浅谈无监督学习模型(K-MEANS, DBSCAN)

一、无监督学习模型（unsupervised learning models）简介

所谓有监督模型，简单来说就是对于我们的数据样本，数据本身有个标签，比如它是属于好瓜还是坏瓜。相反，无监督问题就是问题的本身没有标签。我们在处理将相似的东西分为一组时，大部分用的就是无监督的学习模型，将相似的东西分到一组的过程也称为聚类。

处理聚类问题有一些经典的算法，其中 K-MEANS 和 DBSCAN 算法最为人们熟知。

二、K-MEANS 算法

1.基本概念

（1）应用 K-MEANS 算法，要得到簇（分为几类，每一类叫一簇）的个数，需要指定 K 值，即我们想将数据分为几类。

（2）各簇的质心：均值，即向量各维取平均即可

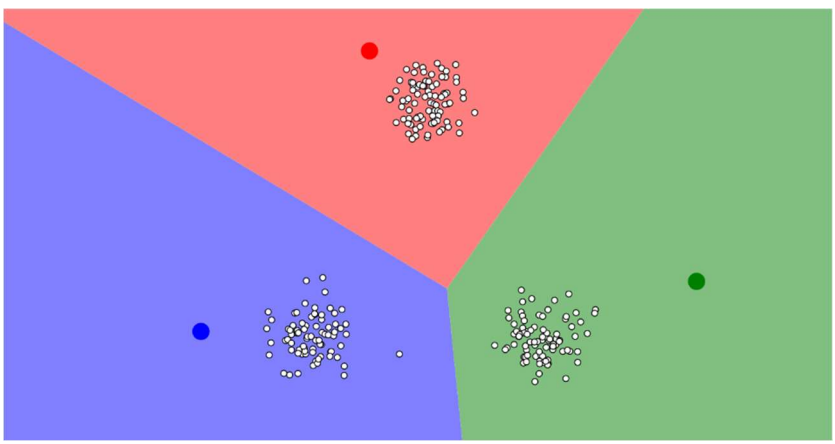
（3）距离的度量（具体操作中会再介绍）：常用欧几里得距离和余弦相似度（先标准化）

