开源软件供应链点亮计划—暑期2020

# 开源软件供应链点亮计划
# ——暑期 2020

# Milvus 中的新聚类算法实现

申请人：　　黄俊鹏

项目编号：　　2018087

完成时间：　　2020 年　6 月　10 日

## 一、项目简介

今年，中科院软件与华为 openEuler 项目共同举办了“开源软件供应链点亮计划——2020 年期”项目。该项目与 Google 的 Summer of Code 形式类似：开源社区提供项目需求并提供导师（mentor），在不论是 GSoC 还是这次的“点亮计划”，都是一种非常好的开源实践范式，能够有效地增进高校学生的校园学生利用暑期时间进行开发，主办方为顺利完成的项目提供一定额度的奖金。对开源的理解，增加与真实社区的接触，并积累宝贵的经验。

Milvus 作为社区加入这一计划，提供一些项目需求。欢迎对代码有热情，对大数据项目有高度兴趣的同学一起参加，也欢迎小伙伴们在终止日期（5.29）前提出更多需求。Milvus 社区愿意和大家一起，为国内的开源生态添砖加瓦。

## 二、Milvus 中的新聚类算法实现要求

项目描述：Milvus 是一款开源的特征向量搜索引擎。聚类是一种常见的数据分类方法，在索引中有着重要的应用。Milvus 目前使用 k-means 来实现聚类。请为它添加更多的聚类算法并添加相应的选择开关。

项目难度：中

项目社区导师：李盛俊

导师联系方式： shengjun.li@zilliz.com

合作导师联系方式（选填）：许笑海 xiaohai.xu@zilliz.com

项目产出要求：

- 使用 C++ 在 k-means 基础上实现 k-means++，bisecting k-means
- 和原先 k-means 相比，测试并整理 k-means++，bisecting k-means 在性能和效果方面的差异
- 实现 k-means 算法的扩展 k-mode 算法，并应用在 Binary 数据上
- 增加参数，让用户可以选择聚类算法

项目技术要求：

- 熟悉 Linux 开发环境
- 能使用 Git 进行协作开发

- 熟悉 C++
- 热爱算法与数据结构

相关的开源软件仓库列表：

https://github.com/milvus-io/milvus

## 三、k-means 聚类算法调研

### （一）经典 k-means 算法

k-means 算法是聚类分析中使用最广泛的算法之一。它把 n 个对象根据它们的属性分为 k 个聚类以便使得所获得的聚类满足：同一聚类中的对象相似度较高；而不同聚类中的对象相似度较小。

K-means 算法流程如下：

| 经典 K-means 算法 |
| --- |
| Step 1：从数据集中随机选取$K$个样本作为初始聚类中心$C = \{c_1, c_2, \ldots, c_k\}$； |
| Step 2：针对数据集中每个样本$x_i$，计算它到$K$个聚类中心的距离并将其分到距离最小的聚类中心所对应的类中； |
| Step 3：针对每个类别$c_i$，重新计算它的聚类中心$c_i = \frac{1}{\lvert c_i \rvert}\sum_{x \in c_i} x$ (即属于该类的所有样本的质心)； |
| Step 4：重复第 2 步和第 3 步直到聚类中心的位置不再变化； |

Figure 1 经典 k-means 算法

### （二）k-means 算法缺点

k-means 算法虽然简单快速，但是存在下面的缺点：

1. 聚类中心的个数 K 需要事先给定，但在实际中 K 值的选定是非常困难的，很多时候我们并不知道给定的数据集应该分成多少个类别才最合适。
2. k-means 算法需要随机地确定初始聚类中心，不同的初始聚类中心可能导致完全不同的聚类结果，比如很有可能算法收敛到局部最优解。
3. k-means 算法不适用于非球形簇的聚类，而且不同尺寸和密度的类型的簇，也不太适合。

针对第一个缺点虽然目前有个可行的方法，叫做 Elbow Method：通过绘制 K-means 代价函数与聚类数目 K 的关系图，选取直线拐点处的 K 值作为最佳的

聚类中心数目。但是因为这个方法在实际的运用中出现拐点的情况很少，比较提倡的做法是从实际问题出发，人工指定比较合理的 K 值，通过多次随机初始化聚类中心选取比较满意的结果。

