

## Kmeans++聚类算法原理与实现

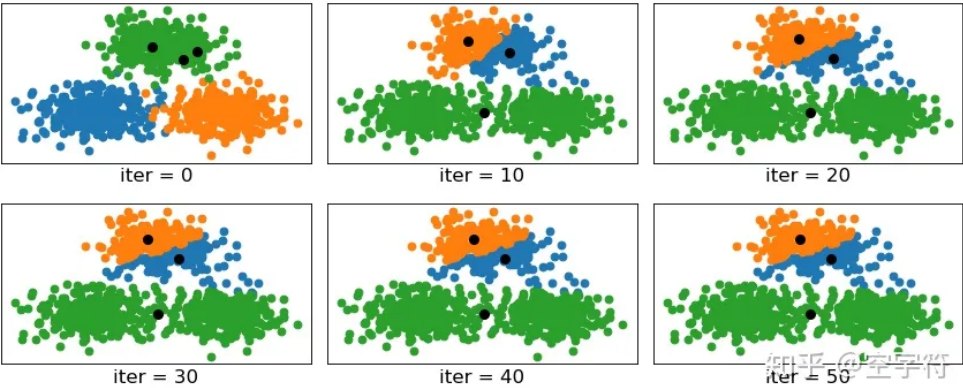


月来客栈

96 人赞同了该文章

### 1 引例

在上一篇文章中，笔者介绍了什么是聚类算法，并且同时还介绍了聚类算法中应用最为广泛的 *Kmeans* 聚类算法。从 *Kmeans* 聚类算法的原理可知，*Kmeans* 在正式聚类之前首先需要完成的就是初始化  $k$  个簇中心。同时，也正是因为这个原因，使得 *Kmeans* 聚类算法存在着一个巨大的缺陷——收敛情况严重依赖于簇中心的初始化状况。试想一下，如果在初始化过程中很不巧的将  $k$  个（或大多数）簇中心都初始化到了到同一个簇中，那么在这种情况下 *Kmeans* 聚类算法很大程度上都不会收敛到全局最小值。也就是说，当簇中心初始化的位置不得当时，聚类结果将会出现严重的错误。



如图所示的数据集仍旧是在上一篇文章中所用到的人造数据集，不同的是在进行聚类时我们人为的将三个簇中心初始化到了到一个簇中。其中  $\text{iter}=0$  时的情况为未开始聚类前根据每个样本的真实簇标签所展示的情况，其中蓝色、绿色和橙色分别表示三个簇的分布情况，三个黑色圆点为初始化的簇中心。从上图中可以发现，*Kmeans* 在进行完第10次迭代后，将最上面的一个簇分为了两个簇，将下面的两个簇划分成了一个簇。同时，当进行完第30次迭代后，可以发现算法已经开始收敛，后续簇中心并没有发生任何变化。最终的结果仍旧是将上面一个簇分为两个，下面两个簇划分为一个。

通过上面这个例子我们知道，初始簇中心的位置会严重影响到 *Kmeans* 聚类算法的最终结果，那么我们应该怎么最大可能的避免这种情况呢？此时就开始轮到 *Kmeans++* 算法登场了。

### 2 Kmeans++聚类算法

式上做了改进，其它地方同 $Kmeans$ 聚类算法一样。

## 2.1 Kmeans++原理

$Kmeans++$ 在初始化簇中心时的方法总结成一句话就是：\*\*逐个选取 $k$ 个簇中心，且离其它簇中心越远的样本点越有可能被选为下一个簇中心。\*\*其具体做法如下（其中引用英文部分论文原文）：

①从数据集 $\mathcal{X}$ 中随机（均匀分布）选取一个样本点作为第一个初始聚类中心 $c_i$ ；

1a. Take one center  $c_1$ , chosen uniformly at random from  $\mathcal{X}$ .

