。常用的密度聚类算法有 DBSCAN、OPTICS、DENCLUE 等。

DBSCAN 通过设定邻域半径和最小密度等参数，将具有足够密度的数据点聚成一个簇；OPTICS 在 DBSCAN 的基础上，通过建立可达距离图来优化聚类结果；DENCLUE 则采用高斯核函数来建模数据点的密度，通过求解梯度的方式来寻找密度峰值，进而进行聚类。密度聚类方法对于数据分布的形态没有特殊要求，对于噪声和离群点的鲁棒性较强，具有广泛的应用价值。

DBSCAN 聚类算法全称为 Density-Based Spatial Clustering of Applications with Noise，它是一种基于密度的聚类方法，是本章要重点介绍的算法。为了方便大家理解 DBSCAN 聚类算法，下面打个比方。

原理

如图 1 所示，限定距离范围内（即圆圈圈定领域），粉丝超过一定数量的点就是 UP 主（头顶皇冠者）；DBSCAN 聚类算法和核心是，如果任意两个 UP 主互粉（在对方的圆圈范围之内），则两个 UP 主及各自粉丝可以被划分为一簇。

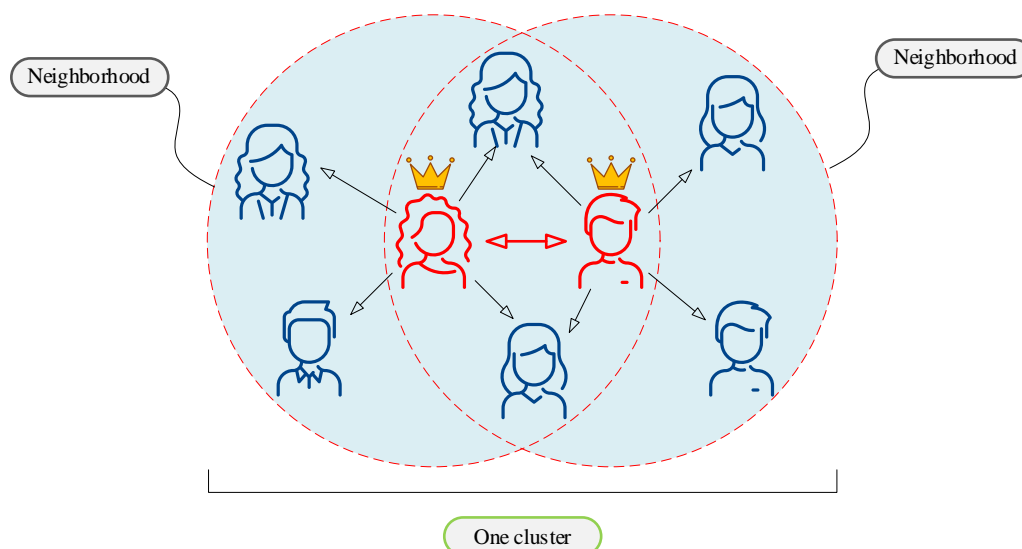


图 1. DBSCAN 算法原理

几个概念

下面介绍 DBSCAN 涉及到的几个概念。

ϵ 邻域 (ϵ neighborhood, epsilon neighborhood) ϵ_{sp} 限定领域范围, ϵ_{sp} 对应图 1 中的圆圈半径。准确地说, ϵ 邻域指的是以某样本数据点为中心、 ϵ_{sp} 为半径的区域。

以空间某点为中心, ϵ_{sp} 为半径邻域内包含至少 $\min_samples$ 数量的数据点, 则称该点为**核心点** (core point), 即前文所说的 UP 主。

核心点 ϵ 邻域内的点, 被称之为**边界点** (border point)。核心点相当于图 1 中 UP 主; 边界点, 相当于粉丝。特别需要读者注意的是, $\min_samples$ 为核心点和边界点数量之和。

样本数据点可以是核心点, 也可以是边界点, 甚至身兼两者角色; 如果数据点既不是核心点, 也不是边界点, 该数据点被称作**噪点** (noise point), 即**离群数据** (outlier)。

