

无监督学习算法输入的样本是不带标签的

聚类算法

k means算法是一种聚类算法也是常用的无监督学习算法

应用场景

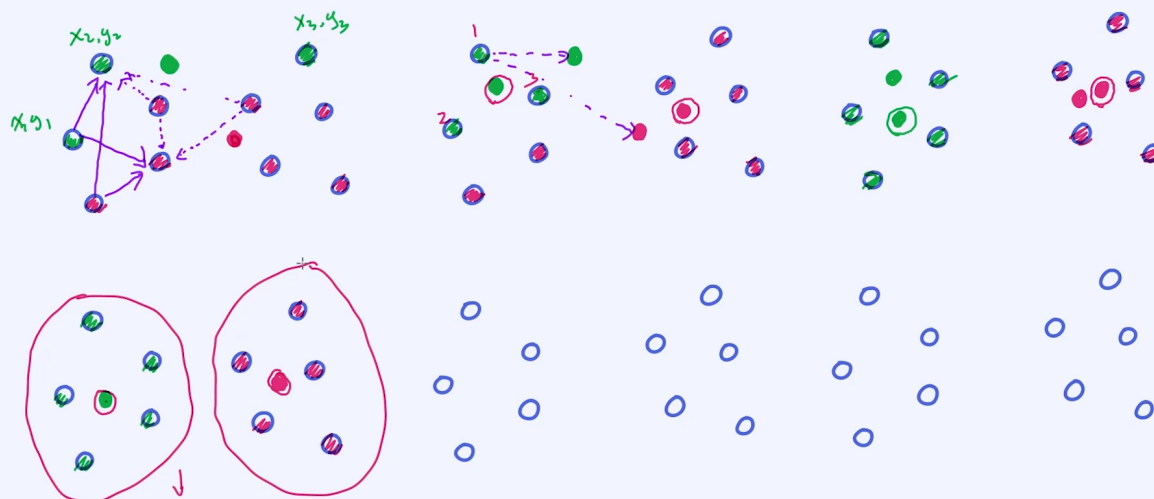
图像分割：把图像分成几个区域



$$\frac{x_1+x_2+x_3}{3}$$

$$\frac{y_1+y_2+y_3}{3}$$

K-Means 算法过程 $k=2$



中心点坐标改变 \rightarrow Done

若离各簇最近的点都是它的中心点

激活 Windows
转到“设置”以激活 Windows。

K-Means 算法

循环迭代式的算法

初始化:

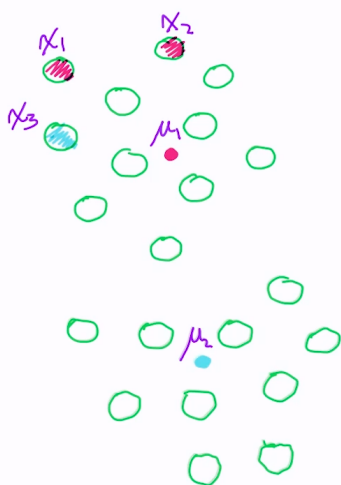
随机选择K个点, 作为初始中心点, 每个点代表一个group

交替更新:

计算每个点到所有中心点的距离, 把最近的距离记录下来
并把对应的group赋给当前的点

针对于每一个group里的点, 计算其平均并作为这个group
的新的中心点.

K-Means 的目标函数



$$D = \{x_1, x_2, \dots, x_N\}$$

$$L = \sum_{i=1}^N \sum_{j=1}^K \|x_i - \mu_j\|_2^2 \leftarrow \text{Objective Function}$$

↓

$$\|x_1 - \mu_1\|_2^2 + \|x_1 - \mu_2\|_2^2 + \|x_2 - \mu_1\|_2^2 + \|x_2 - \mu_2\|_2^2$$

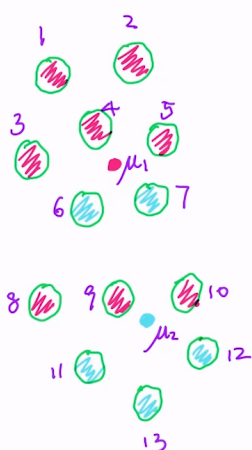
+ ...