②接着计算每个样本与当前已有聚类中心之间的最短距离，用 $D(x)$ 表示；然后计算每个样本点被选为下一个聚类中心的概率 $P(x)$ ，最后选择最大概率值（或者概率分布）所对应的样本点作为下一个簇中心；

1b. Take a new center  $c_i$ , choosing  $x \in \mathcal{X}$  with probability  $P(x)$ .

$$P(x) = \frac{D(x)^2}{\sum_{x \in \mathcal{X}} D(x)^2} \quad (1)$$

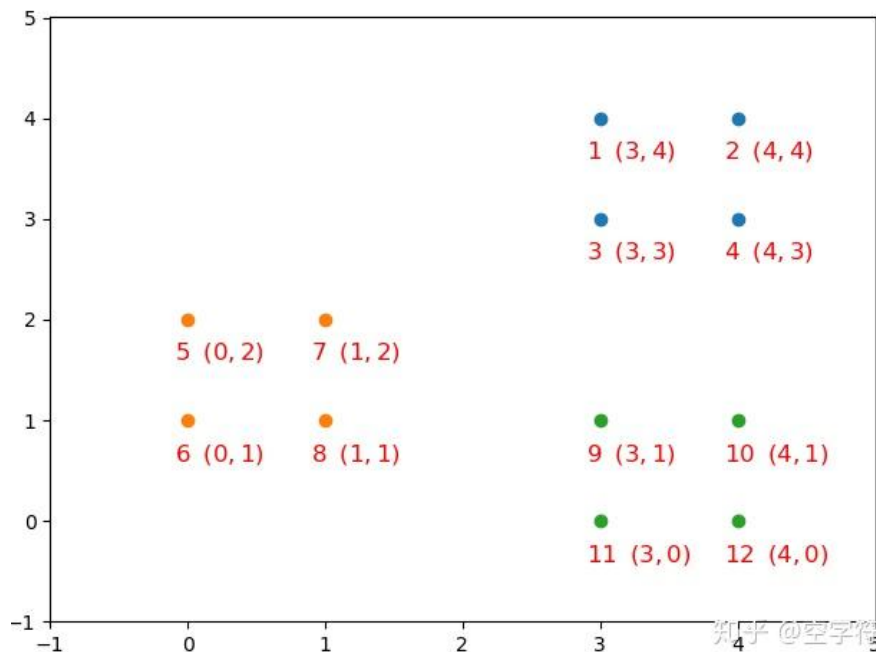
③重复第②步，直到选择出 $k$ 个聚类中心；

1c. Repeat Step 1b. until we have taken  $k$  centers altogether.

从公式(1)也可以看成，距离现有簇中心越远的样本点，越可能被选为下一个簇中心。

## 2.2 计算示例

在上面的内容中，我们已经介绍了 $Kmeans++$ 聚类算法在初始化簇中心时的具体步骤，不过仅仅只是列出公式显然不是本系列文章的风格。下面，我们就通过一个例子来实际计算一下簇中心的选择过程。



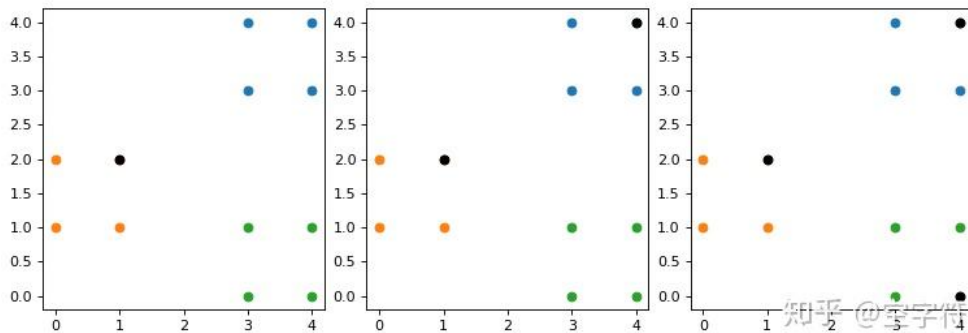
如图所示为所有的样本点，且很明显的就能看成一共包含有3个簇，这也就意味着我们需要找到3个簇中心。如果我们随机选择初始簇中心为点(1, 2)，那么在进行第

编号	1	2	3	4	5	6	7	8	9	10	11	12
$D(x)^2$	8	13	5	10	1	2	0	1	5	10	8	13

从表中可以看出，离7号样本点最远的是2号和12号样本点（其实从图中也可以看出），因此 **Kmeans++** 就会选择2号样本点为下一个聚类中心。接着，我们再次重复步骤二又能得到下面这个表格：

