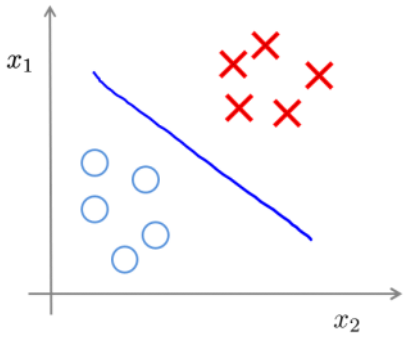
13 聚类

2022年11月24日　　21:20

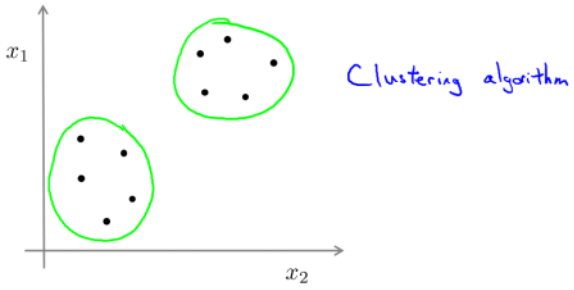
13.1无监督学习



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

我们前面已经了解过，监督学习是对带有标签的训练集进行学习，用假设函数来拟合。

Unsupervised learning



Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

而无监督学习，顾名思义其学习的数据集并不带有标签。我们需要让算法找到一些隐含在数据中的结构，然后进行分簇或者说聚类。

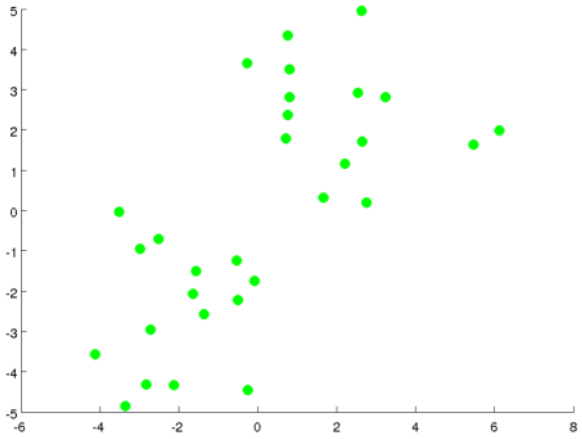
聚类的一些应用在开始的课程中已经提到过。

13.2K-means算法

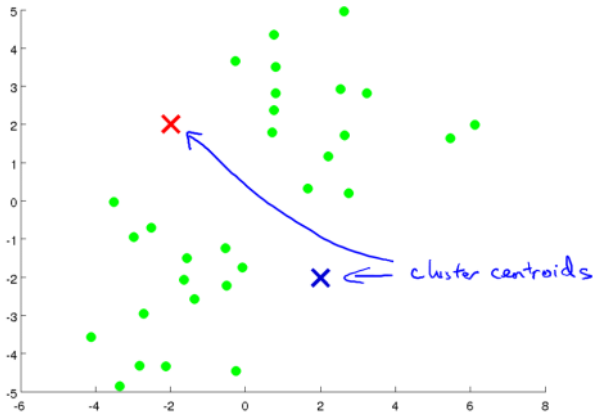
k均值算法是一种广泛运用的聚类算法。

其具体步骤如下所示：

首先，我们需要把一个数据集进行聚类

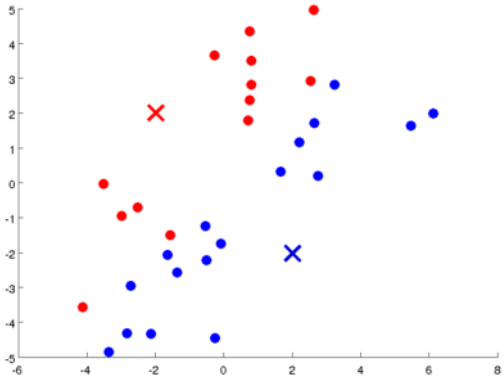


第一步先随机生成两点：



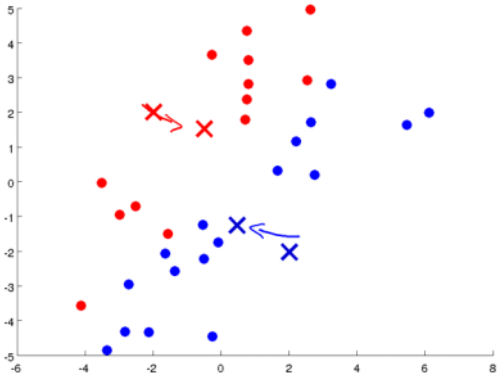
这两个点叫做聚类中心(Cluster centroids)。而K均值算法为一个迭代算法，接下来它要做的两件事分别为簇分配(cluster assignment)，第二个是移动聚类中心(move centroid)。

