Big Data Processing: Homework 4

凌康伟 5140219295

May 15, 2017

1 程序说明

`kmeans` 算法使用 C 语言实现，可视化采用 python(jupyter notebook)。`kmeans` 代码为 src 目录下所有.h .c 文件。可视化 notebook 位于 visualize 文件夹。

输入 `make` 编译生成 main，通过 `./main 3` 运行程序（需要给定参数：cluster 数量）。

2 可视化结果

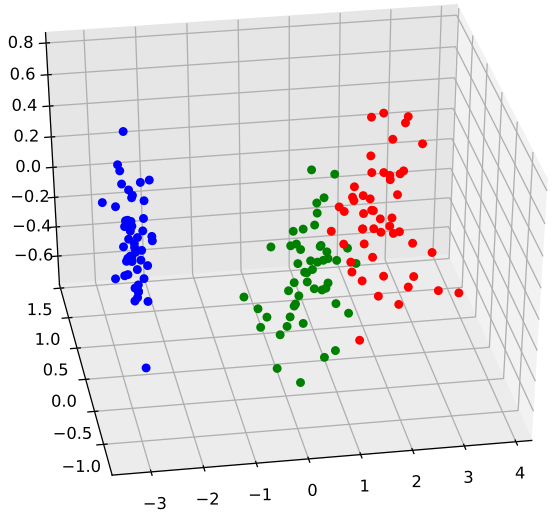

Figure 1: 原数据集可视化

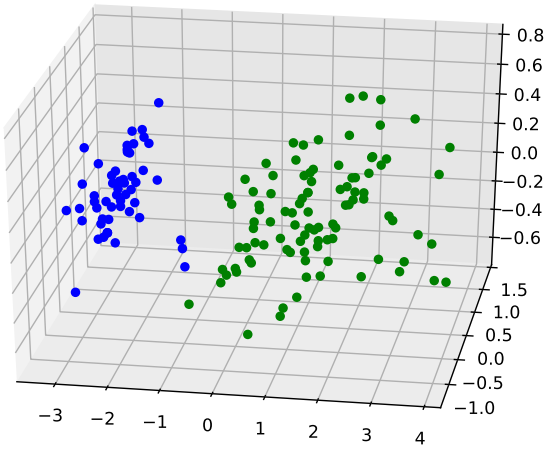

Figure 2: cluster 数为 2

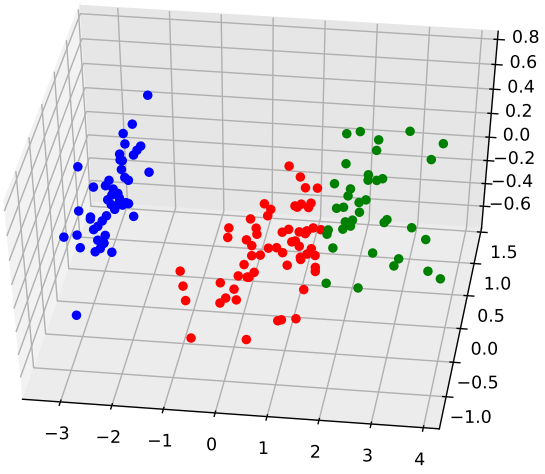

Figure 3: cluster 数为 3

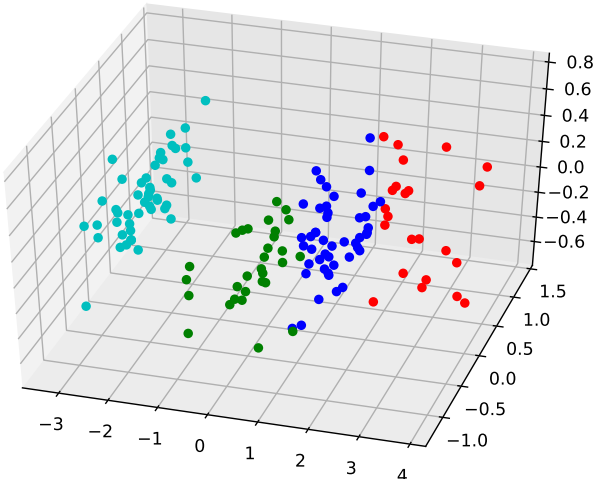

Figure 4: cluster 数为 4

3 结果分析

3.1 cluster 数对聚类效果的影响

由以上各图与原数据集可视化相比较，可以看出，

1. 当 cluster 数较小时 (2 clusters)，得到的聚类结果不能很好的刻画原数据集，两个类分得的数据点不平衡，且密集程度有差别，在临界上的点实际上不能被化为任意一个集群。
2. 当 cluster 数较大时 (4 clusters), 由于原数据本身并没有这么多类，强行聚类出来的结果显示出得到的集群有的间距非常大，也有间距非常小，其中的数据点并无太大区别。
3. cluster 数适中 (3 clusters), 得到的结果与原数据集分类非常相近，效果良好。

