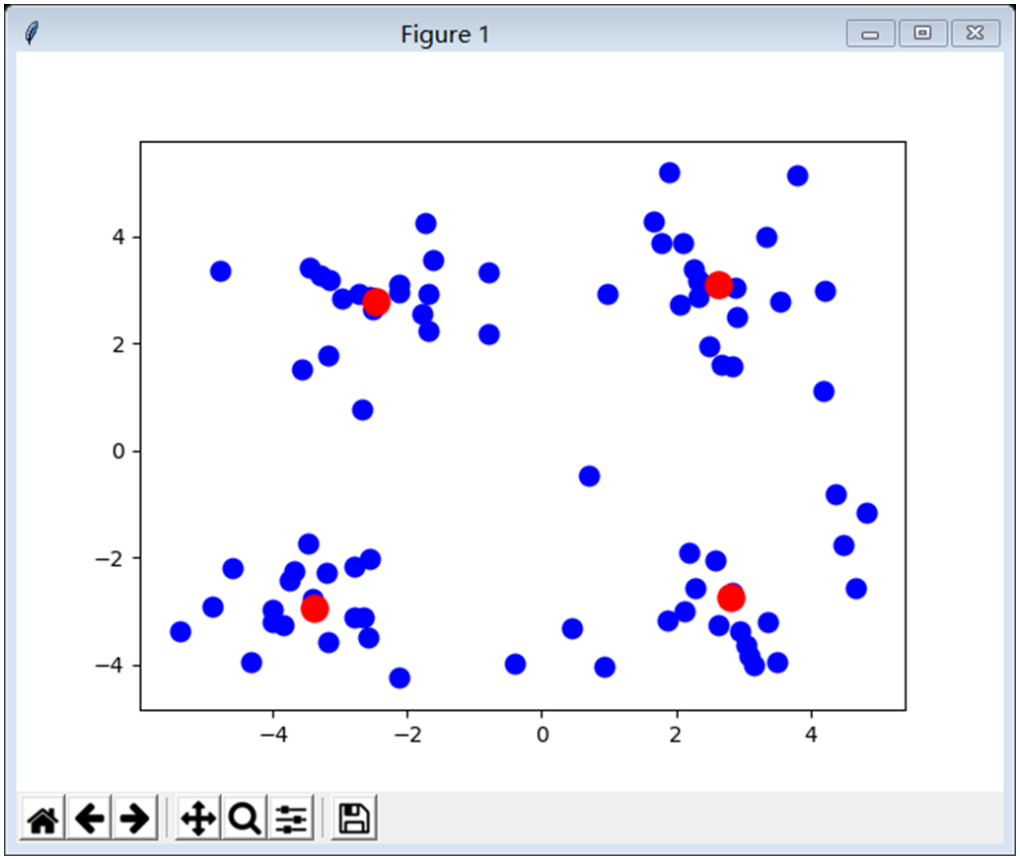
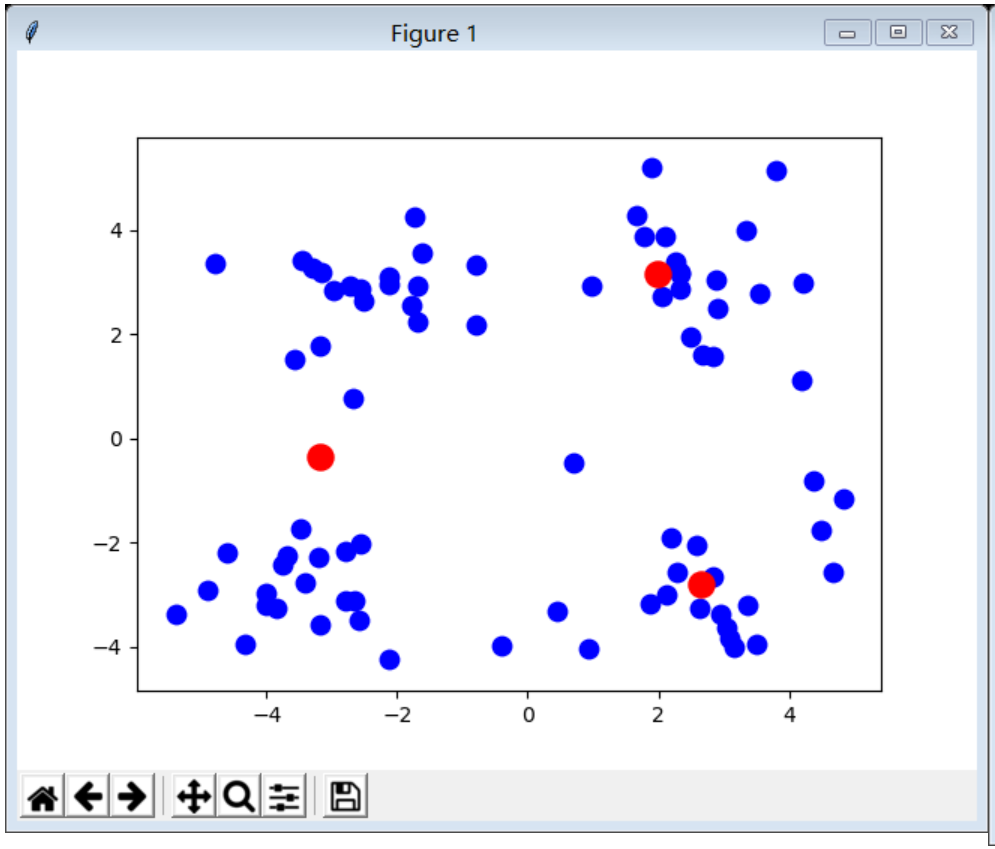
k-means缺点及改进

