原理：

1，随机在数据中抽取三个样本，当作三个类别的中心点（k1,k2,k3）

2，计算其余点分别到这三个中心点的距离（a,b,c）从中选择一个最近的距离当作自己的标记形成三个族群

3，分别计算这三个族群的**平均值**，把三个平均值与之前的三个旧中心点作比较，如果相同，结束聚类，不同则把三个平均值的点当作新的（k1,k2,k3），重复第二步

平均值计算：族群中每一个点的x，y坐标相加取平均，平均值的点有可能不在族群中

优点：

1，原理简单，实现容易

2，依赖K的取值

3，空间复杂度o(N)，时间复杂度o(I*K*N)

```
N为样本点个数，K为中心点个数，I为迭代次数
```

缺点：

1，K均值预先给定，很多情况下K值很难算，对于可以确定K值不会太大但不明确精确的K值的场景，可以进行迭代运算，然后找出Cost Function最小时所对应的K值，这个值往往能较好的描述有多少个簇类。

2，KMeans对初始聚类中心是敏感的，中心不同结果不同

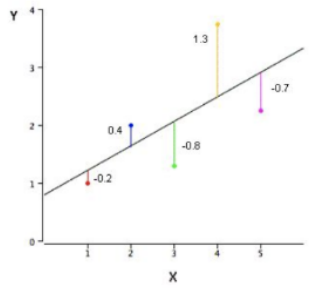
3，**K均值算法并不是很所有的数据类型。**它不能处理非球形簇、不同尺寸和不同密度的簇，银冠指定足够大的簇的个数是他通常可以发现纯子簇。

4，**对离群点的数据进行聚类时，K均值也有问题**，这种情况下，离群点检测和删除有很大的帮助。

评估：

1，误查平方和SSE：

举例:(下图中数据-0.2, 0.4, -0.8, 1.3, -0.7, 均为真实值和预测值的差)



SSE = (-0.2)^2 + (0.4)^2 + (-0.8)^2 + (1.3)^2 + (-0.7)^2 = 3.02

在k-means中的应用：

$$SSE = \sum_{i=1}^{k} \sum_{p \in C_i} \left| p - m_i \right|^2$$

2，肘方法，K值确定

（1）对于n个点的数据集，迭代计算k from 1 to n，每次聚类完成后计算每个点到其所属的簇中心的距离的平方和；

（2）平方和是会逐渐变小的，直到k==n时平方和为0，因为每个点都是它所在的簇中心本身。

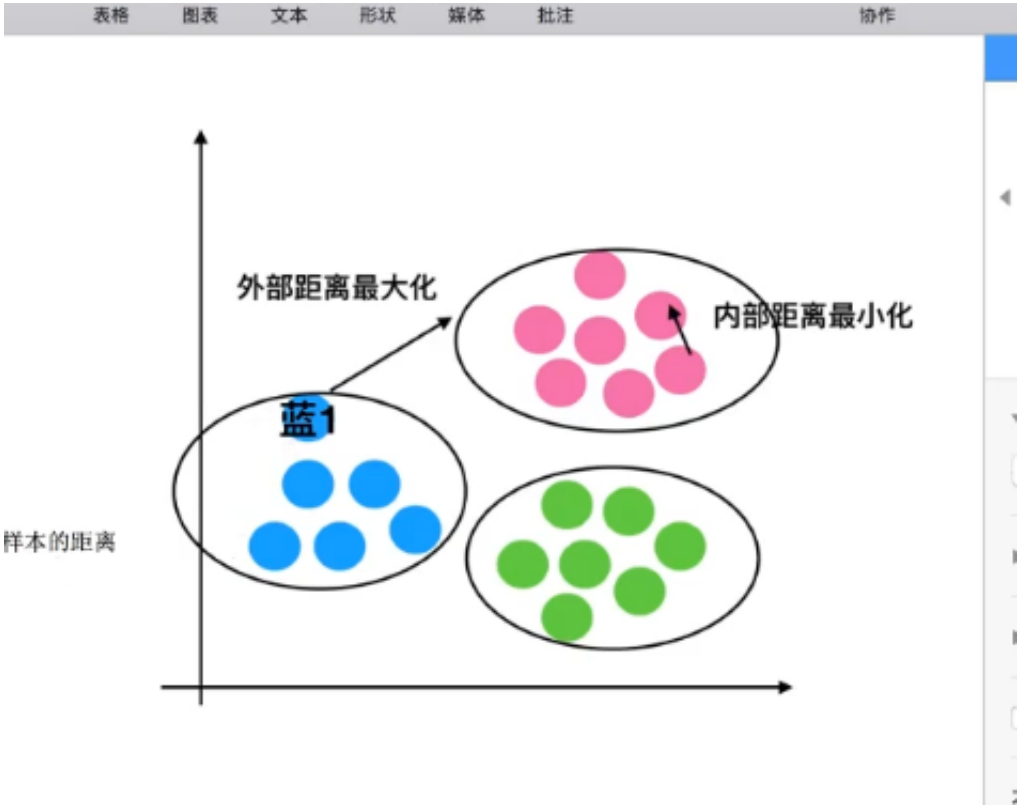
（3）在这个平方和变化过程中，会出现一个拐点也即“肘”点，**下降率突然变缓时即认为是最佳的k值**。