针对第二个问题我们可以通过 k-means++算法来解决第二个缺陷。

### （三）k-means++算法

2007 年由 D.Arthur 等人[1]提出的 K-means++针对图 1 中的第一步做了改进。可以直观地将这改进理解成这 K 个初始聚类中心相互之间应该分得越开越好。整个算法的描述如下图所示：

| K-means++算法 |
| --- |
| Step 1: 从数据集中随机选取一个样本作为初始聚类中心$c_1$； |
| Step 2: 首先计算每个样本与当前已有聚类中心之间的最短距离(即与最近的一个聚类中心的距离)，用$D(x)$表示；接着计算每个样本被选为下一个聚类中心的概率 $\frac{D(x)^2}{\sum_{x\in X} D(x)^2}$。最后，按照轮盘法选择出下一个聚类中心； |
| Step 3: 重复第 2 步直到选择出共$K$个聚类中心； |
| 之后的过程与经典 K-means 算法中第 2 步至第 4 步相同 |

Figure 2 k-means++算法

虽然 k-means++算法可以确定地初始化聚类中心，但是从可扩展性来看，它存在一个缺点，那就是它内在的有序性特性：下一个中心点的选择依赖于已经选择的中心点。

k-means++算法的简单例子说明算法是如何选取初始聚类中心的。

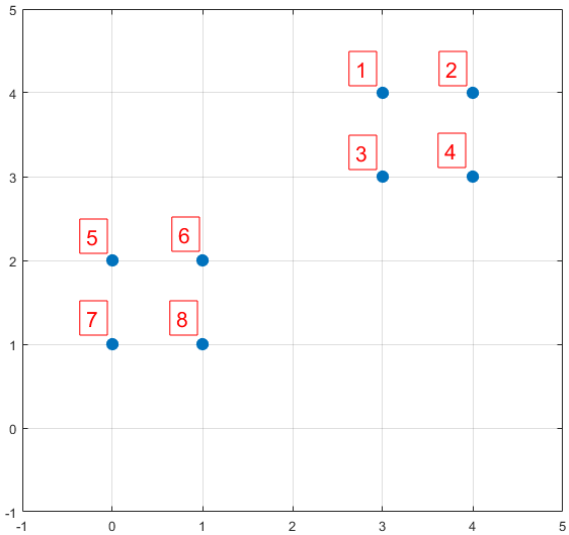



Figure 3 样本图

假设经过图 2 的步骤一后 6 号点被选择为第一个初始聚类中心，那在进行步骤二时每个样本的 D(x)和被选择为第二个聚类中心的概率如下表所示：

| 序号 | ① | ② | ③ | ④ | ⑤ | ⑥ | ⑦ | ⑧ |
|---|---|---|---|---|---|---|---|---|
| $D(x)$ | $2\sqrt{2}$ | $\sqrt{13}$ | $\sqrt{5}$ | $\sqrt{10}$ | 1 | 0 | $\sqrt{2}$ | 1 |
| $D(x)^2$ | 8 | 13 | 5 | 10 | 1 | 0 | 2 | 1 |
| $P(x)$ | 0.2 | 0.325 | 0.125 | 0.25 | 0.025 | 0 | 0.05 | 0.025 |
| $Sum$ | 0.2 | 0.525 | 0.65 | 0.9 | 0.925 | 0.925 | 0.975 | 1 |

Figure 4 算法计算结果图

其中的 P(x)就是每个样本被选为下一个聚类中心的概率。最后一行的 *Sum* 是概率 P(x)的累加和，用于轮盘法选择出第二个聚类中心。方法是随机产生出一个 0~1 之间的随机数，判断它属于哪个区间，那么该区间对应的序号就是被选择出来的第二个聚类中心了。例如 1 号点的区间为[0,0.2)，2 号点的区间为[0.2, 0.525)。

从上表可以直观的看到第二个初始聚类中心是 1 号，2 号，3 号，4 号中的一个的概率为 0.9。而这 4 个点正好是离第一个初始聚类中心 6 号点较远的四个点。这也验证了 K-means 的改进思想：即离当前已有聚类中心较远的点有更大的概率被选为下一个聚类中心。可以看到，该例的 K 值取 2 是比较合适的。当 K 值大于 2 时，每个样本会有多个距离，需要取最小的那个距离作为 D(x)。

### （三）bisecting k-means 算法

针对前面所说的第一个缺点，由于传统的 k-means 算法的聚类结果易受到初始聚类中心点选择的影响，即是属于登山式算法，算法容易陷入局部最优解，它不能保证最终结果是最优的，因为我们一开始选择的中心点是随机的。因此最终的聚类结果和起始点的选择有很大关系，所以在传统的 k-means 算法的基础上进行算法改进，对初始中心点选取比较严格，各中心点的距离较远，这就避免了初始聚类中心会选到一个类上，一定程度上克服了算法陷入局部最优状态。

bisecting k-means 算法采用一种用于度量聚类效果的指标是是 SSE（Sum of Squared Error，误差平方和）。SSE 值越小表示数据点越接近他们的质心，聚类效果也最好。因为对误差取了平方，因此更加重视远离中心的点。一种肯

