4.6 KMeans

CSDN学院

2017年11月

常用聚类算法

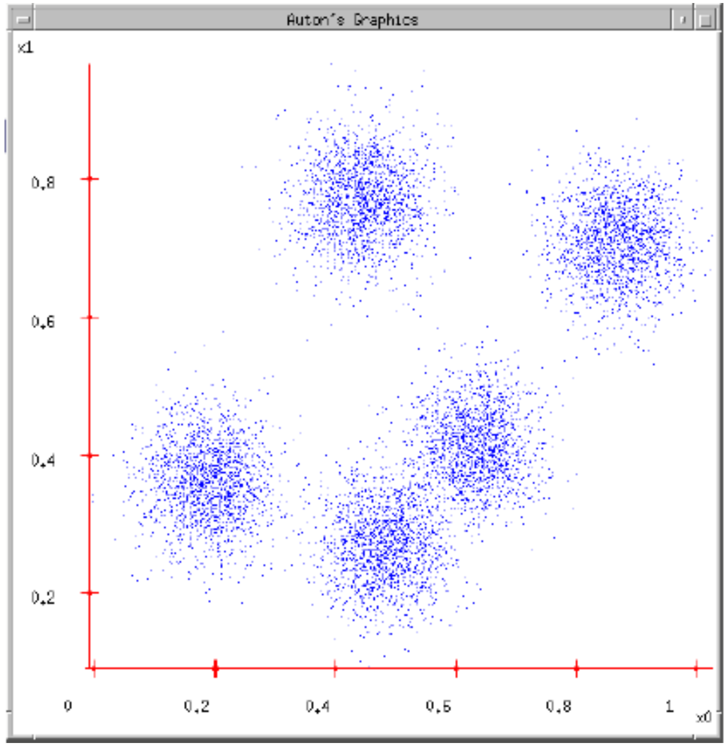
- 基于距离、相似度的聚类算法
  - *K*-means（*K*均值）及其变种（ *K*-centers 、 Mini Batch *K*-Means ）
  - Mean shift
  - 吸引力传播（Affinity Propagation ，AP）
  - 层次聚类
    - 聚合聚类（Agglomerative Clustering）
- 基于密度的聚类算法
  - DBSCAN、DensityPeak（密度最大值聚类）
- 基于连接的聚类算法
  - 谱聚类