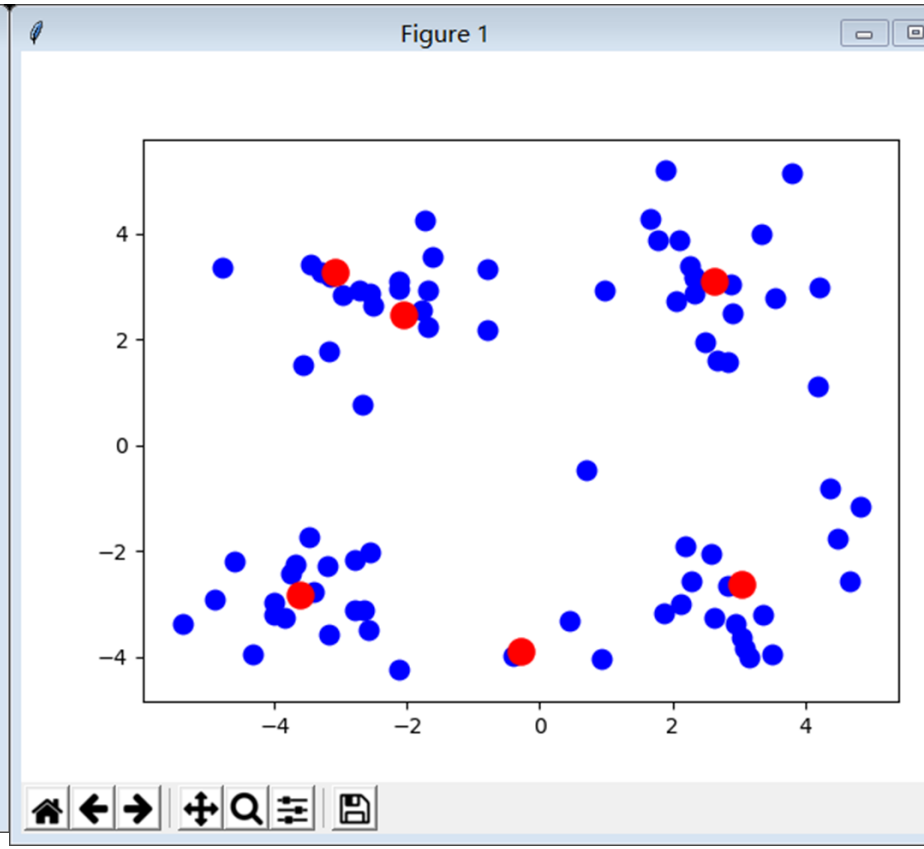
k-means算法缺点1：k值需要预先给定

- K-means算法执行时，如果给定的k值比较合理，算法可以得出比较好的结果



如果给定的k值不合理，就会对结果造成很大的影响



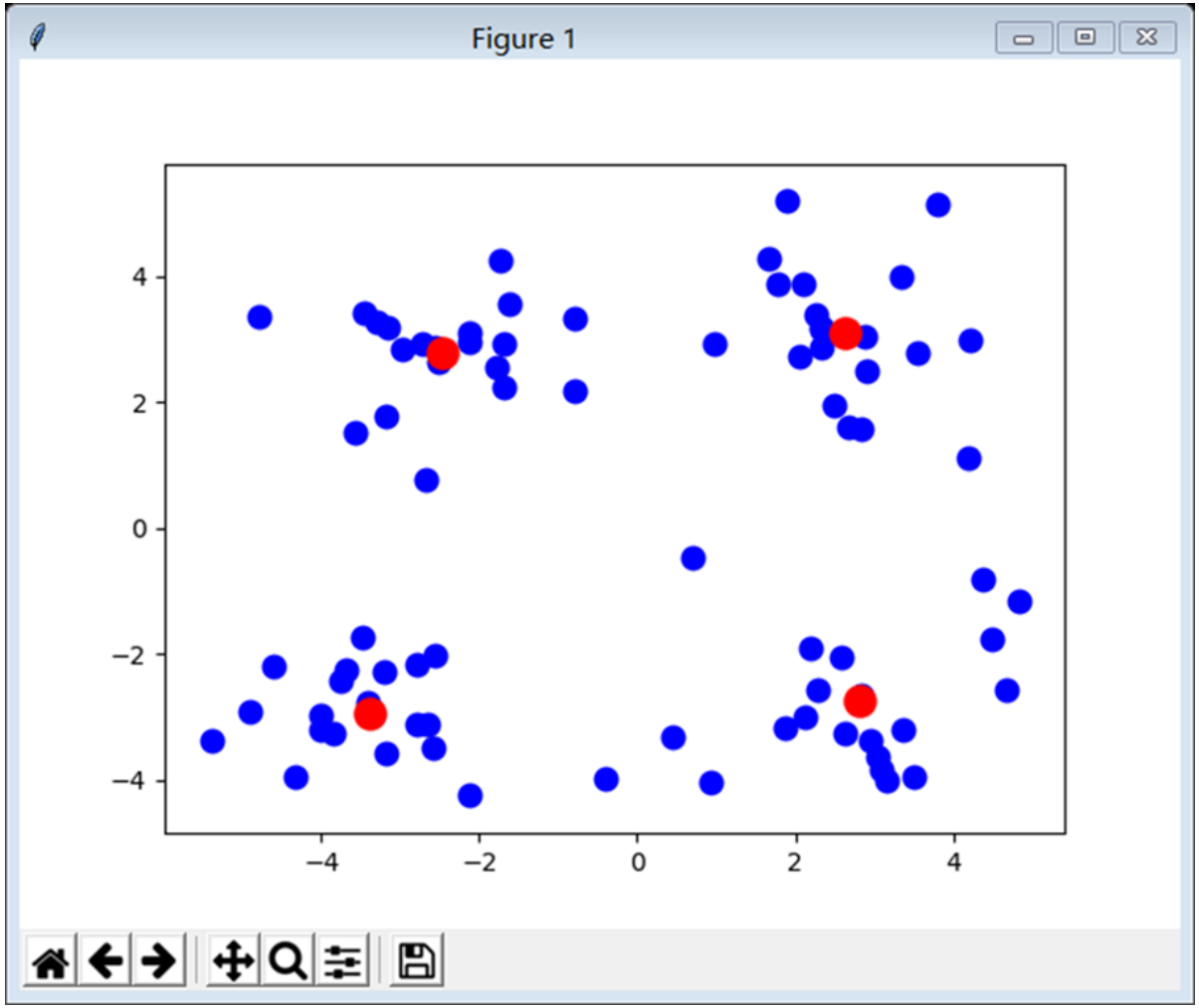


确定k值

K-means算法确定k值主要有以下几种方法：

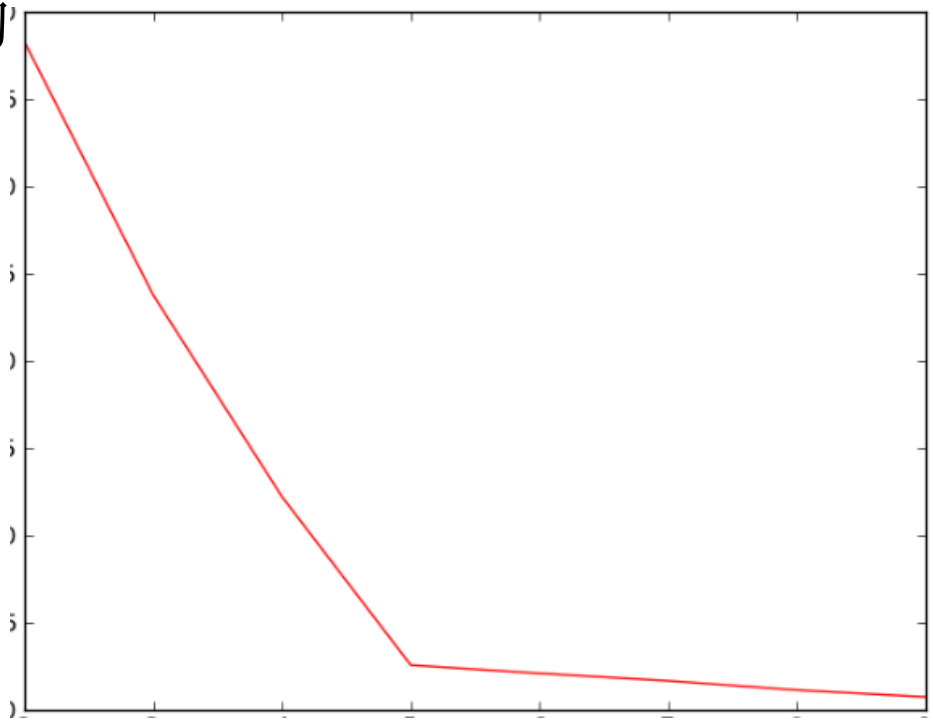
1. 经验法；
2. 肘部法则(逐个实验法)；
3. 密度法；
4. ISODATA算法

肘部法则1：利用SSE



肘部法则1：利用SSE

- 随着聚类数k的增大，样本划分会更加精细，每个簇的聚合程度会逐渐提高，那么误差平方和SSE自然会逐渐变小。并且，当k小于真实聚类数时，由于k的增大会大幅增加每个簇的聚合程度，故SSE的下降幅度会很大，而当k到达真实聚类数时，再增加k所得到的聚合程度回报会迅速变小，所以SSE的下降幅度会骤减，然后随着k值的继续增大而趋于平缓，也就是说SSE和k的关系图是一个手肘的形状，而这个肘部对应的k值就是数据的真实聚类数。

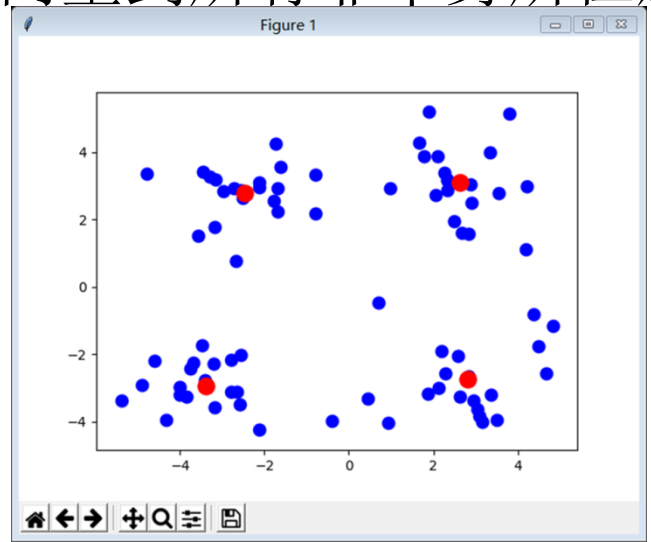


肘部法则2：利用轮廓系数

- 轮廓系数(Silhouette Coefficient)，是聚类效果好坏的一种评价方式。最早由 Peter J. Rousseeuw 在 1986 提出。它结合内聚度和分离度两种因素。