定可以降低 SSE 的方法是增加族的个数，但这违背了聚类的目标。聚类的目标是在保持簇数目不变的情况下提高簇的质量。

$$\mathrm{SSE}=\sum_{i=1}^{k}\sum_{x\in c_i}\mathrm{dist}(c_i,\mathrm{x})^2$$

bisecting k-means 算法的基本思想是：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进行划分，选择哪一个簇进行划分取决于对其划分时候可以最大程度降低 SSE（平方和误差）的值。上述基于 SSE 的划分过程不断重复，直到得到用户指定的簇数目为止。

二分 K-Means 聚类算法伪代码：

1. 将所有点看成一个簇
2. 当簇数目小于 k 时，对于每一个簇
   a) 计算总误差
   b) 在给定的簇上面进行 KMeans 聚类（k=2）
   c) 计算将该簇一分为二之后的总误差
3. 选择使得误差最小的那个簇进行划分操作

## （四）k-mode 算法

k-means 算法是一种简单且实用的聚类算法，但是传统的 k-means 算法只适用于连续属性的数据集，而对于离散属性的数据集，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 作为 k-means 的一种扩展，适用于离散属性的数据集，计算距离的方式采用的是汉明距离。汉明距离通过比较两个向量每一位是否相同，若不同则汉明距离加 1，这样得到汉明距离，向量相似度越高，得到的汉明距离越小。

k-modes 算法伪代码：

假设有 N 个样本，M 个属性且全是离散的，簇的个数为 k

1. 随机确定 k 个聚类中心 C1,C2...CkC1,C2...Ck，CiCi 是长度为 M 的向量，Ci=[C1i,C2i,...,CMi]Ci=[Ci1,Ci2,...,CiM]
2. 对于样本 xj(j=1,2,...,N)xj(j=1,2,...,N)，分别比较其与 k 个中心之间的距离（这里的距离为不同属性值的个数，假如 x1=[1,2,1,3],C1=[1,2,3,4]x1=[1,2,1,3],C1=[1,2,3,4]，那么 x1x1 与 C1C1 之间的距离为 2）
3. 将 xjxj 划分到距离最小的簇，在全部的样本都被划分完毕之后，重新确定簇中心，向量 CiCi 中的每一个分量都更新为簇 ii 中的众数
4. 重复步骤二和三，直到总距离（各个簇中样本与各自簇中心距离之和）不再降低，返回最后的聚类结果。

## （五）k-means 算法小结

K-Means 的主要优点有：

- 原理比较简单，实现也是很容易，收敛速度快（在大规模数据集上收敛较慢，可尝试使用 Mini Batch K-Means 算法）。
- 聚类效果较优。
- 算法的可解释度比较强。
- 主要需要调参的参数仅仅是簇数 k。