K-means

- *K*-means是最简单的聚类方法

▶ *K*-means

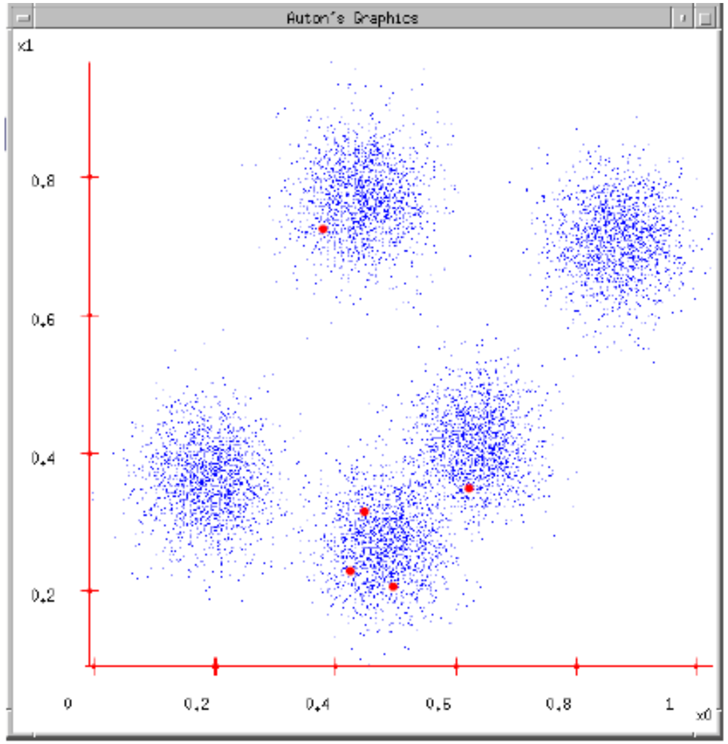
- 1、给定数据



K-means

- 2、确定类别数K（如K=5），并初始化K个类的中心（如随机选择K个点）

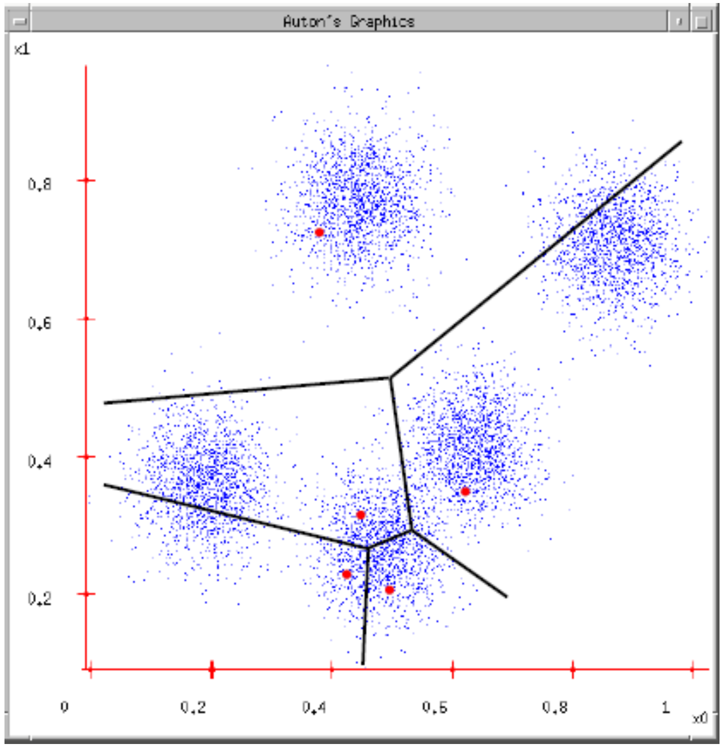
$$\boldsymbol{\mu}^{(0)} = \boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_K^{(0)}$$