- 假设已经通过K-means算法将某数据集分为k个簇，对于数据集中的每一个点i，有

$$a(i) = average(\text{点}i\text{到每一个与}i\text{同簇的点的距离})$$

$$b(i) = min(i\text{向量到所有非本身所在簇的点的平均距离})$$



肘部法则2：利用轮廓系数

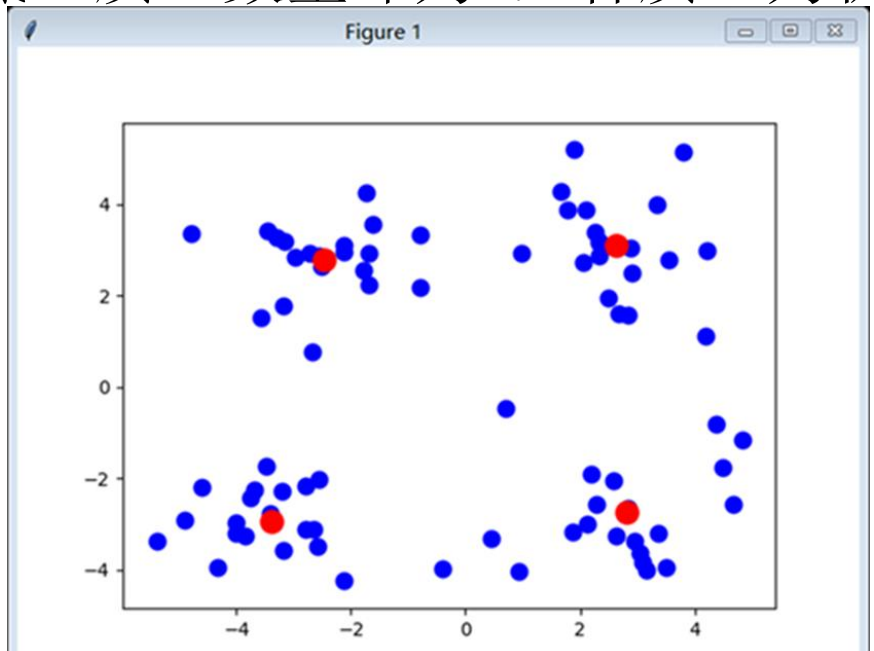
- 点i的轮廓系数为

$$S(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$$

- 数据集中每一个点的轮廓系数的平均数即为该次聚类的总轮廓系数。
- 显然轮廓系数的值是介于［-1,1］，当k=1时，轮廓系数为-1，当k=n(n为样本点的数量)时，轮廓系数为1。
- 对于轮廓系数同样可以利用肘部法则，当轮廓系数随着k值的增加上升速度变得缓慢时说明已经找到了比较好的k值。
- 注意：肘部法则并不总是成立！

密度法确定K值

- 选取距离r，以每个数据点为中心，计算距离该点r范围内的其他数据点的个数，该个数即为该点密度。根据密度对数据集中的各数据点进行排序，密度最大的为第一质心，在该质心半径r范围外选取密度最大的点为第二质心，重复这一步骤直到所有数据点都在各质心范围内或者剩余的数据点密度都很低。质心数量即为k，各质心为初始质心点。