编号	1	2	3	4	5	6	7	8	9	10	11	12
$D(x)^2$	1	0	2	1	1	2	0	1	5	9	8	13

从表中可以看出，离2号和7号样本点最远的是12号样本点，所以下一个簇中心就会是12号样本点。进一步，我们可以得到如下可视化结果：



## 2.3 编程实现

由于 **Kmeans++** 算法仅仅只是在簇中心初始化方法上做了改变，因此只需要重写该函数即可。

```
def InitialCentroid(x, K):
    c0_idx = int(np.random.uniform(0, len(x)))
    centroid = x[c0_idx].reshape(1, -1) # 选择第一个簇中心
    k = 1
    n = x.shape[0]
    while k < K:
        d2 = []
        for i in range(n):
            subs = centroid - x[i, :]
            dimension2 = np.power(subs, 2)
            dimension_s = np.sum(dimension2, axis=1) # sum of each row
            d2.append(np.min(dimension_s))

        # ---- 直接选择概率值最大的 ----
        # new_c_idx = np.argmax(d2)
        # ---- 依照概率分布进行选择 ----
        prob = np.array(d2) / np.sum(np.array(d2))
        new_c_idx = np.random.choice(n, p=prob)

        centroid = np.vstack([centroid, x[new_c_idx]])
        k += 1
    return centroid
```

```
# y_pred = kmeans(x, y)
y_pred = kmeanspp_visual(x,y,K)
nmi = normalized_mutual_info_score(y, y_pred)
print("NMI by ours: ", nmi)

model = KMeans(n_clusters=K, init='k-means++')
model.fit(x)
y_pred = model.predict(x)
nmi = normalized_mutual_info_score(y, y_pred)
print("NMI by sklearn: ", nmi)
```

#结果:

```
# NMI by ours: 0.9456935014894648
# centroids:
# [[3.79243249 1.00756396]
#  [2.20105974 1.00882189]
#  [2.99200232 2.81003461]]
# NMI by sklearn: 0.9456935014894648
# centroids:
# [[2.99200232 2.81003461]
#  [2.20105974 1.00882189]
#  [3.79243249 1.00756396]]
```

最终，我们再次将先前的人造模拟数据进行聚类便可以得到如下可视化结果：

从图示中可以看到，初始化的簇中心彼此相距都十分的远，从而不可能再发生初始簇中心在同一个簇中的情况。同时需要说明的是，sklearn中Kmeans聚类算法的默认中心选择方式就是通过Kmeans++的原理来实现的，通过参数 `init=k-means++` 来控制。

到此为止，我们就介绍完了Kmeans++聚类算法的主要思想。

### 3 总结

在本篇文章中，笔者首先介绍了Kmeans聚类算法在初始化簇中心上的弊端；然后介绍了Kmeans++这种聚类算法对初始化簇中心的改进，当然改进方法也不止一种；最后还介绍了如何通过python来编码实现初始化方法，同时还对聚类结果进行了可视化。本次内容就到此结束，感谢阅读！

若有任何疑问与见解，请发邮件至moon-hotel@hotmail.com并附上文章链接，青山不改，绿水长流，月来客栈见！

[2]示例代码：[github.com/moon-hotel/M...](https://github.com/moon-hotel/M...)

近期文章

[zhuanlan.zhihu.com/p/15...](https://zhuanlan.zhihu.com/p/15...)

[2]原来这就是支持向量机

[3]朴素贝叶斯算法

编辑于 2023-05-18 20:28 · IP 属地上海

[聚类算法](#) [聚类](#) [聚类分析](#)

写下你的评论...