从平均距离 (average distance) 来看, cluster 数为 3 时，也是 average distance 的拐点。

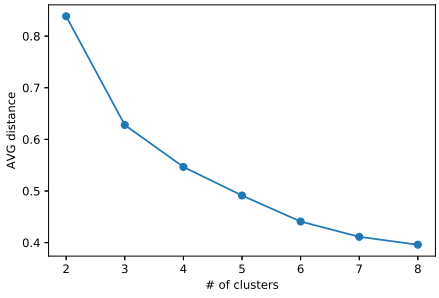

Figure 5: average distance - cluster curve

3.2 初始 seeds 对聚类效果的影响

之前采用的是随机从数据集中抽取 K 个点作为初始的 centroids。以下是以直接数据集中前 K 个作为初始 seed，cluster 数为 3 的结果。

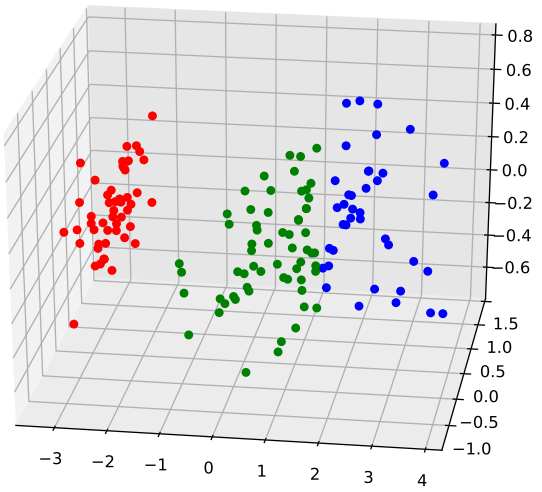

Figure 6: K initial points as centroids, 3 clusters

可以看到，结果与之前的结果相近，但是这种方法相对随机方法来说更加稳定，不会因为初始随机中心的选取而影响运行时间。但是，有些时候，通过直接选取前 K 个作为中心可能会导致部分初始中心在同一个地方，虽然经过迭代可以分开，但是偶尔会导致部分聚类中没有任何元素，下图是将初始 K 个中心都设为第一个数据点的结果。

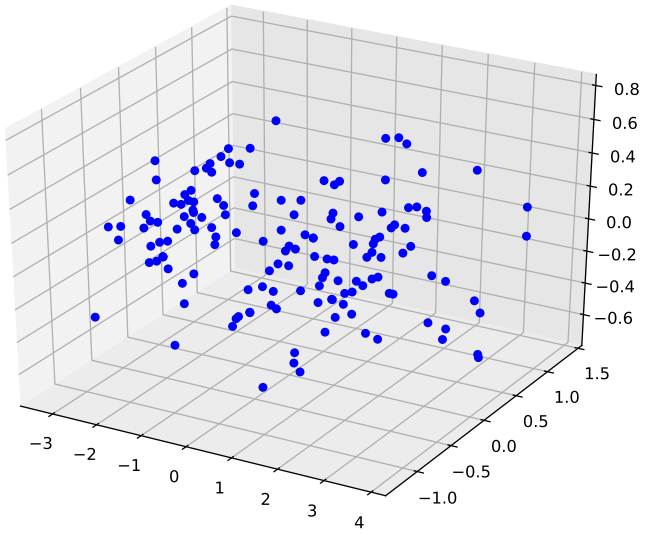

Figure 7: same initial point as centroids, 3 clusters

3.3 运行时间

3.3.1 Cluster 数

随机初始化对 `kmeans` 的迭代次数有很大影响，因此要取多次（500）求平均，结果如下：

# of clusters	2	3	4	5	6	7	8
time / ms	0.04	0.076	0.121	0.132	0.156	0.188	0.209

显然是 cluster 越多，需要的时间越长。这个运行时间是与迭代的次数相关的，不同的初始中心会导致需要的迭代次数不尽相同。

3.3.2 Sample 数

用 `split.py` 将数据集均匀分成 33%, 66% 版本数据集，在通过前面对 cluster 数实验的方法得到运行时间。

% percent of data	33%	66%	100%
time / ms	0.024	0.042	0.080

从结果来看，可以发现 `kmeans` 收敛所需时间与数据量的大小并不成正比关系, 主要原因是数据量的增加不仅仅影响每次迭代所需的时间，还进而导致迭代次数的一定增加。