具体而言就是算法通过循环迭代对每一个数据点进行判断，看它是离红色的点更近还是蓝色的点更近。



把接近红色中心点的点染成红色，把接近蓝色中心点的点染成蓝色。

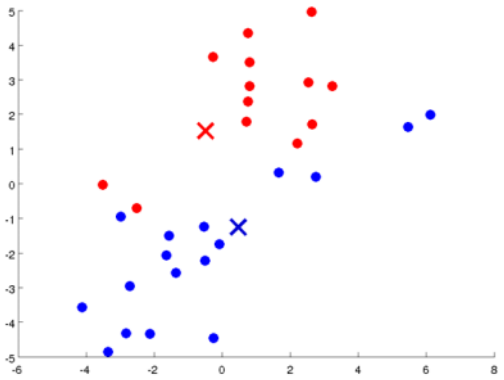
移动聚类中心就是将两个中心点移动到分成的两簇的点的均值中心。



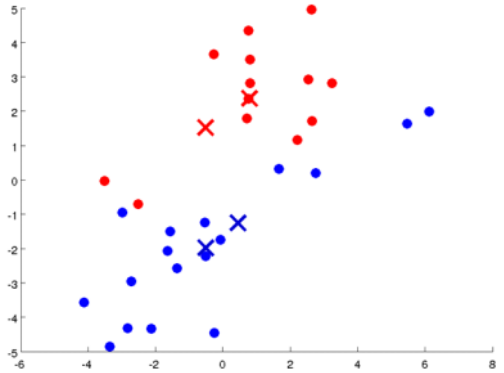
从而移动到新的位置。

重复上述步骤。

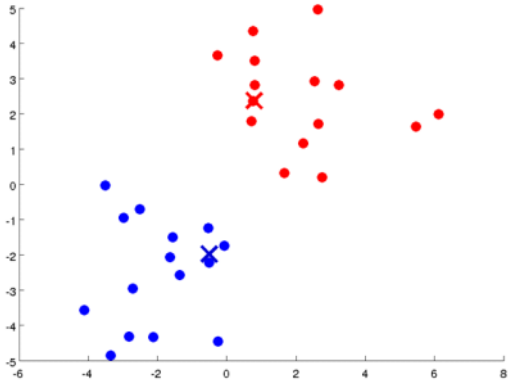
先簇分配，

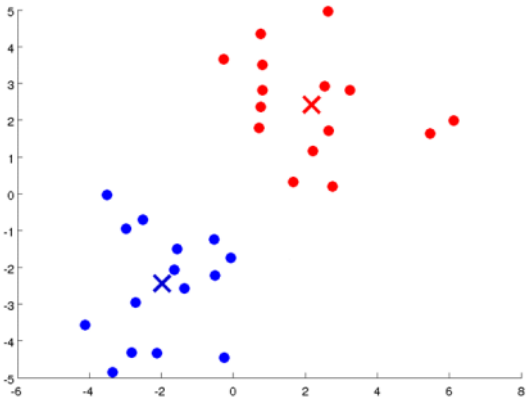


再移动聚类中心，



重复以往：





直到聚类中心和点的颜色都不会再变化了。

综上，K-means算法如下：

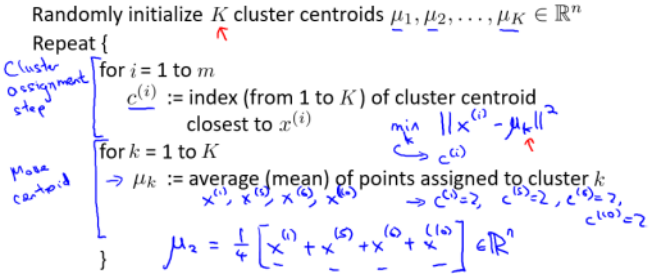
K-means algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

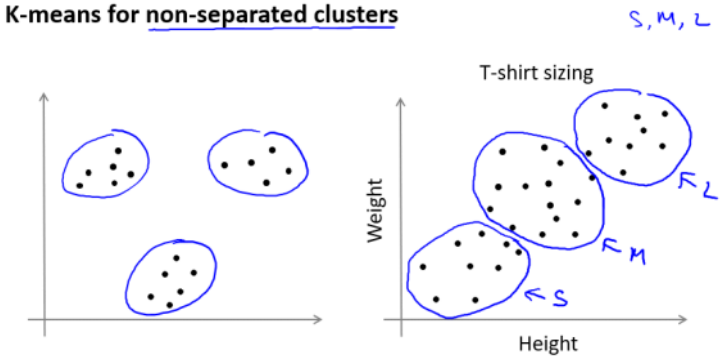
$x^{(i)} \in \mathbb{R}^n$ (drop $x_0 = 1$ convention)

