一种基于标准差的 K-medoids 聚类算法

邓玉芳,张继福
(太原科技大学 计算机科学与技术学院,山西 太原 030024)

摘 要:K-medoids 聚类分析具有对孤立点敏感度较低和良好的鲁棒性等特点,但由于初始聚类中心的选取和中心点迭代更新等,聚类精度和效率较低。文中根据标准差体现数据离散程度,定义了初始中心点候选集,给出了一种基于标准差的 K-medoids 聚类算法。该算法首先利用标准差定义了初始中心点候选集,并采用逐步增加的方式确定初始中心点,从而保证了选取密集程度较大的样本点作初始聚类中心点,同时避免选取到密集程度较低的样本点尤其是孤立点作为初始中心点;其次,按照数据样本归属于最近的中心点的原则,形成初始聚类簇,不断更新聚类中心点,直到聚类误差平方和相同为止,形成聚类簇;最后,在 UCI 数据集和人工数据集上的实验验证了该聚类算法具有良好的聚类精度、效率和鲁棒性。

关键词:K-medoids 聚类算法;初始中心点;标准差;UCI 数据集
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2020)08-0053-08
doi:10.3969/j.issn.1673-629X.2020.08.009

A K-medoids Clustering Algorithm Based on Standard Deviation

DENG Yu-fang, ZHANG Ji-fu
(School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China)

Abstract: The K-medoids clustering algorithm has the advantages of low sensitivity to isolated points and strong robustness. However, due to the selection of initial clustering center and the iterative updating of the center point, the clustering accuracy and efficiency are low. The initial center point candidate set is defined according to the standard deviation, and a K-medoids clustering algorithm based on standard deviation is presented. Firstly, the initial center point candidate set is defined by the standard deviation, and the initial center point is determined by a stepwise increasing, which ensures the selection of dense sample points as the initial center point, and avoid the selection of dense sample points, especially isolated points, as the initial center point. Secondly, according to the principle that the data sample belongs to the nearest central point, the initial clusters is formed, and the cluster center points are continuously updated until the clustering error squares is the same to form clusters. In the end, the experiment on UCI dataset and artificial dataset validates that the proposed algorithm has better clustering accuracy, efficiency and robustness.

Key words: K-medoids clustering algorithm; initial center point; standard deviation; UCI dataset

0 引 言

聚类分析是数据挖掘[1]、模式识别[2]等领域的重要研究内容之一,在识别数据的内在结构方面具有重要的作用[3],并已广泛地应用在金融分析、疾病诊断、假新闻检测[4]、农业灾害预测等实际问题中。聚类分析是一种常用的无监督数据挖掘方法[5],可将数据集划分成若干个簇,目标是使在同一个簇里的数据相似度尽可能得高,不同的簇之间的数据相似度尽可能得低,由此根据数据信息将数据划分为若干簇,揭示数据的原始分布。K-medoids 算法[6]是一类基于划分的聚类分析方法,具有对孤立点敏感度低和良好的鲁棒性等优点,并已得到了广泛应用。

目前,大多数 K-medoids 聚类算法,由于初始聚类中心点的选取和中心点迭代更新等原因,存在着聚类精度和效率较低,且需要额外设置参数等不足。文中利用标准差选择候选初始聚类中心,给出了一种 K-medoids 聚类分析算法。该算法首先利用标准差定义了初始中心点候选集度量公式,有效地避免密集程度较低的样本点,尤其是孤立点作为初始聚类中心;其次采用从两个初始中心点逐步增加中心点直到 K 个中心点的方式,从初始中心点候选集中确定初始中心点,避免初始中心点选择在同一个聚类簇;然后按照将数据样本归属于最近的中心点的原则,形成初始聚类簇;再次更新聚类中心点,直到与上一次的聚类误差平方和

收稿日期:2019-09-20 **修回日期**:2020-01-20
基金项目:国家自然科学基金(61876122)
作者简介:邓玉芳(1994-),女,硕士研究生,研究方向为数据挖掘与并行计算;张继福,博士,教授,博导,研究方向为数据挖掘与并行计算。