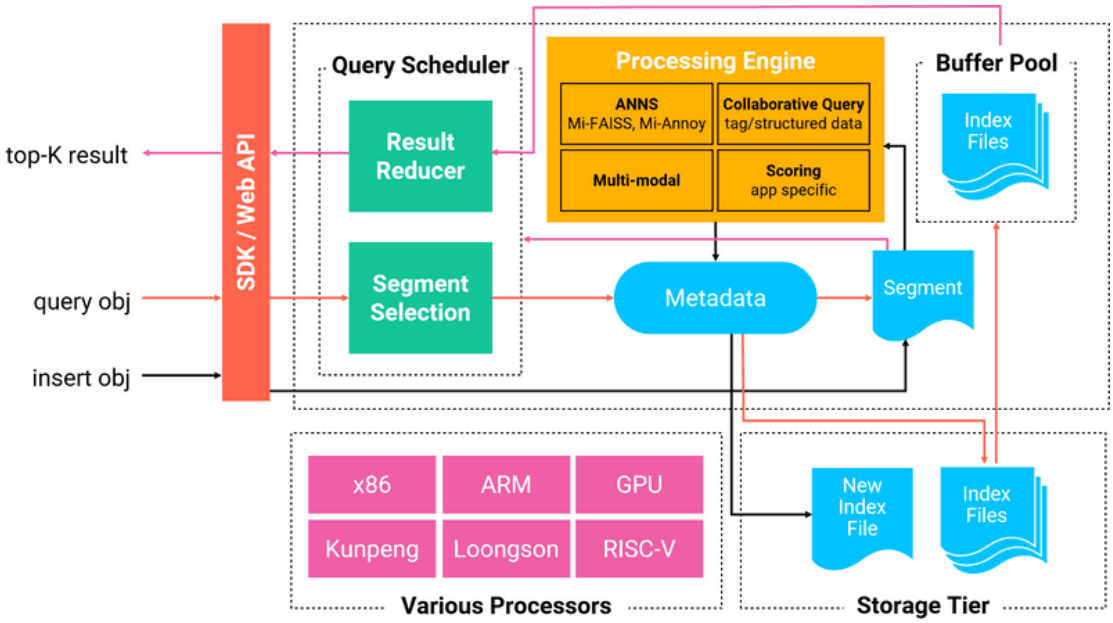
K-Means 的主要缺点有：

- K 值的选取不好把握（实际中 K 值的选定是非常难以估计的，很多时候，事先并不知道给定的数据集应该分成多少个类别才最合适。下篇博客专门讲这个）
- 对于不是凸的数据集比较难收敛（于密度的聚类算法更加适合，比如 DBSCAN 算法，后面再介绍 DBSCAN 算法）
- 如果各隐含类别的数据不平衡，比如各隐含类别的数据量严重失衡，或者各隐含类别的方差不同，则聚类效果不佳。
- 采用迭代方法，得到的结果只是局部最优。（可以尝试采用二分 K-Means 算法）
- 对噪音和异常点比较敏感。（改进 1：离群点检测的 LOF 算法，通过去除离群点后再聚类，可以减少离群点和孤立点对于聚类效果的影响；改进 2：改成求点的中位数，这种聚类方式即 K-Mediods 聚类）
- 对初始值敏感（初始聚类中心的选择，可以尝试采用二分 K-Means 算法或 K-Means++算法）
- 时间复杂度高 O(nkt)，其中 n 是对象总数，k 是簇数，t 是迭代次数。
- 只适用于数值型数据，只能发现球型类簇。

## （六）其他聚类算法

假如时间足够，可以实现 Mean-Shift 聚类、DBSCAN、EM、凝聚层次聚类等等

## 四、Milvus 设计架构




本项目主要改进 Milvus 的向量检索算法，Milvus 面向向量的相似性检索的方法分为精确检索和近似检索两类，本项目关注的是近似检索中的向量量化的编码算法。

基于向量量化的方法通常采用聚类的方式对向量集合中的向量进行划分。该方法通过 k-means 等聚类方法将向量集合划分为多个聚类，并记录各个聚类的中心点的坐标。在向量搜索时，首先依次比对目标向量与各个聚类中心的距离，选择出与目标向量最为接近的若干个聚类中心。接下来获取这些聚类中心所对应聚类中的所有向量，依次计算各向量与目标向量的距离，选择出距离最为接近的若干个向量。

该方法采用聚类的方法将数据集合划分，从而在搜索过程中排除掉与目标向量相似度较低的向量。然而，该方法在高维向量的搜索中容易遗漏部分潜在的与目标向量距离较近的向量，从而难以达到较高的准确度。

## 五、项目计划方案

| | |
|---|---|
| 7 月 1 号-7 月 7 号 | 熟悉 Milvus，阅读 Milvus 中 k-means 源码 |
| 7 月 7 号-7 月 14 号 | 在 k-means 的基础上，用 C++实现 k-means++，bisecting k-means |
| 7 月 14 号-7 月 28 号 | 测试修改算法，测试并整理 k-means++，bisecting k-means 在性能和效果方面的差异，编写测试报告 |
| 7 月 28 号-8 月 4 号 | 实现 k-means 算法的扩展 k-mode 算法，并应用在 Binary 数据上，需要查看二进数数据相关方法 |
| 8 月 4 号-8 月 11 号 | 增加参数，让用户可以选择聚类算法。完善项目代码，测试相关实验结果等等 |

| | |
|---|---|
| 8 月 11 号-8 月 15 号 | 准备中期报告 |
| 8 月 15 号- | 待定 |

# 参考文献


[1] k-means++: The Advantages of Careful Seeding, David Arthur and Sergei Vassilvitskii, 2007

[2] Bisecting k-means 聚类算法实现, http://shiyanjun.cn/archives/1388.html

[3] K-means 聚类算法的三种改进(K-means++, ISODATA, Kernel K-means)介绍与对比, https://www.cnblogs.com/yixuan-xu/p/6272208.html

[4] k-means 、 k-means++ 以 及 k-means|| 算 法 分 析 , https://cloud.tencent.com/developer/article/1455886

[5] 聚类算法之 K-Means 及其变种, https://www.biaodianfu.com/k-means.html