聚类

图 2 给出平面内 8 个样本数据点; 以每个数据点为中心, ϵ 为半径扫描整个平面, 且定义 $\min_samples = 4$ 。

发现只有样本点 $x^{(5)}$ 的 ϵ 邻域内有 4 个样本点 (包括自 $x^{(5)}$ 身); 因此, x_5 为核心点, $x^{(2)}$ 、 $x^{(4)}$ 和 $x^{(7)}$ 为边界点, 剩余其他数据点为噪点。

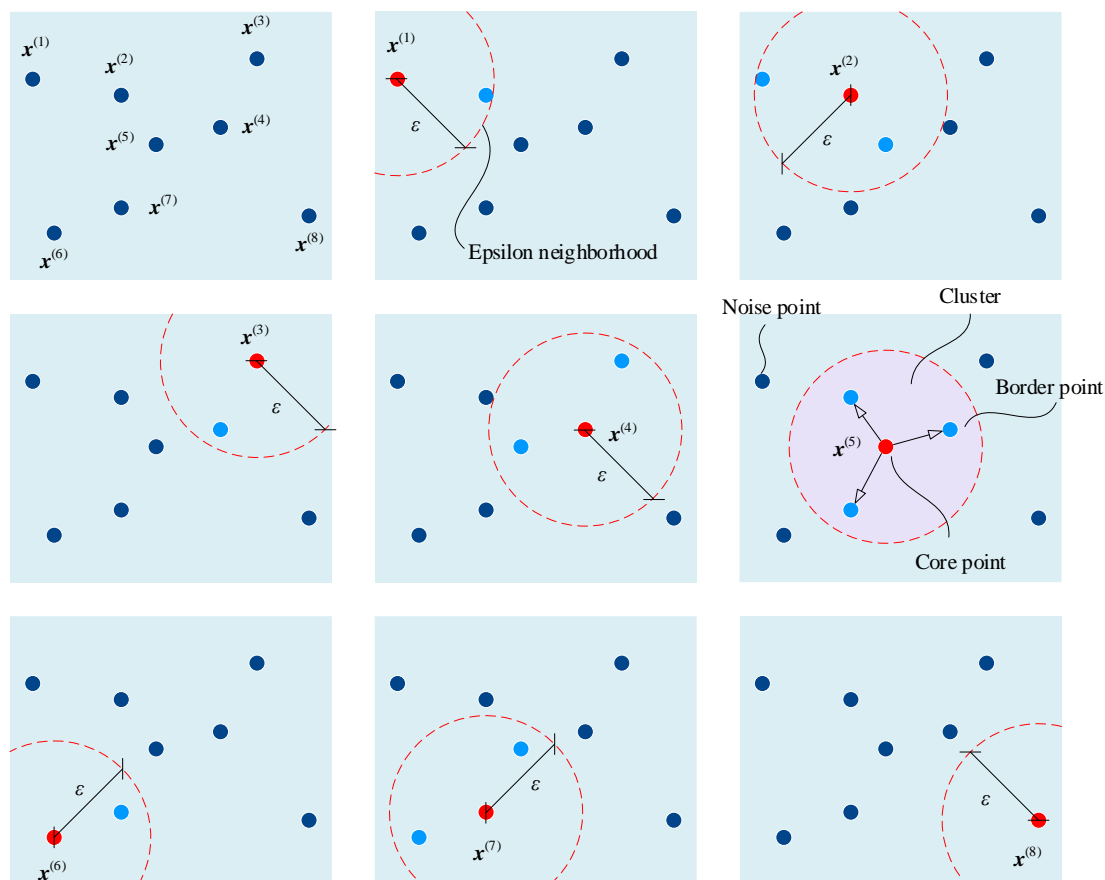


图 2. DBSCAN 算法扫描 8 个样本数据点

如图 3 所示，通过 DBSCAN 算法，空间数据被分为 3 簇。图 3 中，红色数据点为核心点（即 UP 主）。UP 主的最低要求是在以自己为中心的 ϵ 邻域内包括含自己在内有 4 名成员；浅蓝色数据点为边界点，深蓝色为噪点。 C_1 自成一簇；三个 UP 主互粉，三个 ϵ 邻域相互连接，构成 C_2 ；两个 UP 主互粉，两个 ϵ 邻域相互连接，构造图 3 中所示 C_3 。