相同,形成聚类簇;最后采用 UCI 数据集和人工数据集进行实验,验证了该聚类算法的有效性。

1 相关工作

聚类分析是将数据集划分成若干个簇,在簇里的数据对象的相似度尽可能高,不同簇之间的数据对象的相似度尽可能低。常用的聚类分析算法大致分为:基于划分的方法、基于密度的方法、基于层次的方法、基于网格的方法、基于模型的方法[7],以及基于子空间的方法[8-10]。K-medoids 算法由于是以簇中心的实际样本对象作为簇中心点,从而有效地降低了对于噪声数据和孤立点的敏感性。K-medoids 聚类算法在选择初始中心点时,有可能选为离散点或异常点,容易使聚类过程陷入局部极值,同时也需要不断地迭代更新聚类中心点,因此聚类效果和效率较差。

K-medoids 聚类算法经典的 PAM 算法思想是:选取数据集中实际样本对象来代表类簇中心,首先随机选择 K 个初始中心点,将剩余的所有非中心点样本分配到与其最为相似的聚类簇中,计算聚类代价函数。其次选取一个非中心点样本作为新的中心点替换原来的中心点,然后计算替换中心点后的聚类代价函数,如果聚类代价函数减少,则由新的中心点替换原来的中心点,形成新的 K 个中心点,按此方式不断地进行中心点的迭代更新,直到聚类代价函数不再降低,没有可以替换的中心点为止。目前 K-medoids 聚类分析的研究成果主要集中在如下三方面。

(1)初始聚类中心点选取。

Park 等人提出了快速 K-medoids 算法[11],按照密度排序,采用前 K 个样本点作为初始中心点。在更新中心点时采用了 K-means,相比 PAM 算法在计算量上有所降低,在效率上有所提高。但是因为采用的选取初始中心点的方式会导致选取的初始中心点可能在一个聚类簇,因此不能很好地选择出分布在不同簇的初始中心点,使得聚类效果的准确性不高。马箐等人提出了基于粒计算的 K-medoids 聚类算法,采用了粒度概念[12],该聚类算法给出了新的样本相似度函数,并定义了类簇中心,利用等价关系产生粒子,然后依据粒子包含样本的数据量大小来定义粒子密度,最后选择密度较大的前 K 个粒子的中心样本点作为 K-medoids 聚类算法的初始聚类中心,改进 K-medoids 聚类算法的初始中心随机选取对聚类结果的影响,从而提高 K-medoids 聚类算法的效率。谢娟英等人[13]提出了密度峰值优化初始中心的 K-medoids 聚类算法,该聚类算法首先需要做出决策图,然后根据决策图来确定初始中心点,但是同样也会出现初始中心点选择在同一个簇的情况。Yu 等人[14]提出了 INCK 聚类算法,该聚类算法从部分符合要求的数据样本中确定聚类中心,提高了聚类结果的准确性,但是在找寻符合要求的样本时存在参数的选取问题,而参数的选取会影响聚类结果的准确性。为此,文中定义了新的初始中心点候选集,在原始数据上进行聚类。

(2)迭代更新聚类中心点。

Chu 等人[15]推导了一个新的不等式,该不等式可用于最近邻搜索问题。提出了基于新不等式,先前的中心点指标,内存利用,三角形不等式准则和部分距离搜索的四种基于 K-medoids 算法的搜索策略。颜宏文等人[16]提出了基于宽度优先搜索的 K-medoids 聚类算法。该算法利用粒计算初始化获取 K 个有效粒子,从粒子中选出 K 个中心点作为初始中心点。之后分别对 K 个粒子中的对象建立以中心点为根节点的相似对象二叉树,通过宽度优先搜索遍历二叉树迭代出最优中心点。宋红海等人[17]提出了基于优化粒计算下微粒子动态搜索的 K-medoids 聚类算法。该算法是在优化的粒计算前提下,提出了基于微粒子动态搜索策略,以初始中心点作为基点,形成一个微粒子,在微粒子内部,采用离中心点先近后远的原则进行搜索,有效地缩小搜索范围,提高了聚类准确率。余冬华等人[18]提出的 SPAM 算法中在总结的三角不等式的基础上提出了 2 个加速定理,其中一个定理适用于一次交换一个中心点的情况,另一个加速定理是第一个定理的扩展,可适用于一次交换多个中心点的情况。同时 SPAM 算法存储样本到其聚类中心的距离和中心点之间的距离,以提高效率。