$$\|x_1 - \mu_1\|_2^2 + \|x_2 - \mu_1\|_2^2 + \|x_3 - \mu_2\|_2^2 + \dots$$

k-Means的目标函数



贪心科技
GREEDY TECHNOLOGY



$$L = \|x_1 - \mu_1\|_2^2 + \|x_2 - \mu_1\|_2^2 + \|x_3 - \mu_1\|_2^2 + \|x_4 - \mu_1\|_2^2 + \|x_5 - \mu_1\|_2^2 + \|x_6 - \mu_1\|_2^2 + \|x_7 - \mu_1\|_2^2 + \|x_8 - \mu_1\|_2^2 + \|x_9 - \mu_1\|_2^2 + \|x_{10} - \mu_1\|_2^2 + \|x_{11} - \mu_1\|_2^2 + \|x_{12} - \mu_1\|_2^2 + \|x_{13} - \mu_2\|_2^2$$

未知参数: ① 每个样本属于哪个 cluster? $r_{ik} = \begin{cases} 0 & \text{otherwise} \\ 1 & x_i \text{ 属于 } k \text{ 个 cluster } k \end{cases}$
② 中心点 $\mu_1, \mu_2, \dots, \mu_k$

$r = (0, 1, 0, 0, 0) \Rightarrow 2 \text{ 个 cluster, } k=5$

$$L = \sum_{i=1}^N \sum_{k=1}^k r_{ik} \|x_i - \mu_k\|_2^2$$

激活 Windows

对于k-means的目标函数我们通常使用交替优化的方法。前面已经说过，目标函数里有两组不同的参数。我们可以分别对其中的每一组参数优化，但这时候相当于把另外一组参数当做是已知的就可以了。

k-means算法一定会收敛，因为圈1和圈2过程都会使目标函数下降。

k-means算法不同初始化的值会有不同的结果，其核心是非凸函数。如果一个目标函数是非凸函数，那我们其实不能保证或者没有办法得到全局最优解的！

没有标签无法用交叉验证

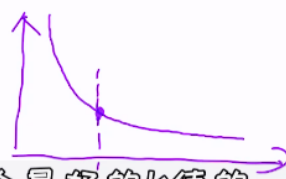
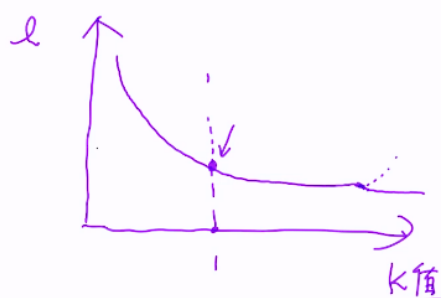
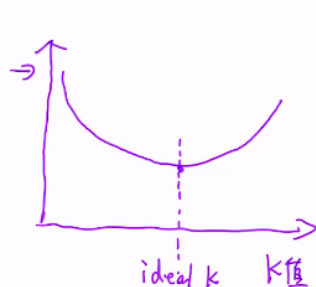
选择拐点为k值