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

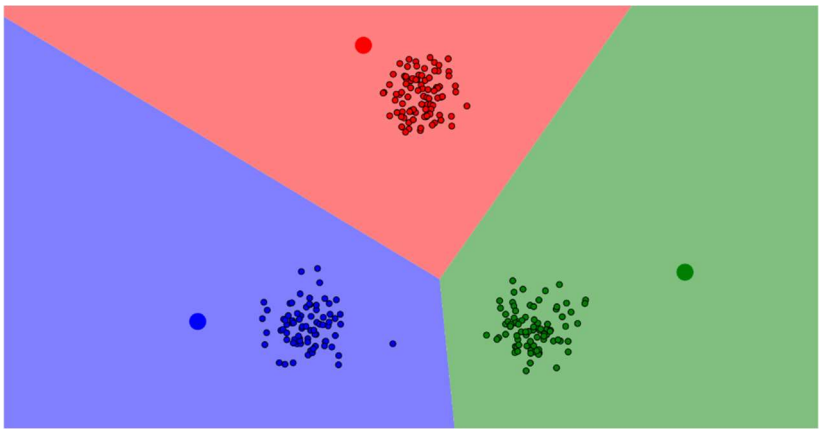
其中 c 是质心，x 是各数据点。

2.工作流程

（1）指定 k 的值，选取 k 个基本点

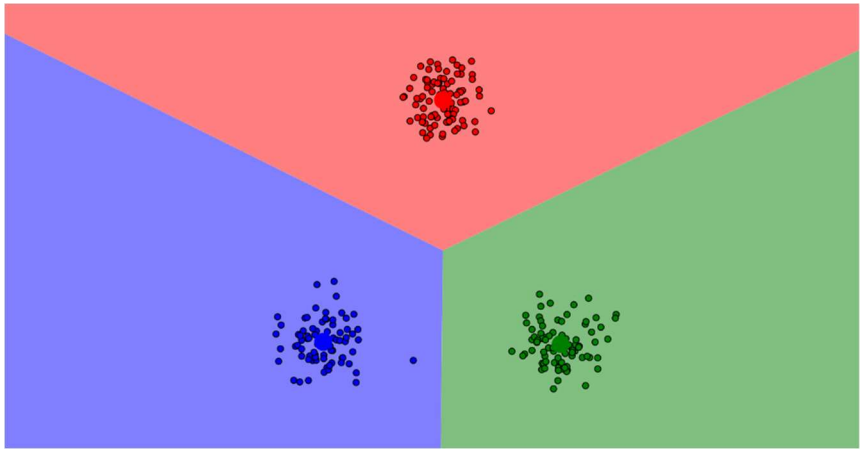


（2）计算所有点到三个基本点的距离，假如某个点离红色基本点最近，那么将它归为红色簇。处理所有点得到：



可以看到所有的点都有各自的簇组，目前看来分的不错。

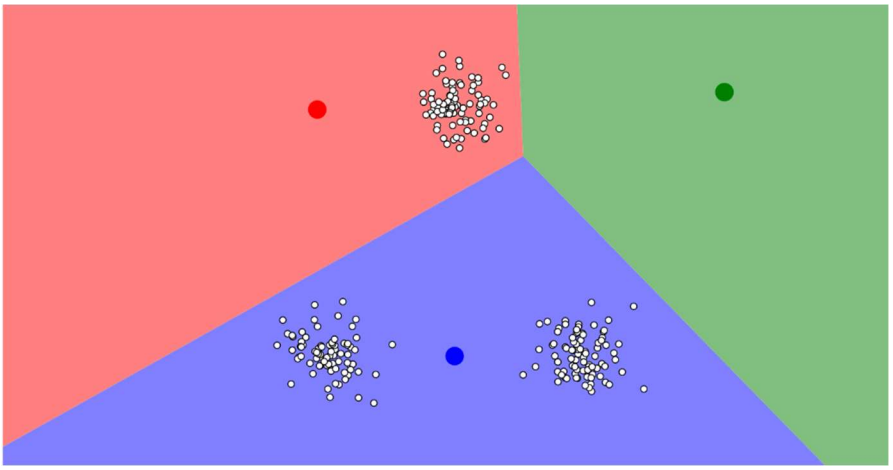
（3）更新质心的位置，即计算各簇的向量中心



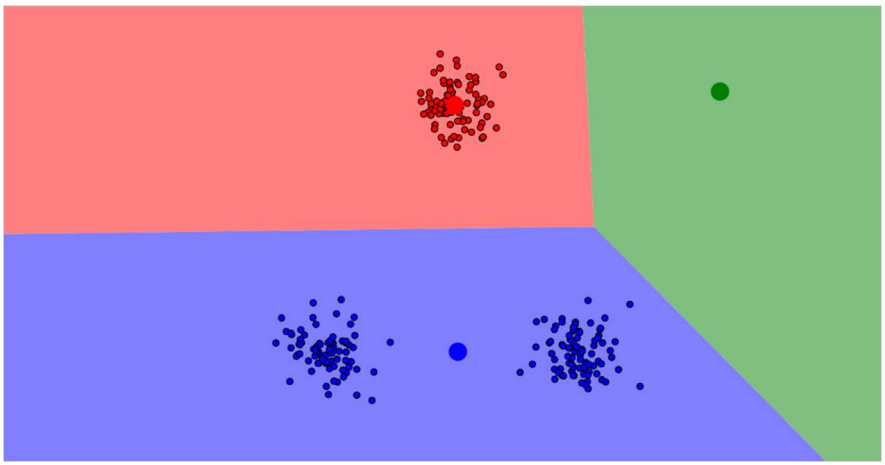
重复（1）~（3）的步骤，直到质心的位置几乎不再变化

就目前的情况来看，迭代一次好像就已经得到了结果，原因是我们的基本点选的还不错，如果基本点误差比较大，就需要更长时间的迭代。但这还不是最糟的，如果我们的基本点选的不好，可能怎么迭代都得不到我们心仪的结果。示例如下：