密度法的优缺点

- 优点：在确定k值的同时确定了初始质心的位置，而且初始质心在数据点比较密集的地方，减少了由于初始质心随机选定造成的收敛于局部最小值的情况。
- 缺点：需要指定r，计算量大。

ISODATA算法

- ISODATA的全称是迭代自组织数据分析法。ISODATA就是针k-means算法k值不易确定的问题进行了改进。它的思想很直观：当属于某个类别的样本数过少时把这个类别去除，当属于某个类别的样本数过多、分散程度较大时把这个类别分为两个子类别。
- ISODATA算法需要首先给定几个参数：预期的k值，每一个簇的最少样本数N_{min}，簇内最大方差σ，两个簇之间的最小距离d_{min}。

ISODATA算法

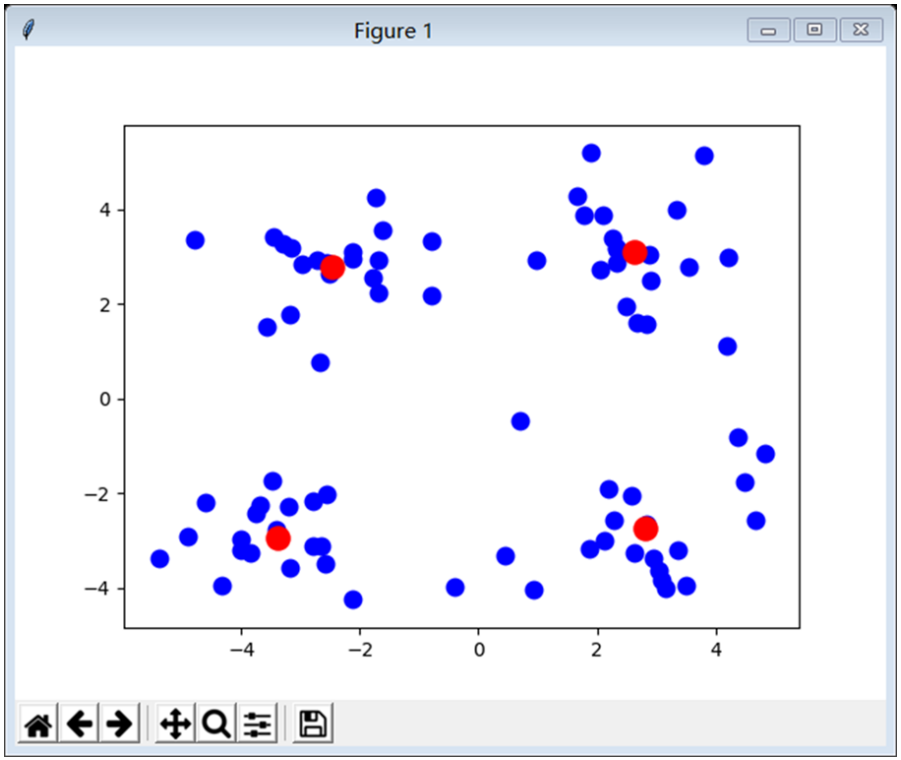
- 预期的k值：虽然在ISODATA运行过程中聚类中心数目是可变的，但还是需要由用户指定一个参考标准。事实上，该算法的聚类中心数目变动范围也由k决定。具体地，最终输出的聚类中心数目范围是$[k/2, 2k]$。
- 簇内最大方差σ：用于衡量某个类别中样本的分散程度。当样本的分散程度超过这个值时，则有可能进行分裂操作。

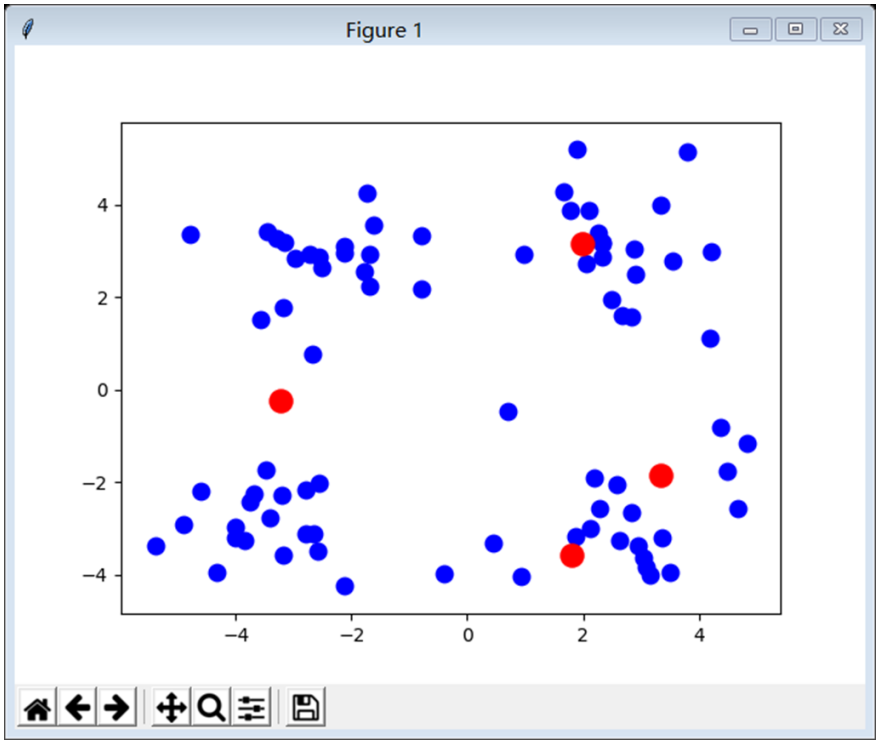
ISODATA算法

- 每一个簇的最少样本数N_{min}：用于判断当某个类别所包含样本分散程度较大时是否可以进行分裂操作。如果分裂后会导致某个子类别所包含样本数目小于N_{min}，就不会对该类别进行分裂操作。
- 两个簇之间的最小距离d_{min}：如果两个类别靠得非常近（即这两个类别对应聚类中心之间的距离非常小），则需要对这两个类别进行合并操作。