首先，输入我们想聚类的簇数K，和一系列用x表示的无标签的数据集。



而后，随机初始化k个聚类中心。然后将所有点离哪一个中心点更近进行索引，也即簇分配；再然后计算每一个中心点与其分配的簇的距离均值。特别地，如果有中心点没有被分配到一个点，那么常见的做法是将这个中心点移除，移除后得到的分类簇数为K-1。

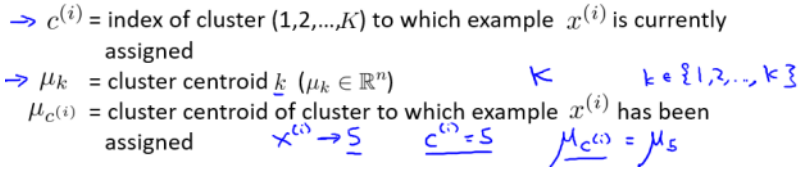
K-均值算法的另一个常见用法：解决分离不佳的簇的问题。



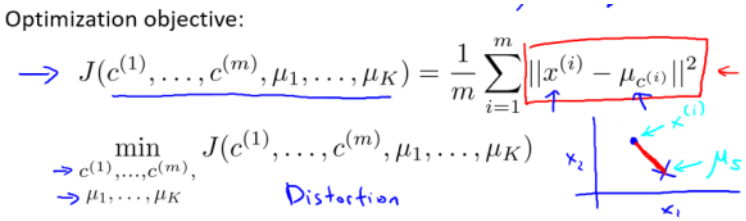
对于如上图右所示的不易分类簇的问题，假如我是一个T-shirt的生产商，我已有一些用户的身高和体重数据，如何将我的T-shirt分成三个尺码，即这三个尺码应适应哪三类人群。一种办法就是对其执行k均值算法，而后将分好的三类人的身材范围作为T-shirt的三种身形依据。

13.3优化目标

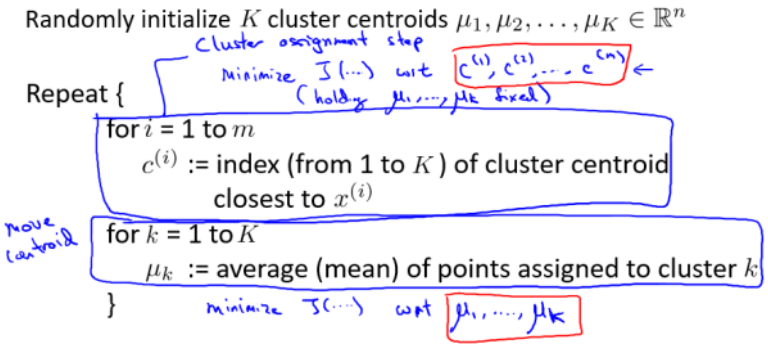
k均值算法和之前我们学习过的监督学习算法一样也有一个用于最小化的代价函数。
在执行k均值算法的时候，需要对两组变量进行跟踪：一组是每一个点离最近的中心点的索引；另一组是聚类的中心点。



现在定义一组新的变量，它表示x_i所属的那个簇的聚类中心。
由此写出聚类算法的优化目标：



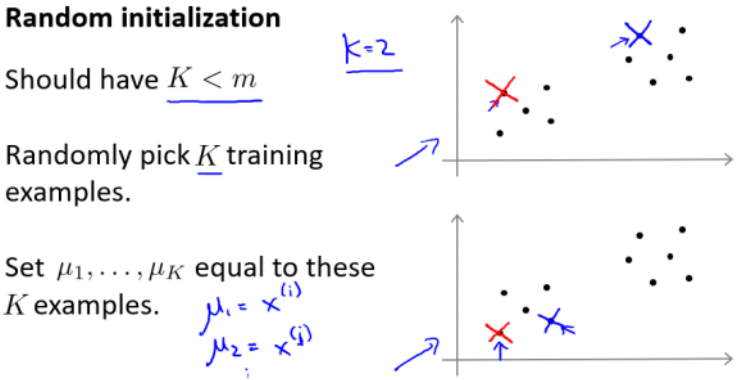
也即表示每一个点到离它最近的聚类中心的距离的均值。它有时也被叫做失真(Distoction)代价函数或K均值算法的失真。