(3)聚类分析的并行化。

在处理海量数据信息时面临的内存容量和 CPU 处理速度的问题上通常采用并行化来处理。Jiang 等人[19]实现了基于 Hadoop 分布式计算平台上的 K-medoids 聚类。每个提交的作业都有许多迭代的 MapReduce 程序,在 Map 阶段每个样本被分到距离中心最相似的那个簇。在 comebine 阶段计算每个簇的中心点,在 reduce 阶段计算新的中心点。当新中心点和原来的中心点相同时停止迭代。Zhao 等人[20]通过引入 Canopy 算法和 Max-Min 距离算法改进了原始的 K-Medoids 算法,并选择了 K 个点作为聚类的初始中心。然后使用 MapReduce 计算框架来并行化算法,改进的聚类算法不仅具有良好的加速性能,而且提高了聚类的准确性和收敛性,在处理大规模数据方面具有很大的性能优势。赖向阳等人提出了一种 MapReduce 架构下基于遗传算法的 K-Medoids 聚类[21]。利用遗传算法的种群进化特点来改进 K-Medoids 算法的初始中心敏感的问题,然后将遗传 K-Medoids 算法再结合 MapReduce 并行,提高算法效率。王永贵等人提出

了一种基于 Hadoop 的高效 K-Medoids 并行算法[22]。该算法通过改进初始中心点选择和中心点替换策略这两个方面提高聚类精度。利用 Hadoop 计算平台结合基于 Top K 的并行随机抽样策略,实现了高效稳定的 K-Medoids 并行算法,之后又通过调整 Hadoop 平台,实现了算法的进一步优化。

综上所述,近些年来很多研究者都对聚类分析做了一定的研究。K-medoids 算法作为一种基于划分的聚类分析方法,以实际样本点作为簇中心点,从而有效地降低了对于噪声数据和孤立点的敏感性,具有良好的鲁棒性。但是在选择初始中心点时,有可能选为离散点或异常点,使得聚类准确性不高。迭代更新中心点需要大量距离计算,使得聚类效率较低。

2　基本概念

聚类分析任务是将给定数据集划分成多个簇,使簇中的数据对象尽可能相似,不同的簇之间的数据对象差异性大,可采用欧氏距离来衡量数据对象之间的相似性[23]。假设数据集 $X=\{x_1,x_2,\cdots,x_n\}$,样本数为 n ,每个样本的维数是 p ,第 i 个样本的第 a 个属性值表示为 x_{ia} 。参照文献[13]两个样本间的欧氏距离,给出样本 x_i 与 x_j 之间的欧氏距离,公式如下:

$$d(x_i,x_j)=\sqrt{\sum_{a=1}^{p}(x_{ia}-x_{ja})^2} \tag{1}$$

其中, $d(x_i,x_j)$ 表示样本 x_i 与 x_j 的距离, $i=1,2,\cdots,n$, $j=1,2,\cdots,n$ 。

参照文献[14],聚类误差平方和、数据集的标准差和每个样本的标准差公式分别定义如下:

$$E=\sum_{i=1}^{k}\sum_{x\in c_i}d(o_i,x)^2 \tag{2}$$

其中, o_i 表示第 i 个簇的簇中心点, c_i 表示第 i 个簇,而 x 是属于第 i 个簇的样本点。数据集的标准差定义如下:

$$v=\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}d(x_i,\bar{x})^2} \tag{3}$$

其中, $\bar{x}$ 是所有数据样本的均值, v 表示数据集标准差的值。

每个样本的标准差公式如下:

$$v_i=\sqrt{\frac{1}{n-1}\sum_{j=1}^{n}d(x_i,x_j)^2} \tag{4}$$

其中, v_i 就是每个样本的标准差值。

3　基于标准差的 K-medoids 聚类分析

3.1　初始中心点候选集

K-medoids 聚类算法中初始中心点的选取影响最终的聚类结果。初始中心点选择在接近最终聚类中心点区域时,聚类结果的准确性相对较高且迭代更新中心点的次数较少。当初始中心点选择为严重偏离最终聚类中心区域的样本或者孤立点时,聚类过程容易陷入局部极值,聚类结果准确性较低且迭代更新中心点的次数较多。在 K-medoids 聚类算法中,初始中心点选择已成为提高聚类分析效果和效率的关键因素。