k值如何选择？

$$L = \sum_{i=1}^N \sum_{k=1}^k r_{ik} \|x_i - \mu_k\|_2^2$$



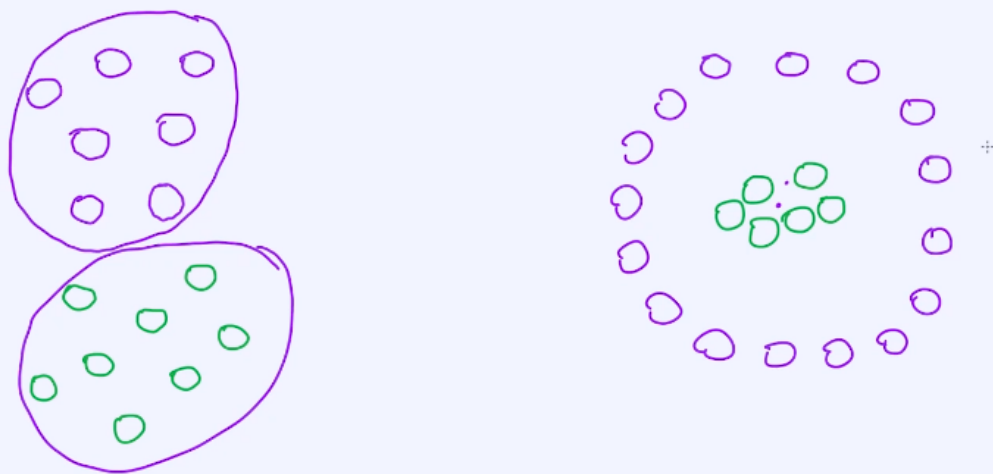
贪心科技
GREEDY TECHNOLOGY



选择一个最佳的k值的

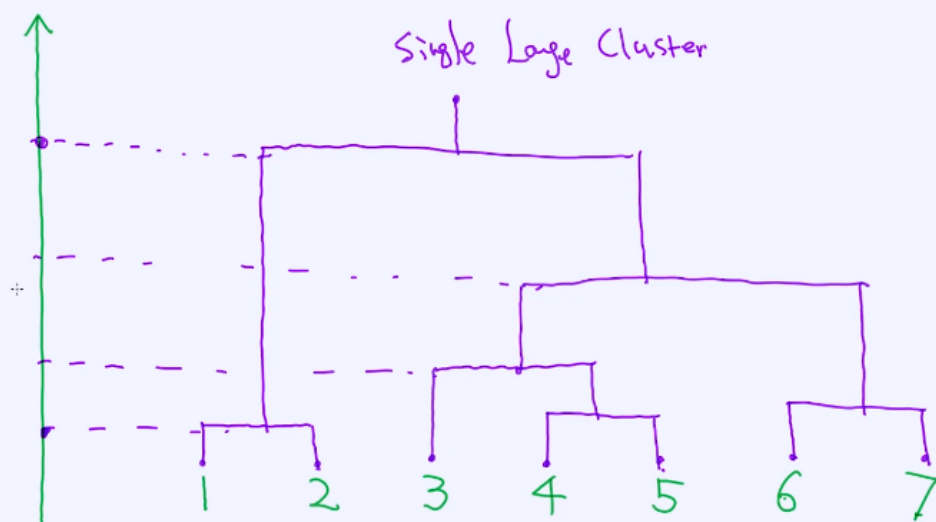
层次聚类不用选择k值，kmeans算法不能捕捉样本层次关系

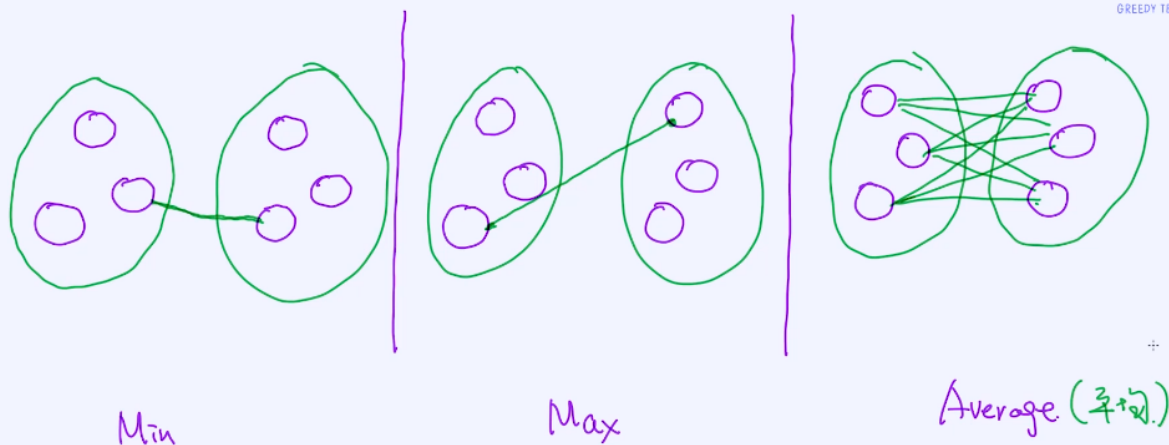
K-means



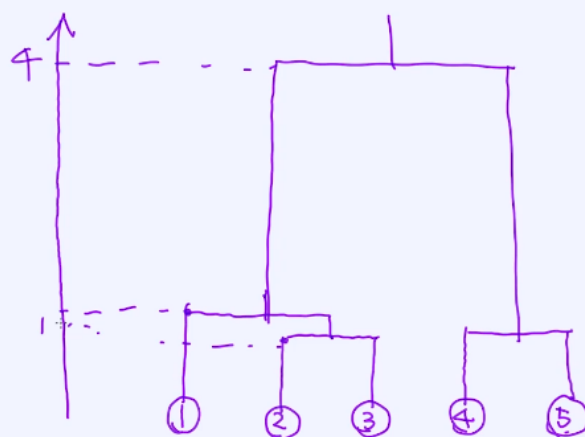
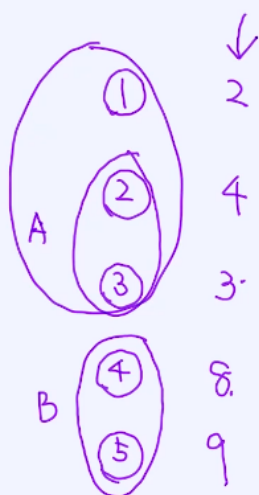