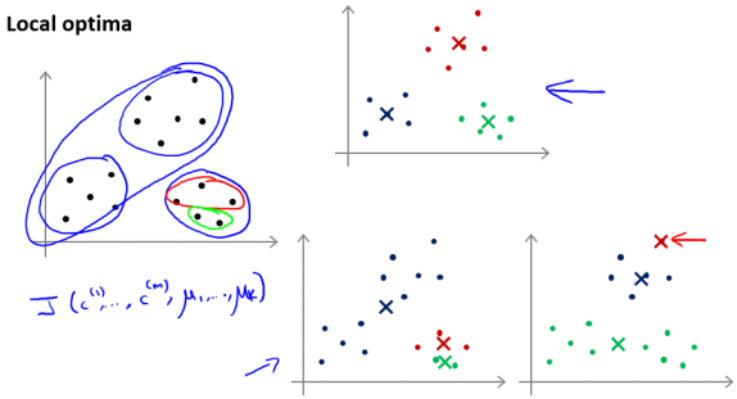
所以上节提到的算法的第一步，其实是在不改变聚类中心位置的前提下，最小化代价函数；算法的第二步，则是通过移动聚类中心来最小化代价函数。

13.4随机初始化

有一种值得推荐的方法对聚类算法进行初始化。



如果要运行一个聚类算法，那么初始化的聚类数目应当小于样本数目m。
然后随机挑选几个训练样本设定为μ_1到μ_k作为初始化的聚类中心点。初始化的中心点不同可能导致不同的运算结果。
局部最优是很可能出现的一种情况。



局部最优就是指畸变的代价函数落在了局部的最优值处，如上图的下面两种情况，它们似乎不是最好的分类结果。
一种避免局部最优发生的办法就是多次随机初始化，而不是仅仅初始化一次。以下是具体做法：

For i = 1 to 100 {

Randomly initialize K-means.
Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.
Compute cost function (distortion)
$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$
}

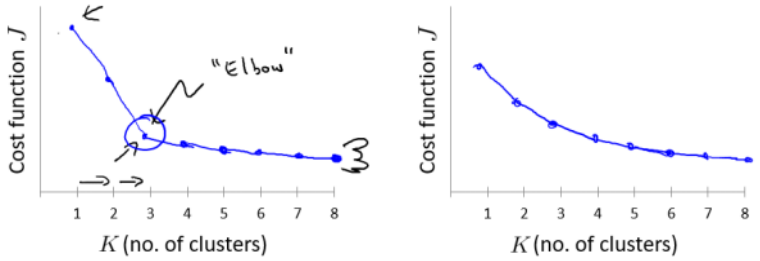
Pick clustering that gave lowest cost $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

对算法运行50-100次，每一次都随机初始化K均值算法，并得到一系列的结果，最后计

算代价函数。最终选择代价值也即畸变值最小的一种情况。如果聚类数很少的话，如2-10，通常能保证找到较好的局部最优解；而如果K很大的话，即使是成千上万次的运算也不一定会对算法的结果有所改善，更有可能我们第一次初始化的结果就已经算很好了。

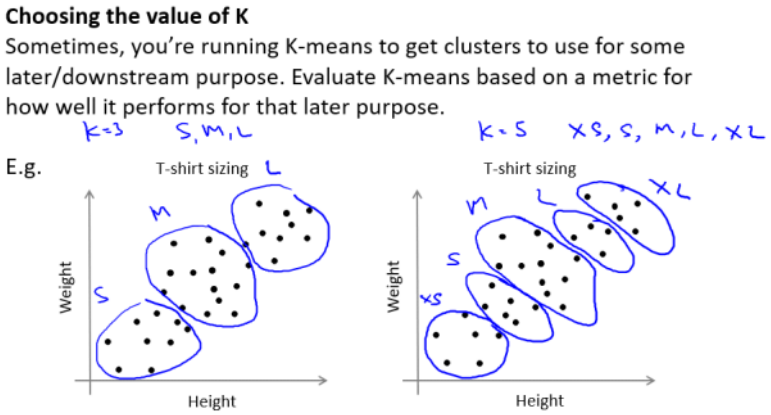
13.5选择聚类数量

数据集中有多少个聚类往往是不清楚的。用自动化的方法选择聚类数量很困难。
肘部法则(Elbow Method)：改变K计算相应的代价函数或者说畸变数



随着K值的增加，畸变值下降。而曲线会出现一个肘部(elbow)。在肘部之前，畸变值下降迅速，而肘部之后，畸变值下降缓慢。这个肘部可能是一个比较合适的点。
但是这个法则也并不是那么常用。比如上图右的情况，实际问题中曲线可能会比较平滑和模糊，不易判断出肘部。

最后，还有一种思路。



依照我们聚类的目的，我们可能关注于上游或下游甚至于之前所说的T-shirt尺码划分的市场分割问题。那么对于聚类的数目可以根据后续目的来选取。依然以T恤为例：T恤尺寸的划分可能有很多种，从商业角度来看哪一种分类方式更能满足客户需求以及做到商业效益的最大化。