▶ *K*-means

- 3、对每个数据点，计算离其最近的类（使得每个类拥有自己的一些数据）

$$C_i^{(t)} = \arg\min_k \left(\boldsymbol{\mu}_k - x_i\right)^2$$



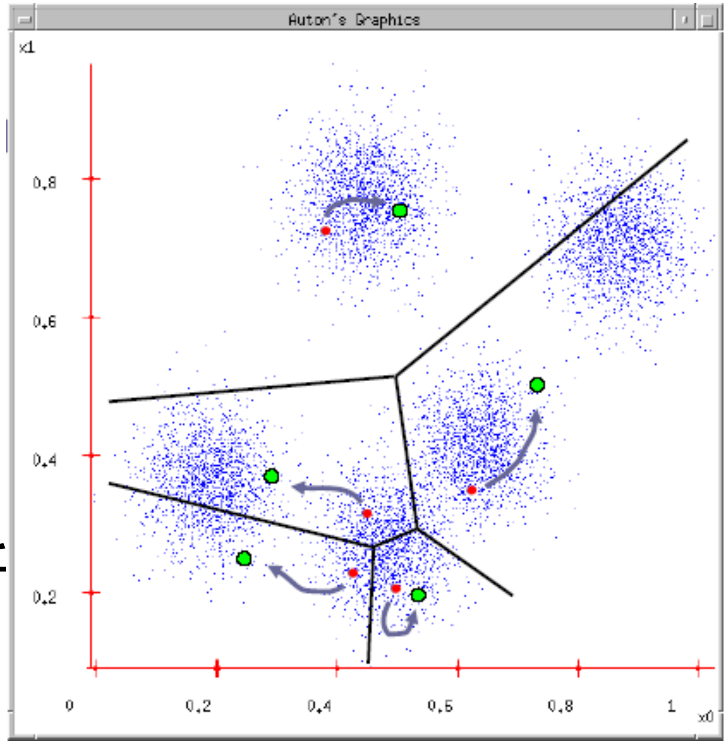
▶ *K*-means

- 4、对每个类，计算其所有数据的中心，并跳到新的中心

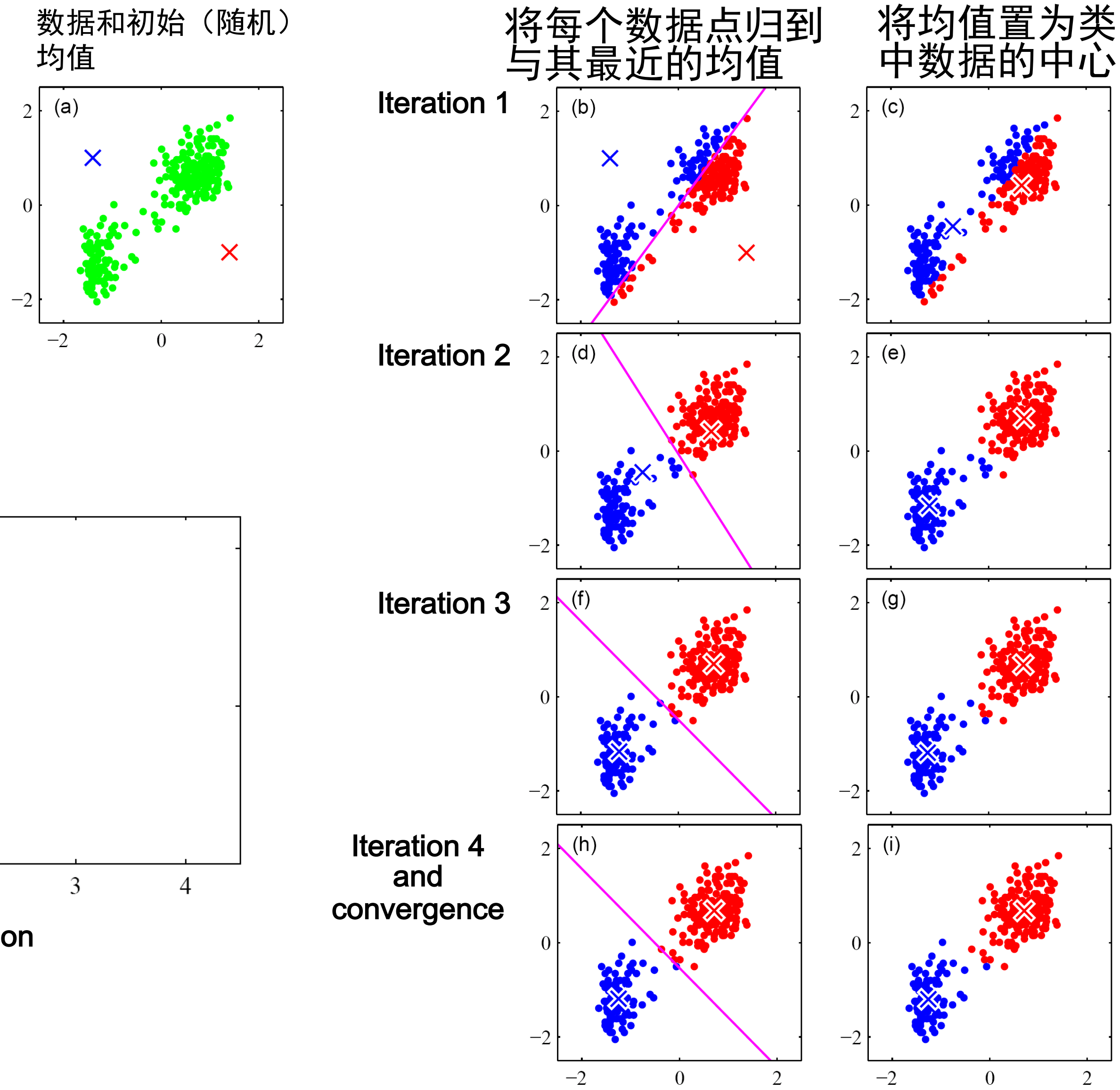
$$\boldsymbol{\mu}_k^{(t+1)} = \arg\min_{\boldsymbol{\mu}} \sum_{i \in C(k)} \left(\boldsymbol{\mu} - x_i\right)^2$$