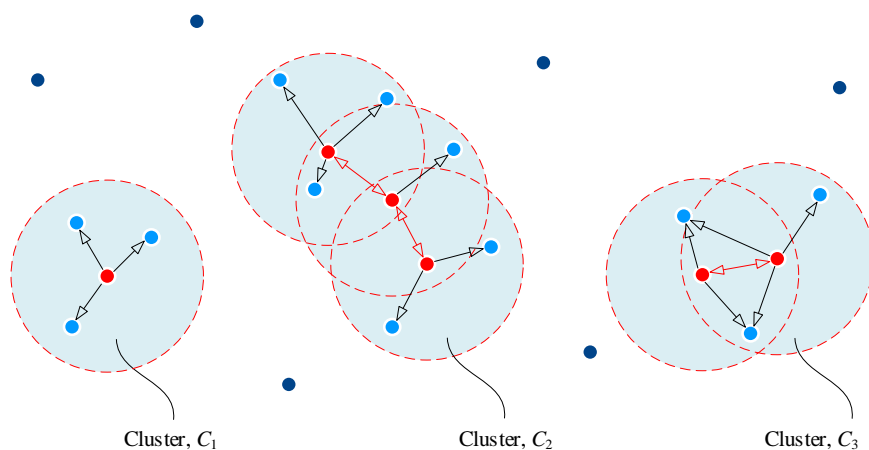


图 3. 通过 DBSCAN 算法，数据被分为 3 簇

16.2 调节参数

邻域范围

eps 控制邻域范围大小。**eps** 值选取过大，会导致整个数据集被分为一簇；但是 **eps** 取值过小，会导致簇过多且分散，并且标记过多噪音点。

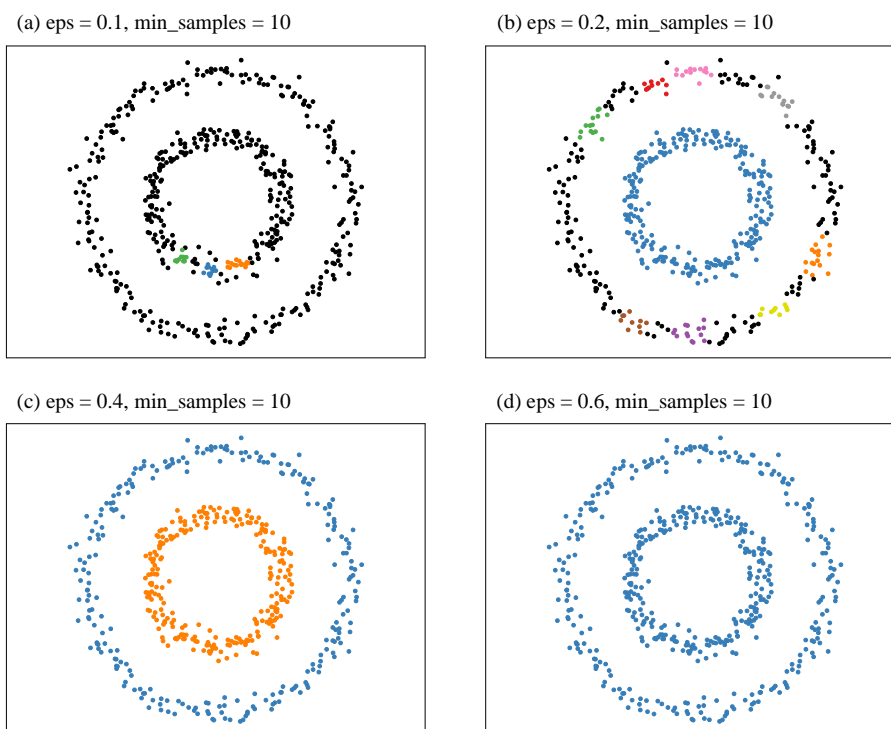
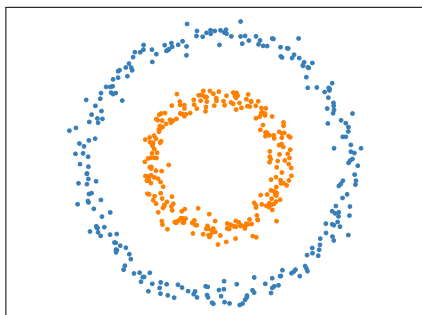
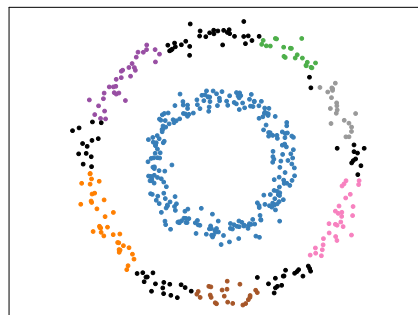
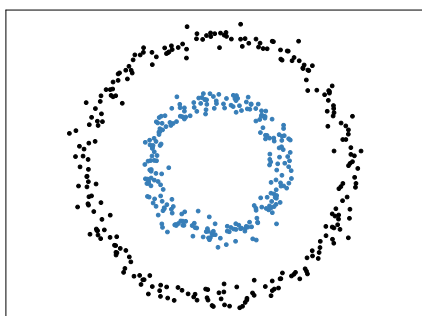
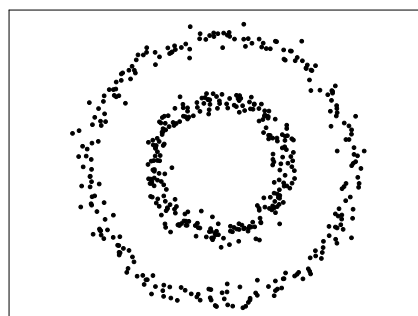
图 4. ϵ 对聚类结果影响