合并越顶层cluster之间的距离越大

Dendrogram

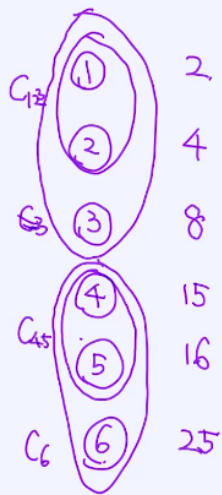




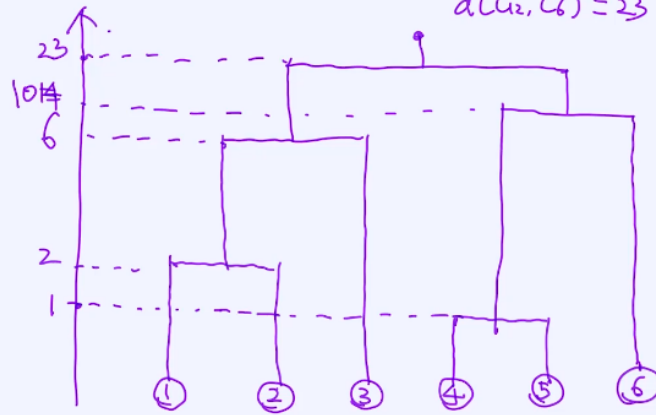
Min - Distance



Max - Distance.

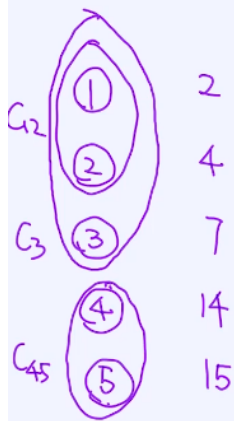


$$\begin{aligned} d(C_{12}, C_3) &= 6 & d(C_3, C_{45}) &= 8 \\ d(C_{12}, C_{45}) &= 14 & d(C_3, C_6) &= 17 \\ d(C_{12}, C_6) &= 23 & d(C_{45}, C_6) &= 10 \end{aligned}$$