K-means算法缺点2：收敛于局部最优解

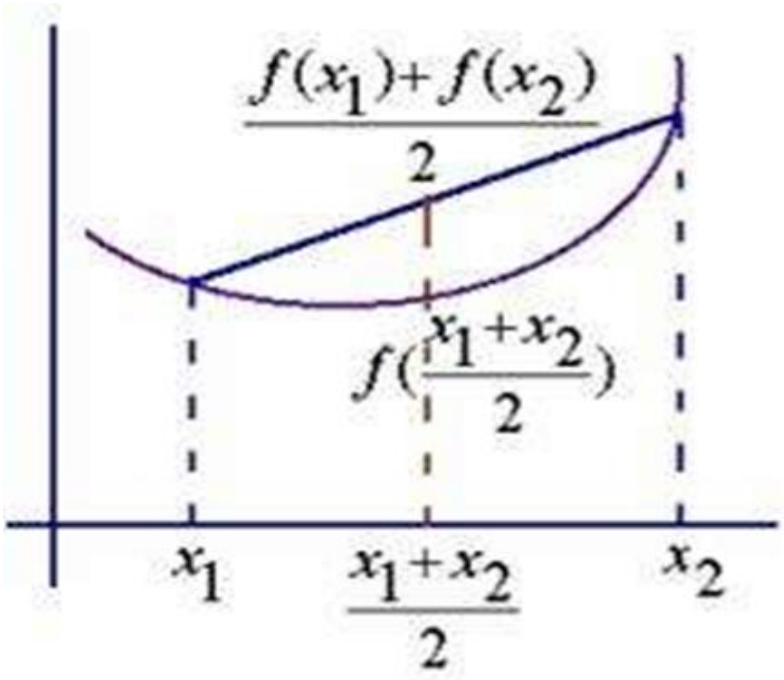
- K-means算法如果初始质心选取不当会收敛于局部最优解。

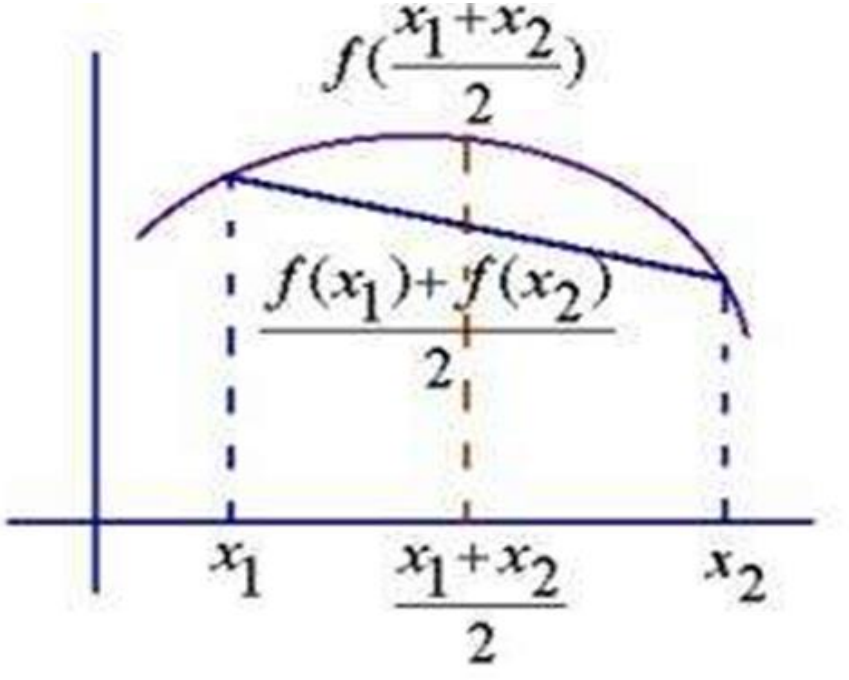




凸函数

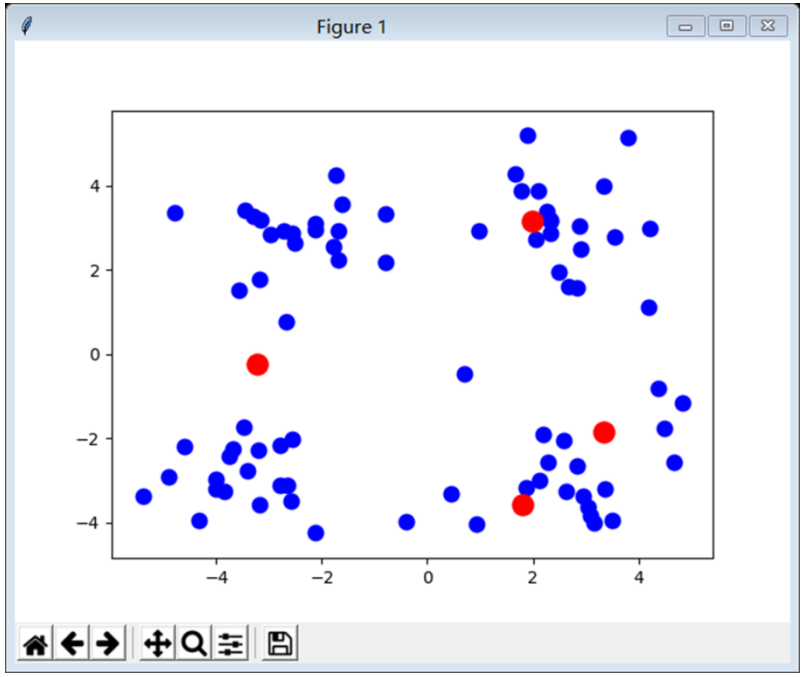
- 如果函数任意两点连线上的值大于对应自变量处的函数值则该函数称为凸函数，如下面左图。





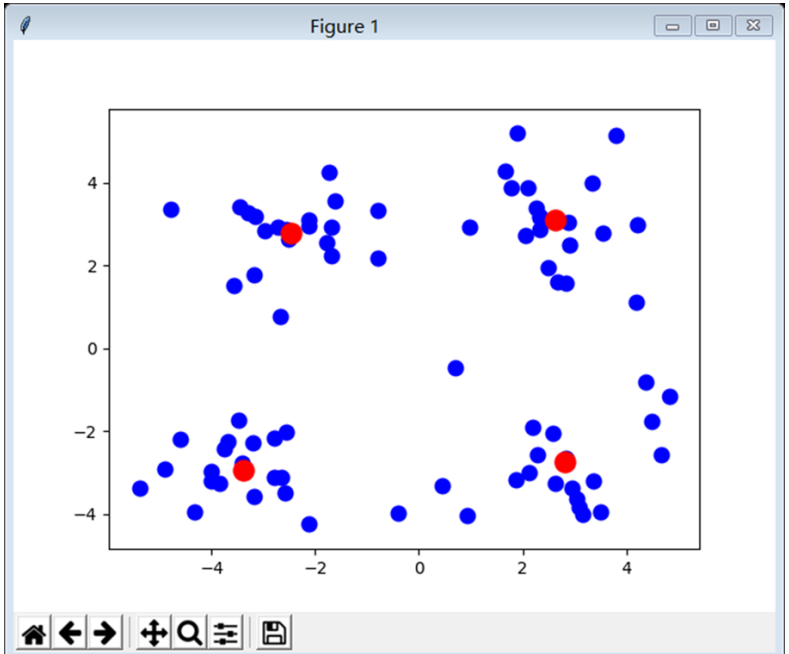
- 有的数学资料定义下面右图为凸函数，机器学习中统一认为左侧是凸函数

- 凸函数只有一个极小值，所以极小值就是最小值，但包括SSE在内的非凸函数没有此特征，如下图所示，SSE有时会在局部最小值处收敛，导致聚类结果较差。



K-means++算法

- K-means++按照如下的思想选取K个聚类中心：假设已经选取了n个初始聚类中心(0<n<k)，则在选取第n+1个聚类中心时：距离当前n个聚类中心越远的点会有更高的概率被选为第n+1个聚类中心。在选取第一个聚类中心(n=1)时同样通过随机的方法。