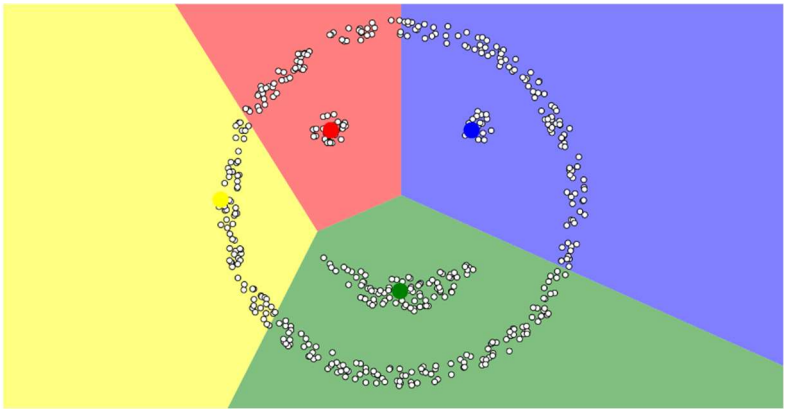
我们还是同样选择三个基本点：



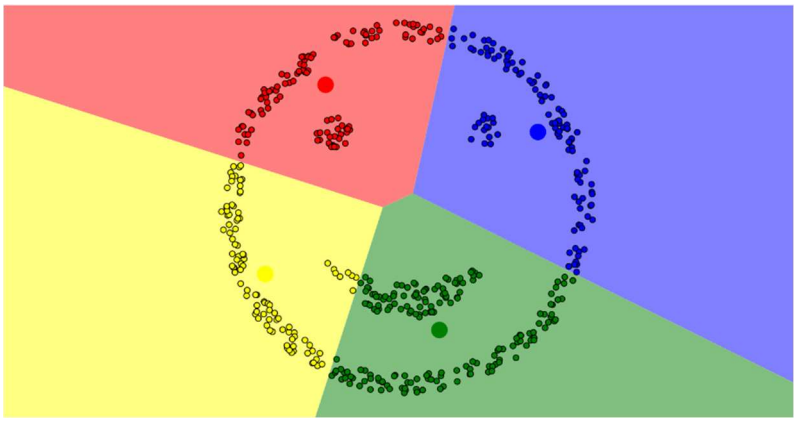
经过很多次（1）~(3)的迭代，我们得到：



可以看到，结果与我们期待的相去甚远，甚至它根本就是不对的，不过这已经是 K-MEANS 算法基于这三个点能做出的最好结果了。
再换个样本呢，我们看看这个数据集：



我们尽我们的常规思路将其分为四份（因为它人眼看去就应该被分为四份），让我们看看这次 K-MEANS 分类的结果：



显然，不太行。

3.K-MEANS 算法的优劣

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

三、DBSCAN 算法(Density-Based Spatial Clustering of Applications with Noise)

1.基本概念

(1)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 minPts）

(2) ϵ-邻域的距离阈值：设定的半径 r

(3)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

(4)密度可达：若有一个点的序列 q0、q1、…qk，对任意 qi-qi-1 是直接密度可达的 ，则称从 q0 到 qk 密度可达，这实际上是直接密度可达的“传播”。

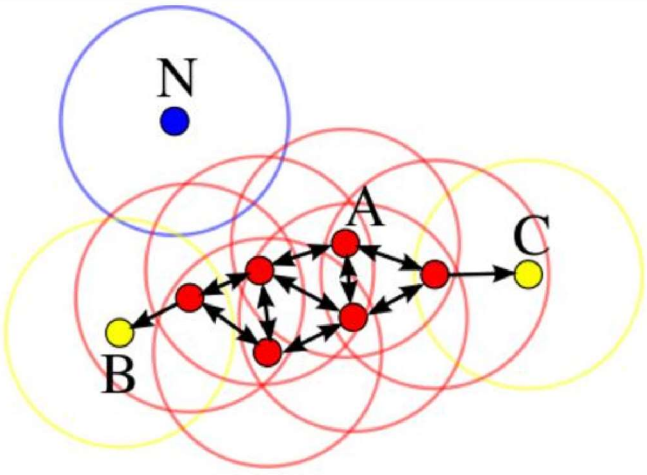
(5)密度相连：若从某核心点 p 出发，点 q 和点 k 都是密度可达的 ,则称点 q 和点 k 是密度相连的