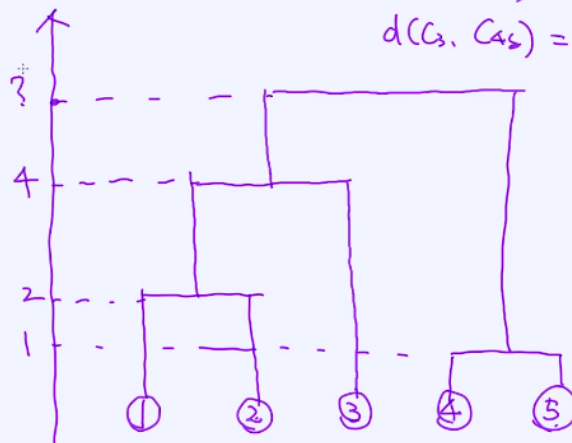


这三种方法里最稳定的方法

Average - Distance



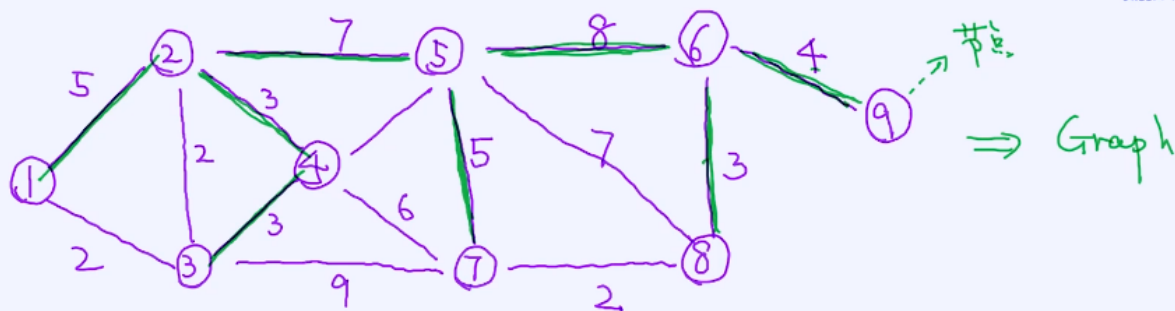
$$\begin{aligned} d(C_{12}, C_3) &= \frac{5+3}{2} = 4 \\ d(C_{12}, C_{45}) &= \frac{12+13+10+11}{4} = 11.5 \\ d(C_3, C_{45}) &= \frac{7+8}{2} = 7.5 \end{aligned}$$



$$\frac{12+13+10+11+7+8}{6}$$

MST

Minimum Spanning Tree (MST)



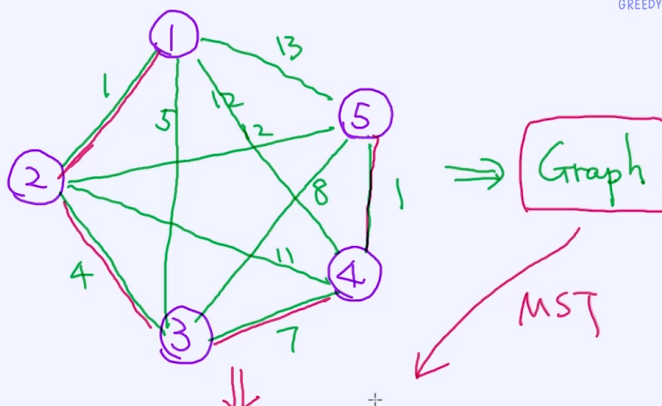
Prime,

自上而下聚类：最小生成树

按 Esc 即可退出全屏模式



- ① 2
- ② 3
- ③ 7
- ④ 14
- ⑤ 15



一个 cluster ① — 1 — ② — 4 — ③ — 7 — ④ — 1 — ⑤ (MST-)

二个 clusters ① — 1 — ② — 4 — ③ ④ — 1 — ⑤

三个 clusters ① — 1 — ② ③ ④ — 1 — ⑤

这里其实可以用类似于Elbow的方法。当我们合并完某两个clusters之后，就会发现距离会迅速变大，这个节点其实就是我们需要的！

在这里其实可以用类似于Elbow的方法。当我们合并完某两个clusters之后，就会发现距离会迅速变大，这个节点其实就是我们需要的！