43 条评论

默认 最新



6点要吃饭

请问为什么要计算概率？分母都一样的话，直接用最大距离的点作为下一个中心不是和“取概率最大的对应的点”结果一样嘛？还可以少一步计算。

2021-05-29

回复 3



月来客栈 作者

1，为了保持和论文描述的一致性，所以这里依旧介绍的是概率；2，在实际的代码实现过程中就直接使用的是距离。

2021-05-29

回复 3



可为为 tensorlu

你说的是对的，作者这里理解错了

05-08

回复 赞

展开其他 3 条回复 >



evol

呜呜呜，我的散点图也是迭代到30次就开始线条状的切割，网上找了半天没答案，我还以为自己K-means算法写错了，原来是初始选取问题，谢谢作者大大了！

2021-05-07

回复 1



锦恢

很优雅的解释

2021-04-14

回复 1



Lyve

朋友代码都很工整 感谢感谢

2021-03-08

回复 1



月来客栈 作者

添加了根据概率分布来选择下一个簇中心

05-18

回复 赞



nothingthere

最远点采样

05-17

回复 赞

 回复  赞

● 回复 ● 赞

回复 赞

● 回复 ● 赞

● 回复 ● 赞

● 回复 ● 赞

回复 赞

回复 赞

回复 1

● 回复 ● 赞

 回复
  赞

● 回复 ● 赞


回复 赞

 回复  赞

回复 赞

**ckh**

6/8


月来客栈 1F 自

论文里面虽然没有明确说明，但是分析也能干得出。 $D(x)$ 表示当前样本离已有簇中心的最近距离，由于大家的分母都一样所以只看分子即可。如果选择 $D(x)$ 最大者所对应的样本作为下一个簇中心，实际上就等价于在最近距离中寻找最远的那一个样本点作为下一个簇中心。相反，如果选择 $D(x)$ 最小的对应的样本点为下一个簇中心，那就在最近距离中寻找最近的样本点，而这显然不符合算法的思想，因为这样找到的簇中心肯定会出现多个簇中心在一起的情况。因此应该选择 $D(x)$ 最大所对应的样本点。

2021-04-03

回复

赞


yang

请问算法描述中有提到是根据概率来选择样本的，代码中没有体现这点。有体现根据概率选择的代码吗？

2020-11-24

回复

赞


月来客栈 作者

由于分母是一样的，所以在实现的时候只考虑分子就行了。分子除以分母就是概率

2020-11-24

回复

赞


空白格了

你好，这个我按照这个数据写了一下，选出来的  
第一个点是 (1, 1) 第二个点是 (4, 4) 第三个是 (0, 1) ，这种情况怎么处理

2020-11-12

回复

赞


月来客栈 作者 > 不愿透露姓名的付同学

这个没有。这个关键在于对数据的处理吧，你可以直接reshape成向量然后聚类。

2020-11-21

回复

赞

不愿透露姓名的付同学 > 月来客栈


博主你好，有没有关于目标检测聚类的k-means++代码呢

2020-11-20

回复

赞

展开其他 1 条回复 >


MikatsukiSora

请问代码中的np.nan\_to\_num有什么用呢？

2022-06-01

回复

赞

MikatsukiSora

还有那个np.where () 返回的是啥？为什么后面要加[0][0]

2022-06-01

回复

赞


月来客栈 作者 > MikatsukiSora

实际去调一调代码就会知道

2022-06-02

回复

赞

瞬时速度

最后不应该用轮盘赌选择吗，只是概率大的选中概率大一些，不一定是最大的那个吧

2022-05-31

回复

赞


月来客栈 作者

你可以看一下原文

2022-05-31

回复

赞

h1236822

请问楼主，使用kmeans++如何确定最优的k呢？之前用kmeans时候都是先用手肘图来判断k的取值的

2022-04-07

回复

赞

月来客栈 作者

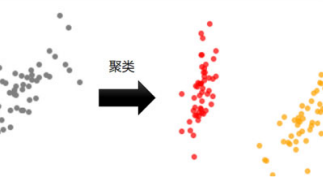
也可以那样去判断，kmeans++仅仅是初始化簇中心的策略变了，其他都一样

写下你的评论



跟我一起机器学习

推荐阅读



第二个算法-Kmeans聚类算法  
(含Python代码和实例)

姚在路上

Kmeans 聚类算法

什么是聚类聚类可以将相似的样本归到同一簇中，是无监督学习中最常见的问题。聚类的结果是，簇内的对象越相似，聚类的效果就越好。一个簇的意思就是一个类别。以人学习习题为例，仔细观察...

凌晨伴太阳

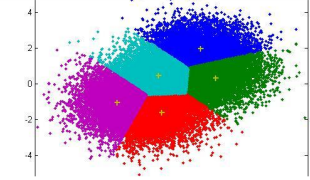
发表于黑洞科技

聚类算法kmeans及  
kmeans++介绍(含python...

1、kmeanskmeans, k-均值聚类算法，能够实现发现数据集的 k 个簇的算法，每个簇通过其质心来描述。kmeans步骤：（1）随机找 k 个点作为质心（种子）；（2）计算其他点到这 k 个种子的距...

来咯兔子

发表于常见机器学...



聚类算法 | KMeans

飞鱼Talk