在决定什么时候停止训练时，肘形判据同样有效，数据通常有更多的噪音，在**增加分类无法带来更多回报时，我们停止增加类别**。

3，轮廓系数

轮廓系数：

计算公式：$sc_i = \frac{b_i - a_i}{\max(b_i, a_i)}$



（1）计算蓝1到自身类别点距离的平均值a_i

（2）计算蓝1分别到红色类别、绿色类别的点的距离，求平均值b1,b2取最小值当作b_i

极端：b_i>>a_i：完美情况，轮廓系数趋近为1

a_i>>b_i：最差，趋近为-1

蓝1的轮廓系数[-1,1]

一般超过0.1就算好，不可能达到1

4，CH系数

计算类中各点与类中心的距离平方和来度量类内的紧密度，计算各类中心点与数据集中心点距离平方和度量数据集的分离度，CH指标由分离度与紧密度的比值得到。从而，**CH越大代表着类自身越紧密，类与类之间越分散，即更优的聚类结果。**

评估时考虑多个指标评估结果更好

优化：

Canopy：

1，选择一点，以T1为半径画圆，圆内点为一个类A，在以T2为半径画圆，圆内点为一个类

2，再找T2半径外另外一点，以T1半径画圆，圆于之前的T1圆不相交，T1半径内为类B

3，迭代直到所有点都被划分

优点：

1.Kmeans对噪声抗干扰较弱，通过Canopy对比，将较小的NumPoint的Cluster直接去掉有利于抗干扰。

2.Canopy选择出来的每个Canopy的centerPoint作为K会更精确。

3.只是针对每个Canopy的内做Kmeans聚类，减少相似计算的数量。

缺点：

1.算法中 T1、T2的确定问题 ， 依旧可能落入局部最优解

KMeans++：

$$P = \frac{D(x)^2}{\sum_{x \in X} D(x)^2}$$

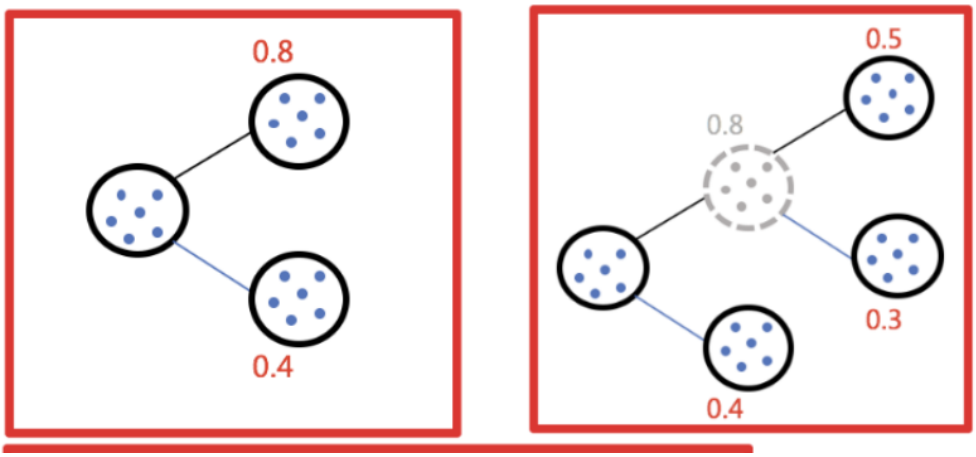
随机选取一点为质心，计算其他点的P值（其他点为质心的概率），最大的最有可能为下一个质心
D(x) = 其他点到质心点的距离

> $$\sum_{x \epsilon X} D(x)^2 = 1^2 + 2^2 + 1^2 + 2 \ldots\ldots$$

分母为其他店到质心的距离平方和。

二分KMeans：

1.所有点作为一个簇
2.将该簇一分为二
3.选择能最大限度降低聚类代价函数（也就是误差平方和）的簇划分为两个簇。
4.以此进行下去，直到簇的数目等于用户给定的数目k为止。



二分K均值算法可以加速K-means算法的执行速度，因为它的相似度计算少了并且不受初始化问题的影响，因为这里不存在随机点的选取，且每一步都保证了误差最小

K-medoids：（K中心聚类算法）

K-medoids和K-means是有区别的，**不一样的地方在于中心点的选取**

- K-means中，将中心点取为当前cluster中所有数据点的平均值，对异常点很敏感!
- K-medoids中，将从当前cluster 中选取到其他所有（当前cluster中的）点的距离之和最小的点作为中心点。

流程：

1，总体n个样本点中任意选取k个点作为medoids

2，形成簇

3，计算每个簇内的点到其所在簇的其他点的距离，距离之和最小的形成新的medoids

4，重复2-3

5，最终确定K个类

优点：

1，受噪声影响小

2，只能对小样本起作用，大样本就太慢了，样本多的时候少数噪音对质心影响小，K-Means应用比K-mediods多

优化方法：

优化方法	思路
Canopy+kmeans	Canopy粗聚类配合kmeans
kmeans++	距离越远越容易成为新的质心
二分k-means	拆除SSE最大的簇
k-medoids	和kmeans选取中心点的方式不同
kernel kmeans	映射到高维空间
ISODATA	动态聚类，可以更改K值大小
Mini-batch K-Means	大数据集分批聚类

大数据：样本多于一万，大数据优先用Mini-batch K-Means

动态聚类：不需要K值

经验：

1，聚类模型考虑多个评估指标

2，及时剔除不必要特征

3，特征不宜过多使模型过于复杂

4，个别特征对模型影响过大，根据业务需求可以舍去

5，K值得选择，不必要非选效果最好得K值，考虑实际业务需求