标准差在概率统计中最常使用统计分布程度上的测量,标准差定义为方差的算术平方根,反映数据的离散程度。标准差越小,反映数据分布比较密集,标准差越大,反映数据分布比较离散。聚类中心点的密集程度是较高的,所以中心点样本的标准差相对是较小的。相反的孤立点样本的密集度是较低的,孤立点样本的标准差相对是较大的。对于上一章节给定的数据集 X ,依据式(3)和式(4),将每个样本 x_i 的标准差 v_i 与整体数据集的标准差 v 进行比较,当 v_i 小于 v 时,表明 x_i 在分布密集程度相对较高的区域,因而成为聚类中心点的可能性要大;当 v_i 大于 v ,表明 x_i 在分布密集程度相对较低的区域,因而成为初始中心点的可能性要低。但是不能排除当 v_i 略大于 v 时,样本 x_i 是中心点的可能性,所以以大于 v 的所有样本的平均标准差作为初始中心点候选集的上界。从而既可以使密集程度较大的样本点在初始中心点候选集内,又可以使离散程度不太大的样本点也在初始中心点候选集里。

为了避免孤立点或者密集度较低的样本点被选为初始中心点,同时也为了使初始中心点被选为密集度较大的样本点,定义了初始中心点候选集,以进一步提高聚类的效果和效率。当样本的标准差小于超出数据集标准差的所有样本的平均标准差时,该样本点就有可能是初始中心点,初始中心点候选集 s_m 的定义如下:

$$s_m=\{x_i \mid v_i\leqslant v',i=1,2,\cdots,n\} \tag{5}$$

其中, v' 是 v_i 大于 v 的所有 v_i 的均值。

在 K-medoids 聚类分析中,不必从全部数据对象中选择初始中心点,仅从初始中心点候选集中选择即可,从而有效地提高了聚类中心点选取效率和效果。

3.2　聚类分析

在 K-medoids 聚类算法中,初始中心点选择尤为重要。在初始中心点的选取上,为了避免选取到孤立点作为初始中心点,同时又为了选取到密集程度较大的样本点作为初始中心点,文中利用式(5)定义的初始中心点候选集,从初始中心点候选集选取初始中心点,并迭代更新,其聚类过程参考 INCK 聚类算法由如下两步来实现。

首先在初始中心点候选集中选取两个初始中心点,并迭代更新两个初始中心点。在选择第一个初始中心点 o_1 时,选取距离到所有样本点距离之和最小的

样本点作为中心点,公式如下:

$$o_1 = \underset{x_i \in s_m}{\operatorname{argmin}} \{ d_i \mid i = 1,2,\cdots,n \} \tag{6}$$

其中,样本 x_i 到所有样本点的距离之和 d_i 的公式如下:

$$d_i = \sum_{j=1}^{n} d(x_i, x_j) \tag{7}$$

第二个初始中心点 o_2 的选取为初始中心点候选集中距离第一个初始中心点最远的样本点,使初始中心点尽可能地选择在不同的聚类簇中,避免出现在同一聚类簇里,公式如下:

$$o_2 = \underset{x_i \in s_m}{\operatorname{argmax}} \{ d(x_i, o_1) \mid i = 1,2,\cdots,n \} \tag{8}$$

然后按照就近原则聚类,把所有的样本点归属于距离最近的中心点,计算聚类误差平方和,之后更新每个簇中心,使每个簇内新中心点距离其簇中所有样本的距离之和最小,公式如下:

$$o_j^{\sim} = \underset{x_i \in c_j}{\operatorname{argmin}} \{ \sum_{x_l \in c_j} d(x_i, x_l) \mid i = 1,2,\cdots,n \} \tag{9}$$

其中,c_j 表示第 j 个聚类簇,x_l 要和 x_i 一样是属于 c_j,如果 x_l 不属于 c_j,则不将其与 x_i 的距离算在内。用新中心点代替原中心点,然后聚类计算聚类误差平方和,若与上一次聚类误差平方和一样则不再更新中心点,否则继续更新中心点。

其次从初始中心点候选集中,选取其余的 $k-2$ 个初始中心点。假设已经得到 g ($2 \leqslant g < k$)个中心点,在选择第 $g+1$ 个初始中心点时,在每个聚类簇中选出距离聚类簇中心点最远的样本点,同时该样本点要属于初始中心点候选集,公式如下:

$$o_i' = \underset{x_l \in c_i \cap s_m}{\operatorname{argmax}} \{ d(x_l, o_i) \mid l = 1,2,\cdots,n \} \tag{10}$$

g 个簇中选出的点集如 $o' = \{o_1', o_2', \cdots, o_g'\}$ 所示。从 o' 中选出距离中心点最远的样本点作为第 $g+1$ 个中心点,确保第 $g+1$ 个中心点最有可能是增加的簇的中心点,公式如下:

$$o_{g+1} = \underset{o_j \in o}{\operatorname{argmax}} \{ d(o_j', o_j) \mid j = 1,2,\cdots,g \} \tag{11}$$