(6)边界点:属于某一个类的非核心点,不能发展下线了

(7)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。

(8)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

举例：A：核心对象，B,C：边界点，N：离群点



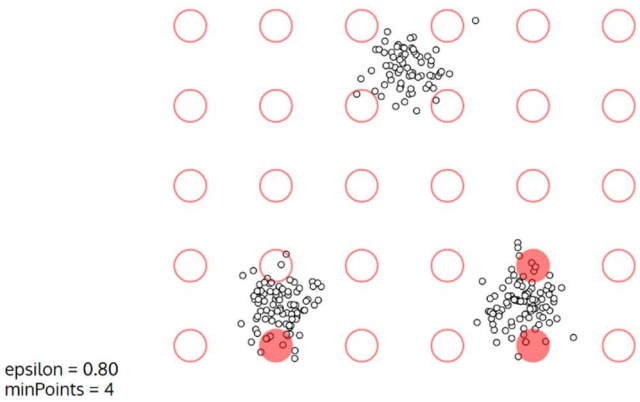
2.工作流程

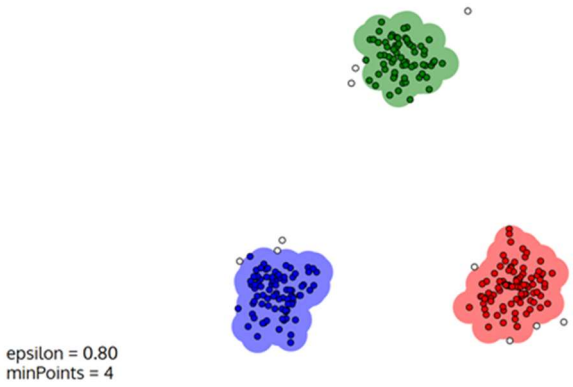
1. 标记所有对象为 unvisited;
2. Do
3. 随机选择一个 unvisited 对象 p;
4. 标记 p 为 visited;
5. If p 的 ε – 领域至少有 MinPts 个对象
6. 创建一个新簇 C，并把 p 添加到 C;
7. 令 N 为 p 的 ε – 领域 中的对象集合
8. For N 中每个点 p'
9. If p' 是 unvisited;
10. 标记 p' 为 visited;
11. If p' 的 ε – 领域至少有 MinPts 个对象，把这些对象添加到 N;
12. 如果 p' 还不是任何簇的成员，把 p' 添加到 C;
13. End for;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

其中：参数 D：输入数据集，参数 ϵ：指定半径，MinPts：密度阈值

我们还是直接来看看效果。

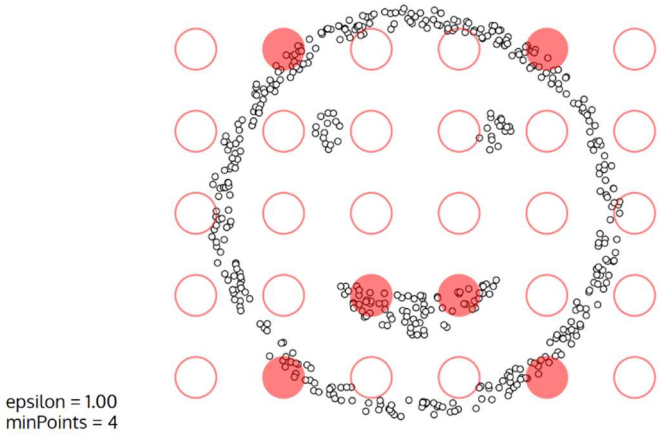
这次我们无需选择 k 值也无需选择基本点，但需要指定半径 ϵ 和密度阈值 MinPts



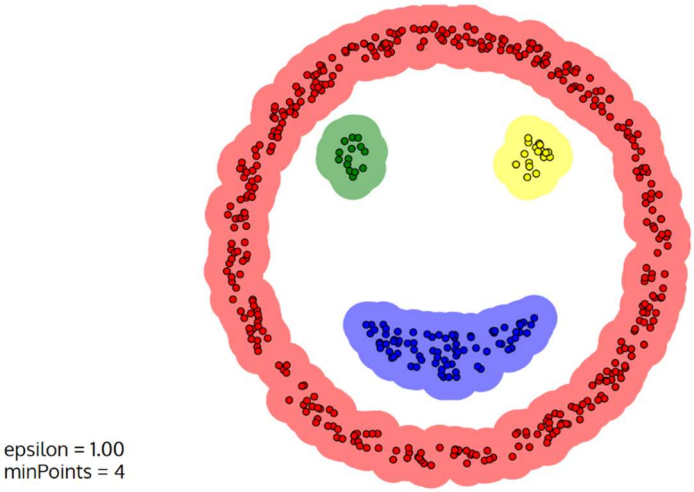


显然这个简单的问题难不倒它，与 K-MEANS 不同的是，DBSACAN 算法下出现了一些未被归类的点，即离群点。

我们再试试之前 K-MEANS 无法解决的“笑脸”数据。



设定半径为 1，密度阈值为 4。我们得到：



显然，在这个数据集中，DBSCAN 得到了不错的结果。

3. DBSCAN 算法的优劣

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：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）

这里有一些选择参数的建议：
半径 ϵ，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

https://www.naftaliharris.com/blog/visualizing-dbscanclustering/
https://www.naftaliharris.com/blog/visualizing-k-means-clustering/