k-means++算法流程

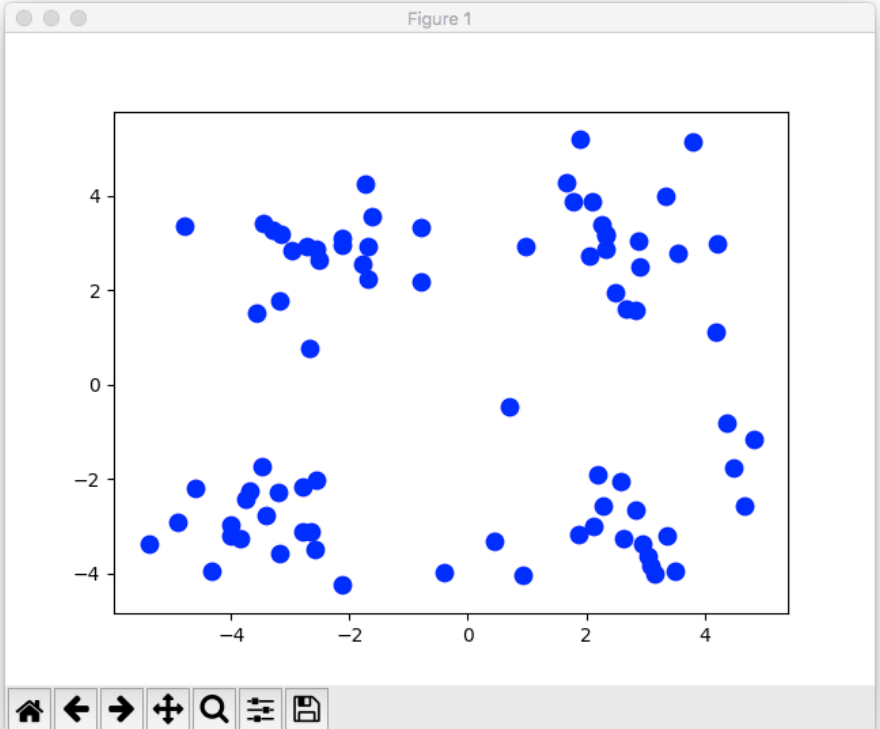
第一步：随机在数据集X中选取一个点作为第一个质心；

第二步：计算数据集X中每个样本与已有的质心之间的最短距离，用$D(x)$表示，计算该点成为下一个聚类中心的概率$p = \frac{D(x)^2}{\sum_{a \in X} D(a)^2}$，按照轮盘法确定下一个聚类中心。

第三步，重复第二步直到选定k个初始质心。

二分K-means算法

- 二分k-均值算法首先将所有点作为一个簇，然后将该簇一分为二，之后选择其中一个簇继续划分，选择哪个簇进行划分取决于是否可以最大程度降低SSE，上述过程不断重复，直到簇的数量达到指定的数量k。