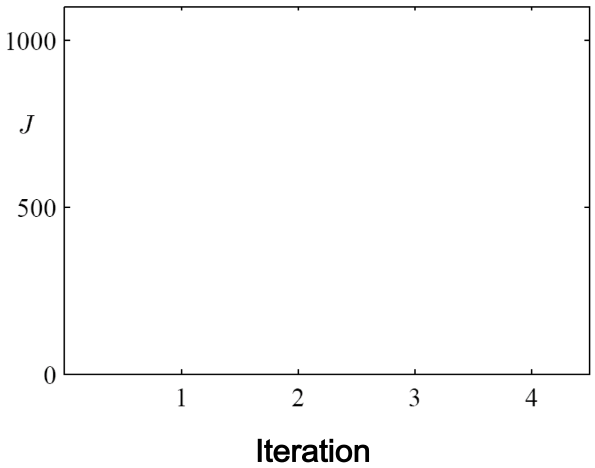
  – 等价于μ_i 为类中数据的均值

- 5、重复3~4步，直到数据点所属类别类不再改变



例：


数据和初始（随机）
均值
将每个数据点归到
与其最近的均值
将均值置为类
中数据的中心
Iteration 1
Iteration 2
Iteration 3
Iteration 4
and
convergence
(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)
(i)
2
0
−2
−2
0
2


1000
J
500
0
1
2
3
4
Iteration

► K-means的优化目标

- 均值$\boldsymbol{\mu}$和数据点所属集合C的势能函数为

$$F(\boldsymbol{\mu}, C) = \sum_{i=1}^{N} \left(\boldsymbol{\mu}_{C(i)} - \mathbf{x}_i\right)^2$$

平方误差损失

  – $C(i)$: 样本点i所属的簇

- K-means的优化目标

$$\min_{\boldsymbol{\mu}} \min_{C} F(\boldsymbol{\mu}, C)$$

► *K*-means的收敛性

- 优化势能函数

$$\min_{\boldsymbol{\mu}} \min_{C} F(\boldsymbol{\mu}, C) = \min_{\boldsymbol{\mu}} \min_{C} \sum_{k=1}^{K} \sum_{i \in C(k)} (\boldsymbol{\mu}_k - \mathbf{x}_i)^2$$

- 固定**μ**，优化C

- 固定C ，优化**μ**

▶ K-means的收敛性

- 求 $\min_a \min_b F(a,b)$

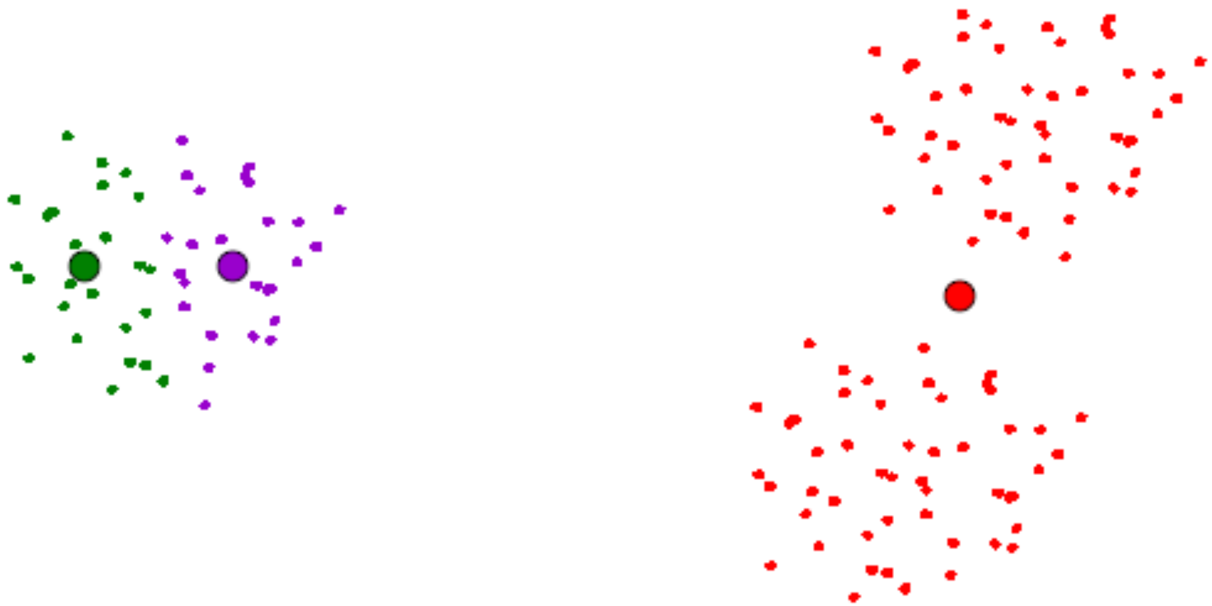
$$\min_{\boldsymbol{\mu}} \min_C F(\boldsymbol{\mu}, C) = \min_{\boldsymbol{\mu}} \min_C \sum_{k=1}^{K} \sum_{i \in C(k)} (\boldsymbol{\mu}_k - \mathbf{x}_i)^2$$