然后聚类迭代更新中心点。按此方式逐步增加初始中心点直到确定 k 个中心点,并得到最终聚类结果。

3.3 聚类分析算法

根据上一小节聚类分析的基本思想,基于标准差的 K-medoids 聚类分析算法伪代码描述如下所示。

算法 1:SDK 聚类算法(standard-deviation-based K-medoids clustering algorithm)

输入:数据集 X,簇数 k

输出:k 个聚类簇

(1) s_m = getsm( )

(2) for i = 1 to n do

(3) for j = 1 to n do

(4) 根据式(7)得到 d_i

(5) end for

(6) end for

(7) for i = 1 to s do

(8) 根据式(6)得到第一个初始中心点 o_1, s 为初始中心点候选集大小

(9) end for

(10) for i = 1 to s do

(11) 根据式(8)得到第二个初始中心点 o_2

(12) end for

(13) Cluster( )

(14) for i = 1 to n do

(15) 按照式(2)计算聚类误差平方和 E

(16) end for

(17) Update( );

(18) if k > 2 then

(19) for g = 2 to k - 1 do

(20) for j = 1 to s do

(21) 根据式(10)和式(11)得到第 g + 1 个初始中心点

(22) end for

(23) Cluster( );

(24) for h = 1 to n do

(25) 按照式(2)计算聚类误差平方和 E

(26) end for

(27) Update( );

(28) end for

(29) end if

算法 2:getsm( )

输入:数据集 X

输出:初始中心点候选集 s_m

(1) for i = 1 to n do

(2) for j = i to n do

(3) 计算得到样本间距离 $d(x_i, x_j)$

(4) end for

(5) end for

(6) for i = 1 to p do

(7) for j = 1 to n do

(8) 计算得到均值 $\bar{x}$

(9) end for

(10) end for

(11) for i = 1 to n do

(12) 根据式(3)得到数据集标准差 v

(13) end for

(14) for i = 1 to n do

(15) for j = 1 to n do

(16) 根据式(4)得到每个样本点的标准差值 v_i

(17) end for

(18) end for

(19) for i = 1 to n do

(20) 根据式(5)得到 s_m;

(21) end for

算法 3:Update( )

输入:g 个中心点

输出:更新后的 g 个中心点和 g 个聚类簇
(1)While true do
(2)　　for i =1 to n do
(3)　　for j =1 to n do
(4)按照式(9)找到更新后的中心点
(5)　　end for
(6)　　end for
(7)　　Cluster( );
(8)　　for h =1 to n do
(9)按照式(2)计算聚类误差平方和 newE
(10)　　end for
(11)　　if newE= =E then
(12)　　break;
(13)　　else
(14)　　E=newE
(15)　　end if
(16)end while

算法 4:Cluster( )
输入:g 个中心点
输出:g 个聚类簇
(1) for i =1 to n do
(2) $d = d(x_i, o_1)$
(3) $\text{setlabel}_i = 1$
(4)　　for j =2 to k do
(5)　　newd= $d(x_i, o_j)$
(6)　　if newd<d then
(7) $\text{setlabel}_i = j$
(8) d =newd
(9)　　end if
(10)　　end for
(11) end for

3.4　时间复杂度分析

在 SDK 聚类算法中,由算法 2 计算样本间距离的时间复杂度是 $o(n^2)$,计算均值的时间复杂度是 $o(np)$,计算数据集的标准差的时间复杂度是 $o(n)$,计算所有样本的标准差的时间复杂度是 $o(n^2)$,计算初始中心点候选集的时间复杂度是 $o(n)$,在算法 1 中计算 d_i 的时间复杂度是 $o(n^2)$,选取初始中心点的时间复杂度是 $o(ks)$。在算法 4 中,样本聚类的时间复杂度为 $o(nk)$,因此整体的聚类时间复杂度是 $o(nk^2)$,在算法 3 中,假设更新中心点的最大迭代次数是 t 次,则更新中心点的时间复杂度是 $o(tn^2)$,所以整体的更新中心点的时间复杂度为 $o(tkn^2)$,因此 SDK 聚类算法的整体的时间复杂度是 $o(n^2 + np + n + n^2 + n + n^2 + ks + nk^2 + tkn^2)$,最终时间复杂度表示为 $o(n^2 + tkn^2 + nk^2)$。