图 4 (a) 所示，当 $\epsilon = 0.1$ 时，环形样本数据多数被标记为噪音（黑色点）。当 ϵ 增大到 0.2 时，被标记为噪音点减少，且小环被划分为一簇（蓝色点），如图 4 (b) 所示。当 $\epsilon = 0.4$ 时，环形样本数据被正确地分类为两簇，如图 4 (c) 所示。当 ϵ 增大到 0.6 时，所有样本数据被划分为一簇。请读者注意， ϵ 邻域半径 ϵ 这个距离，未必是欧氏距离。

邻域内样本点数

min_samples 调节 DBSCAN 算法对噪声的容忍度；当数据噪音过大时，应该适当提高 min_samples 。

k 均值和 GMM 聚类算法需要预先声明聚类数量；但是，DBSCAN 则不需要。DBSCAN 聚类不需要预设分布类型，不受数据分布影响，且可以分辨离群数据。但是，DBSCAN 算法对 ϵ 和 min_samples 这两个初始参数都很敏感；协同调节 ϵ 和 min_samples 两个参数显得非常重要。

(a) $\text{eps} = 0.4, \text{min_samples} = 10$ (b) $\text{eps} = 0.4, \text{min_samples} = 20$ (c) $\text{eps} = 0.4, \text{min_samples} = 30$ (d) $\text{eps} = 0.4, \text{min_samples} = 40$ 图 5. min_samples 对聚类结果影响

代码 Bk7_Ch15_01.py 绘制图 4、图 5。



DBSCAN 是一种基于密度的聚类算法，其特点是可以自动识别出任意形状的簇，并将离群点视为噪声数据。DBSCAN 将密度定义为在给定半径内的数据点数量，利用这一度量将数据点分为核心点、边界点和噪声点三类。

在聚类过程中，DBSCAN 通过不断扩展核心点的密度直到达到最大密度，将核心点和边界点划分到同一簇中。优点是不需要事先设定聚类数量，鲁棒性强，可以处理不同形状、大小和密度的簇。缺点是对于密度分布较为均匀的数据集，可能出现聚类失效的情况。



OPTICS (Ordering Points To Identify the Clustering Structure) 聚类算法和 DSCAN 非常相似。不同的是，需要用户输入 `eps` 和 `min_samples` 两个参数；而 OPTICS 虽然也需要输入这两个参数，但是对 `eps` 不敏感。请读者自行学习下例。

https://scikit-learn.org/stable/auto_examples/cluster/plot_optics.html