- 坐标下降(coordinate ascent)
  - 固定a，最小化b
  - 固定b，最小化a
  - 重复
- 收敛性
  - 若F有界，会收敛到一个局部极小

- K-means为坐标下降算法!

K-means会找到最优结果？

- K-means不保证达到全局最优（收敛，但没有达到全局最优的情况）
  - 例：K=3

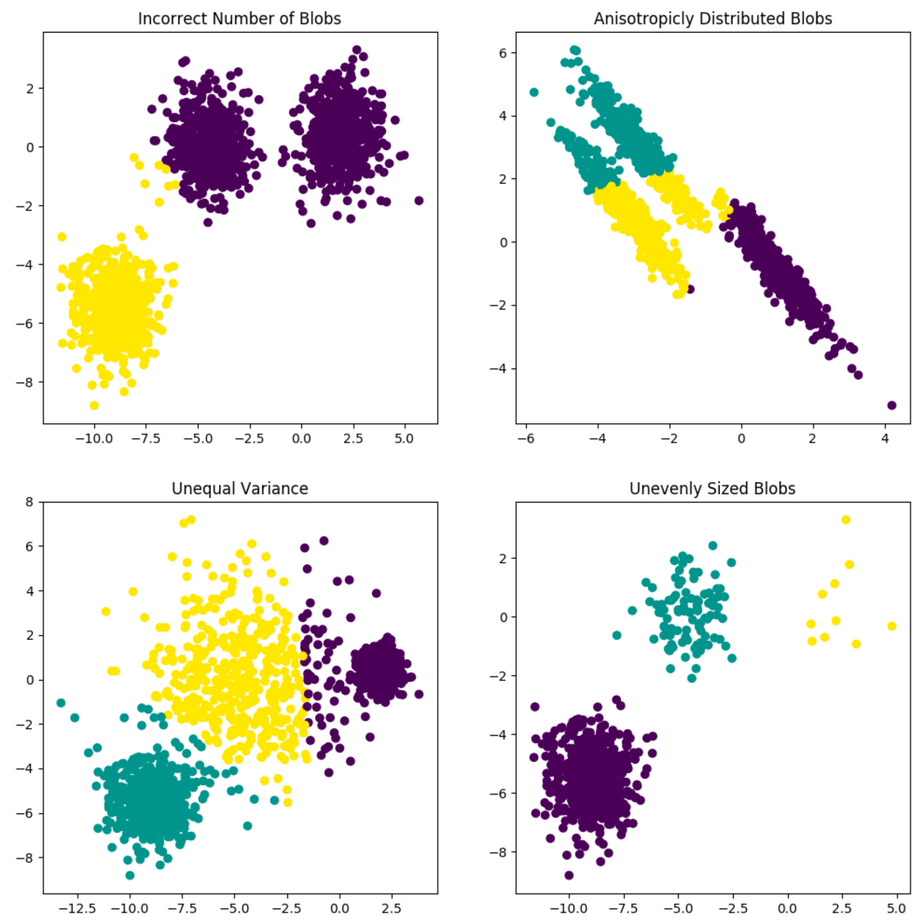


▶解决方案

- 仔细寻找初始值
  - 随机确定第一个类的中心，其他类中心的位置尽量远离已有类的中心
  - Scikit learn中*K*-means实现中参数（ init='k-means++' ）将初始化 centroids （质心）彼此远离，得到比随机初始化更好的结果。

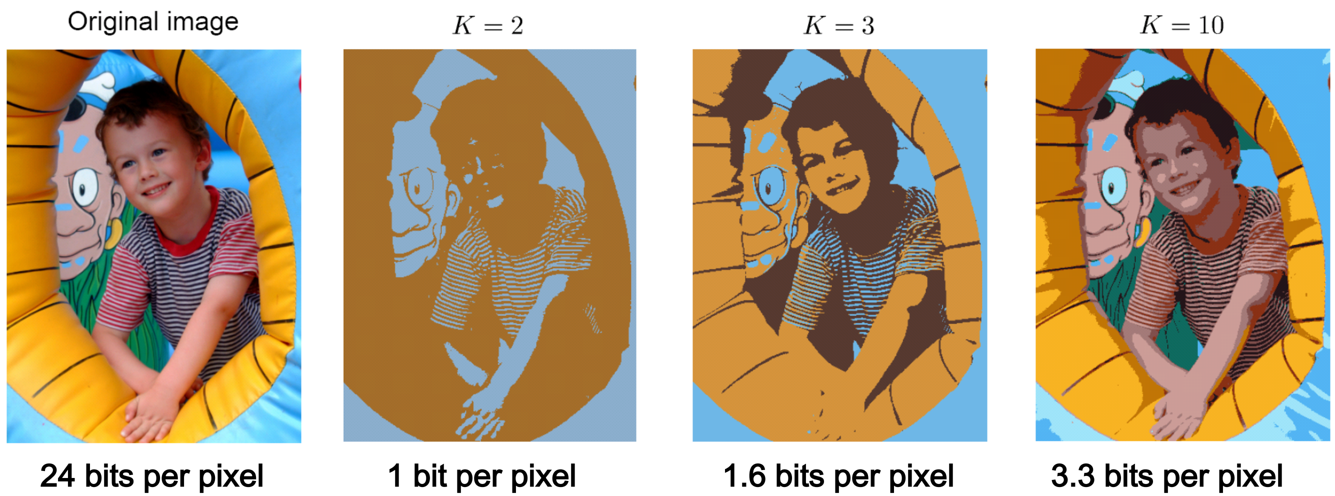
- 重复多次（如10次），每次初始值不同，最后选择使目标函数最小的结果

▶ K-means失败的场景



例：颜色量化

- 将每个像素视为一3-D 向量(RGB) 并采用K-means 聚类，找到K 个”代表颜色”
- 每个像素用$\log_2(K)$ 比特存储，颜色质量会损失



K-means

- 可以证明，*K*-means可以解释为硬分类时的球形高斯混合模型（各类的协方差相等），这使得可以将*K*-mean参数化
- 在数据压缩中，亦称为向量量化(Vector Quantization)
  - 最小化平方误差损失函数亦称为最小化失真

K-centers聚类

- 亦被称为K-median聚类、 K-mediods聚类
- 与K-means类似，只是将K-means聚类中的均值换成中值
  - 均值极有可能是一个不存在的样本点，不足以代表该簇中的样本，而中值是一个样本集合中真实存在的一个样本点
  - 同时中值相对均值而言，对噪声（孤立点、离群点）不那么敏感

▶ 非向量空间的聚类

对*K*-means聚类算法（和高斯混合模型（GMM））

- 数据必须在某个向量空间，这样欧氏距离是一个很自然的相似性度量
- 也可以采用类似核函数$k(x_i,x_k)$ 的方式来度量相似性，然后基于这种方法的进行聚类
  - 用$s(i,k)$表示 x_i 与x_k 之间的相似性（与核函数类似，但形式上不必相同）