二分K-means算法伪代码

将所有点看成一个簇

当簇的数目小于k时

    对于每一个簇

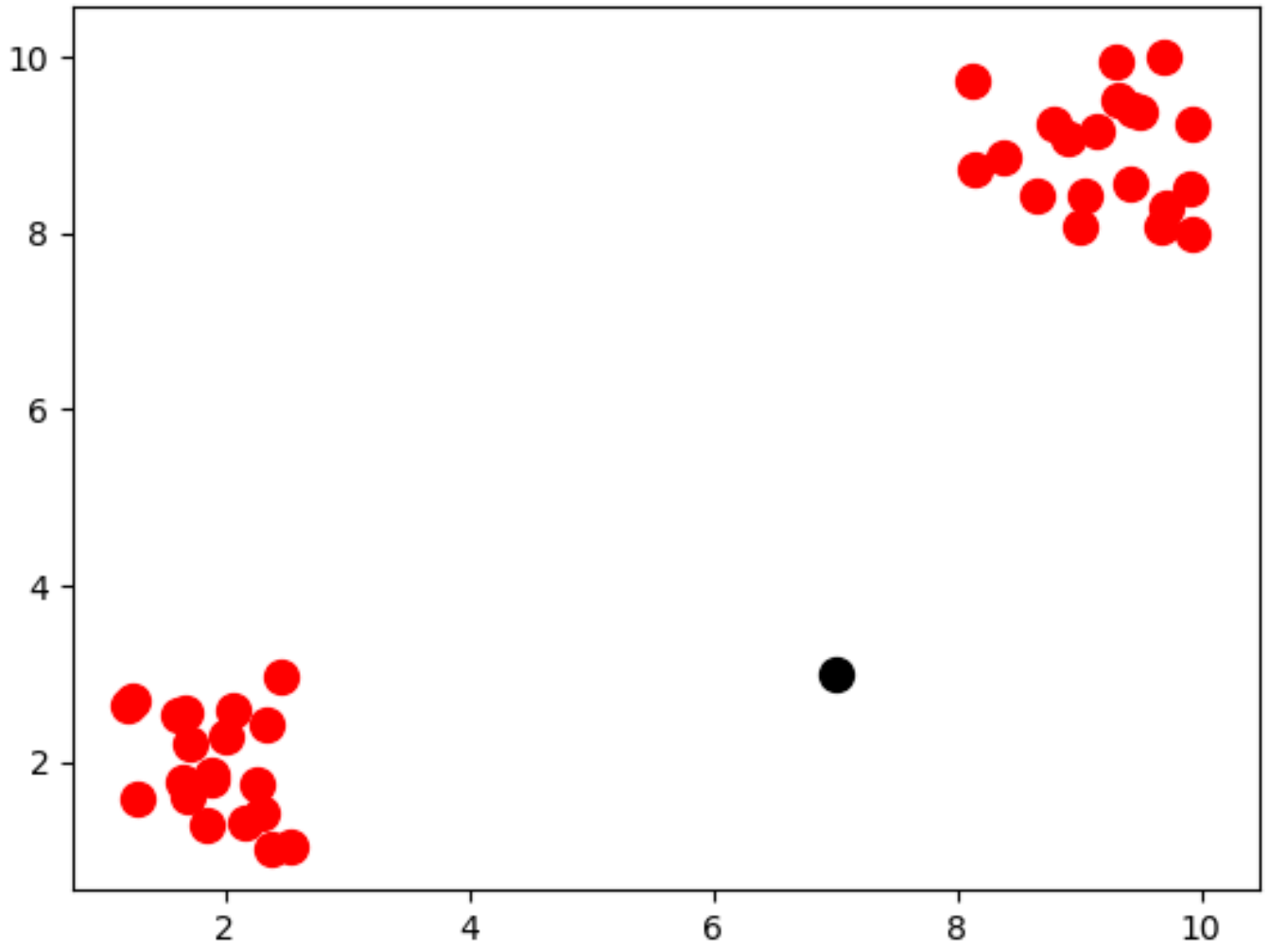
        计算总误差

            在给定的簇上面进行K-means聚类(k=2)

            计算将该簇一分为二后的总误差

    选择使得误差较小的那个簇继续划分

缺点3：对噪声点和离群点敏感



k-medoids(k-中心点)算法

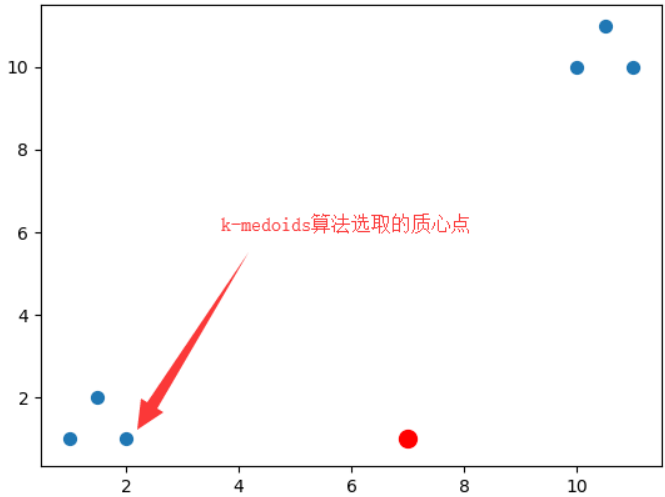
- 为了克服k-means算法对噪声点和离群点敏感的缺点，有人提出了k-medoids算法。
- k-medoids算法与k-means算法主要有两点不同。

1.k-medoids算法的质心一定是数据集中的某个点，但k-means可以是任意点。

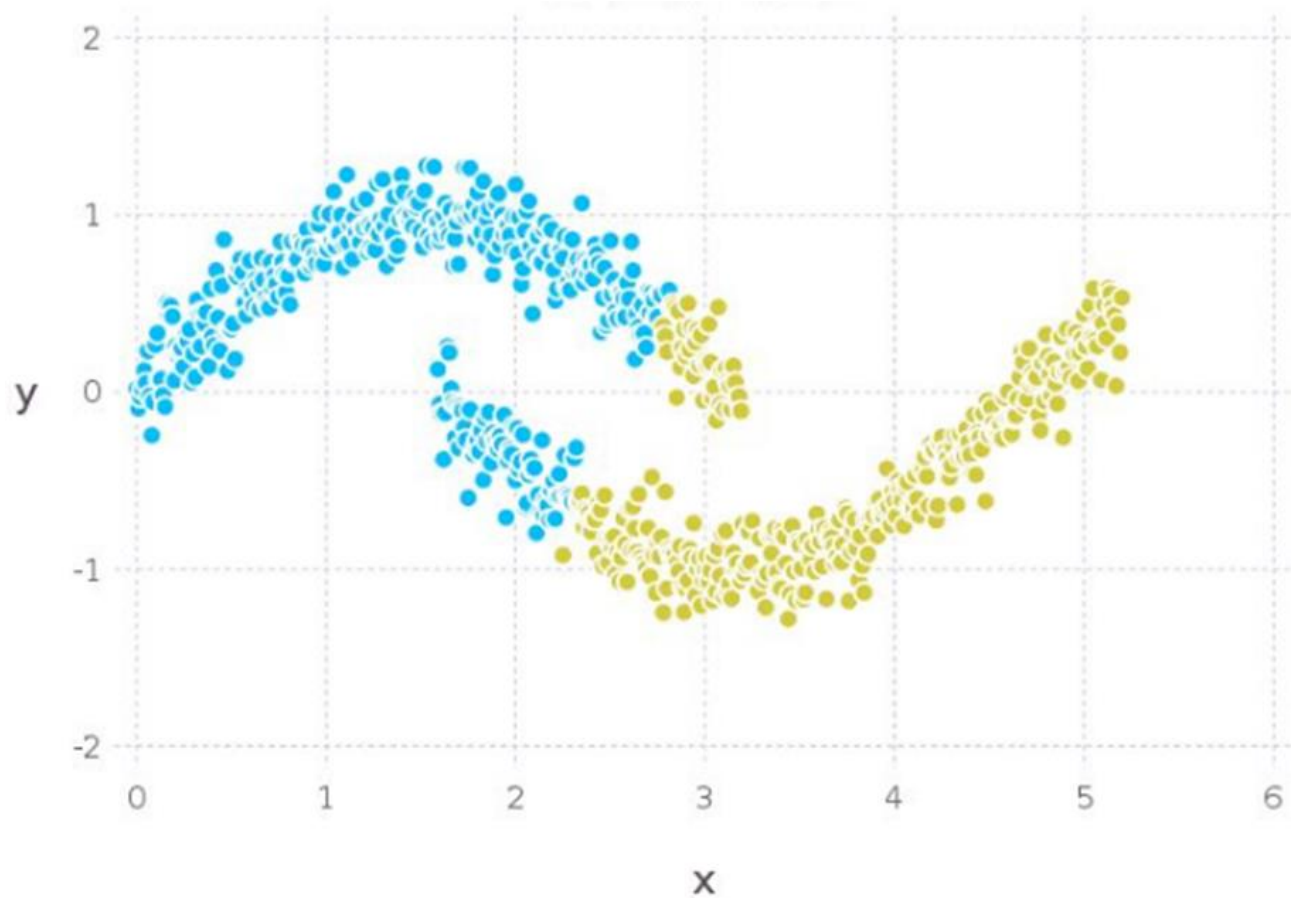
2.质心位置更新时不再根据均值，而是分别对簇内所有点进行计算，由代价最小的作为新的质心。代价也可以用SSE表示。

k-medoids(k-中心点)算法

- k-medoids可以较好的解决离群的和噪声点的问题，但时间复杂度较高。
- 如图所示，如果令k=2，可能会使得红色的点属于左下角的簇，而该簇的质心会在左下角三个蓝色点之外，质心位置偏离实际值较大，但如果用k-medoids方法，质心点会选择三个蓝色点中的左下角那个点，偏离实际程度不大。