K-means聚类的优点

- 一种经典的聚类算法，简单、快速
- 能处理大规模数据，可扩展型好
- 当簇接近高斯分布时，效果较好

▶ *K*-means聚类的缺点

不止于代码

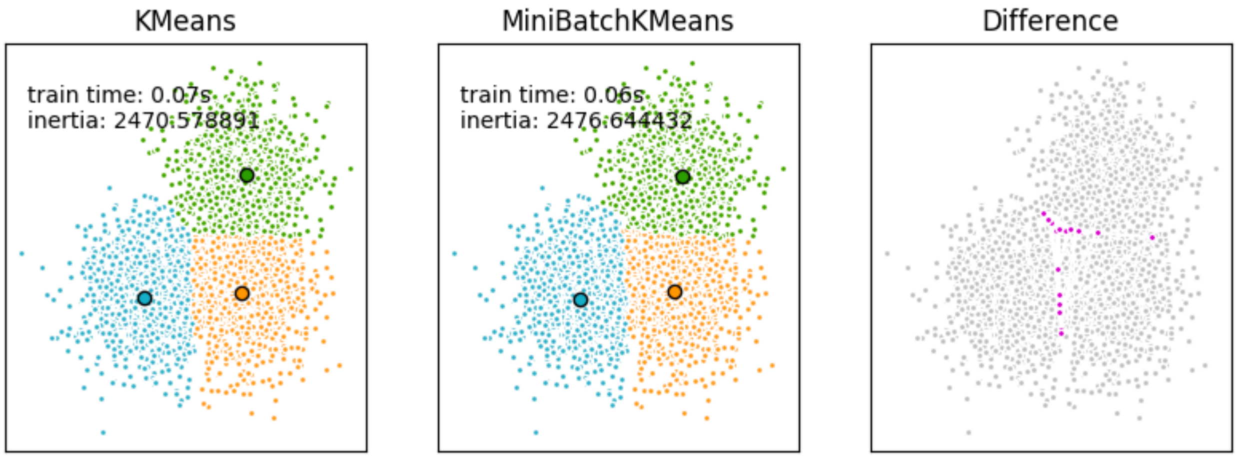
- 假设簇是凸且各向同性（通常与实际情况不符）。当簇的形状被拉长或是不规则的流形时，效果不好。
- 本质上不是一个归一化度量（ normalized metric ）: 我们只知道较低的值更好，并且零是最优的。
- 是在高维空间中，欧氏距离往往会变得膨胀（维度灾难）。在 *K*-means 聚类之前运行诸如 PCA 之类的降维算法可以减轻这个问题并加快计算速度。
- 要事先确定簇数*K*

Scikit learn中的*K*-means实现

- *class* sklearn.cluster.KMeans(*n_clusters=8, init='kmeans++', n_init=10, max_iter=300, tol=0.0001, precompute_distances='auto', verbose=0, random_state=None, copy_x=True, n_jobs=1, algorithm='auto')*
  - *n_clusters* ：聚类数量，默认为8
  - *init:* 初始化质心，可以是'k-means++'、 'random' 或者数组
  - *n_init:* 重复次数。为了弥补初始质心的影响而使算法陷入局部极值点，算法默认会初始10个质心，实现聚类，然后返回多次聚类的最好结果。
  - *precompute_distances* ： 这个参数会在空间和时间之间做权衡
    - True ：存储样本之间的距离（快但是占用更多内存）
    - False：不预计算存储样本之间的距离
    - auto ：在数据样本大于featurs*samples 的数量大于12M时设置为False

▶ Mini Batch *K*-Means

- MiniBatchKMeans 是 *K*-Means 算法的一个变体：
  - 使用 mini-batches （小批量）减少计算时间
  - 目标函数相同
  - 收敛速度比 *K*-Means 快，但是结果的质量会降低（在实践中，质量差异可能相当小）



THANK YOU