缺点4：不适合非球状簇



- 该缺点可采用核k-means算法或者基于密度的聚类等其他聚类算法解决。

缺点5：只适合数值型数据

- k-means算法是一种简单且实用的聚类算法，但是传统的k-means算法只适用于数值型的数据集，而对于非数值型的数据集，计算簇的均值以及点之间的欧式距离就变得不合适了。

K-modes算法

- k-modes算法是对k-means算法的扩展。k-modes算法采用差异度来代替k-means算法中的距离。k-modes算法中差异度越小，则表示距离越小。一个样本和一个聚类中心的差异度就是它们各个属性不相同的个数，不相同则记为一，最后计算一的总和(如果觉得属性之间的差异单纯的用0和1表示过于理想化，也可以加入权重，比如(2,3)和(2,10)两个点可以认为它们的距离为1，也可以认为3和10的差距过大，所以距离用曼哈顿距离表示，认为距离为7)。这个和就是某个样本到某个聚类中心的差异度。该样本属于差异度最小的聚类中心。

K-modes算法

- K-modes算法流程为：

步骤1；随即确定k个质心；

步骤2：计算各个样本与这k个质心的距离，并将其分配给距离最小的质心；

步骤3：更新质心位置，新的质心的各个属性由该簇中所有数据点各个属性的众数决定。

重复步骤2和步骤3，直到总距离不再降低。

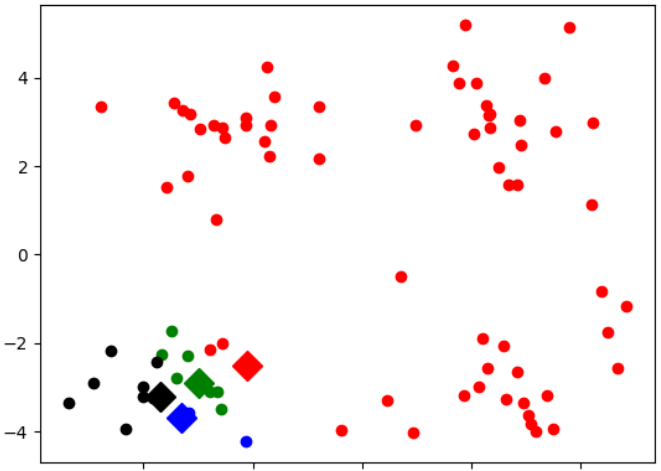
Mini Batch(分批处理) k-means算法

- 普通k-means算法虽然时间复杂度已经比较低了，但仍不能令人满意，所以又有人提出了Mini Batch(分批处理) k-means算法，该算法流程分为两步：

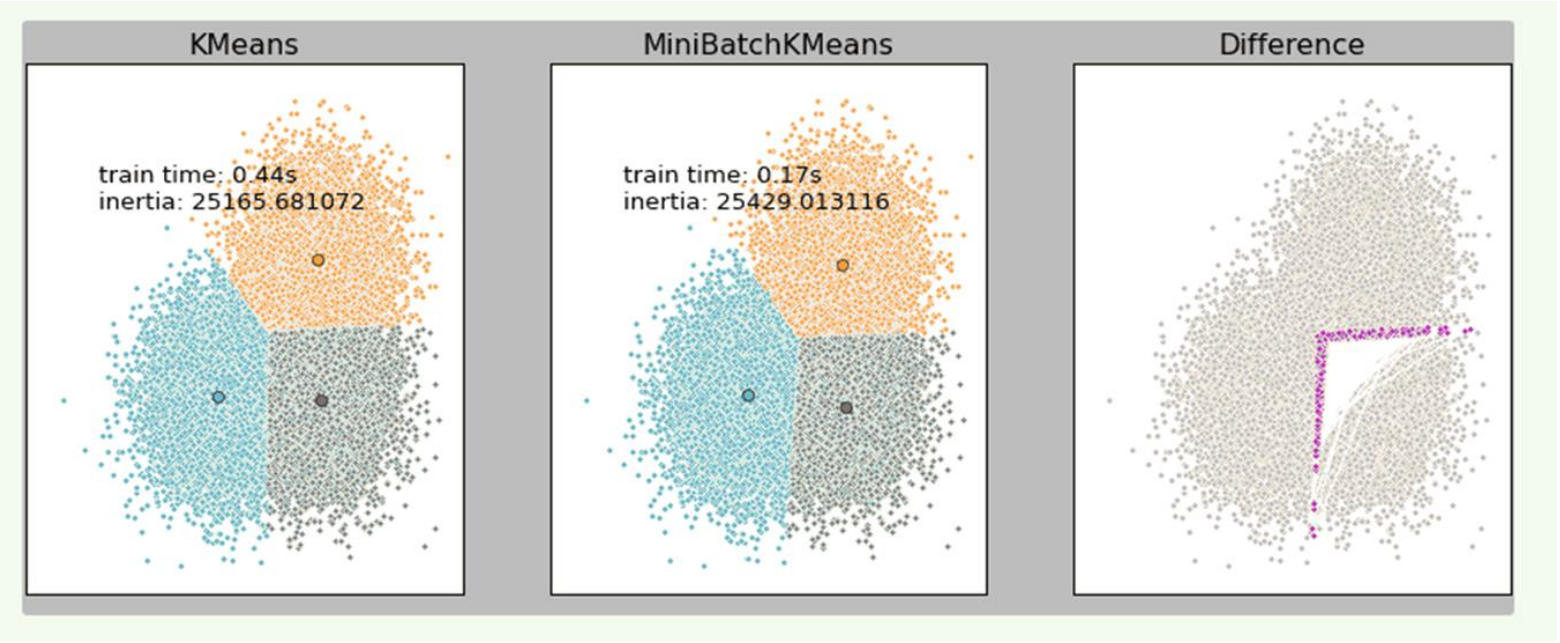
  1. 从不同类别的样本中抽取一部分样本为代表

     计算相应的质心；

  2. 更新质心。



- 该算法在数据量较少时没有什么明显的优势，甚至有可能造成计算时间的增加，但当数据量比较大时会有更快的收敛速度，但精度有所降低。
- 该思想还广泛应用于梯度下降、深度网络等机器学习和深度学习算法。



- 3万个样本点分别使用K-Means和Mini Batch K-means进行聚类的结果。