4　实验结果及相关分析

实验环境:Intel(R) Core(TM) i5-8265U CPU, 8 G内存,windows10 操作系统,eclipse 作为开发平台,采用 java 语言实现 SDK 聚类算法。文中为验证 SDK 聚类算法的准确性以及鲁棒性,选用 SPAM 聚类算法,INCK 聚类算法和经典的聚类算法 k-means[24] 在 UCI 数据集和人工数据集上进行实验验证。在对准确性的验证上采用了 Rand 指数还有 F-measure 值这两个评价指标。在验证 SDK 聚类算法的聚类效率时,与同样是 K-medoids 聚类算法的 SPAM 聚类算法、INCK 聚类算法进行实验对比。在实验中 SDK 聚类算法、SPAM 聚类算法以及经典 k-means 聚类算法的参数只需要 k 值,INCK 聚类算法除了需要设定参数 k 值外,还需要一个额外的参数 λ。

4.1　数据集

文中在实验中采用了 UCI 数据集,用来分析比较不同聚类算法的准确性和效率。在实验中采用的所有数据集的名称,数据的样本数,样本属性个数和真实的簇数如表 1 所示,其中 sensor 表示的是 sensor_readings_24 数据集。同时为了实验分析聚类算法的鲁棒性,采用了 Matlab 工具,随机生成了 5 组符合正态分布的人工数据集。在每组数据集分为 4 类,每类有 500 个数据样本,属性维度是 10 维的基础上,分别增加 15%、20%、25%、30%、35% 的噪声数据,从而形成 5 组人工数据集。表 2 为生成人工数据集的各种具体参数。按照表 2 给出的参数,随机生成了 5 组人工数据集,生成的 5 组人工数据集的具体的样本数,属性个数以及真实的簇数如表 3 所示。

表 1　UCI 数据集

数据集	样本数	属性个数	真实簇数
banknote	1 372	5	2
bupa	345	7	2
haberman	306	3	2
Iris	150	4	3
wavaform21	5 000	21	3
waveform40	5 000	40	3
sensor	5 456	24	4
wdbc	569	30	2

表 2　生成人工数据集的参数

类序号	均值	标准差	噪声数据标准差
第一类	5	1.5	
第二类	2	1.3	
第三类	7	1.1	
第四类	9	0.5	2

表 3　人工数据集

数据	样本数	属性个数	真实簇数
15%	2 300	10	4
20%	2 400	10	4

续表 3

数据	样本数	属性个数	真实簇数
25%	2 500	10	4
30%	2 600	10	4
35%	2 700	10	4

4.2 初始中心点候选集

在 SDK 聚类算法中,由于采用了初始中心点候选集,所以聚类初始中心点选择不必从全部的数据样本中去确定,并且更为准确且有效。在实验验证中各个数据集的初始中心点候选集的样本个数如表 4 所示。

表 4 初始中心点候选集

数据集	候选集样本数
banknote	868
bupa	275
haberman	185
Iris	75
waveform21	2 874
waveform40	2 830
sensor	2 943
wdbc	405

由表 4 可以看出,在实验中采用的所有数据集的初始中心点候选集样本数相较于原来整体数据集都减少了,从而在确定初始中心点时,不必从全体数据中寻找,减少了计算量,有效地提高了聚类效率。

4.3 聚类精度

为了验证 SDK 聚类算法的聚类效果,实验采用 SDK 聚类算法与 SPAM 聚类算法和 INCK 聚类算法以及 k-means 聚类算法在 UCI 数据集上进行了聚类精度的实验对比,其中实验结果数据采用了各算法运行 10 次的平均值。

从图 1 和图 2 的结果可以看出,无论是 Rand 指数还是 F-measure 值,表现最好的是 SDK 聚类算法,因此整体上,聚类的准确性最高的是 SDK 聚类算法。SDK 聚类算法的聚类准确性相对较好主要是因为在对初始中心点的选取上,避免了选取孤立点为初始中心点,同时又尽可能地避免选取的初始中心点在同一个簇的情况,并且使初始中心点选在密集度相对较高的样本上,因而聚类准确性上表现较好。

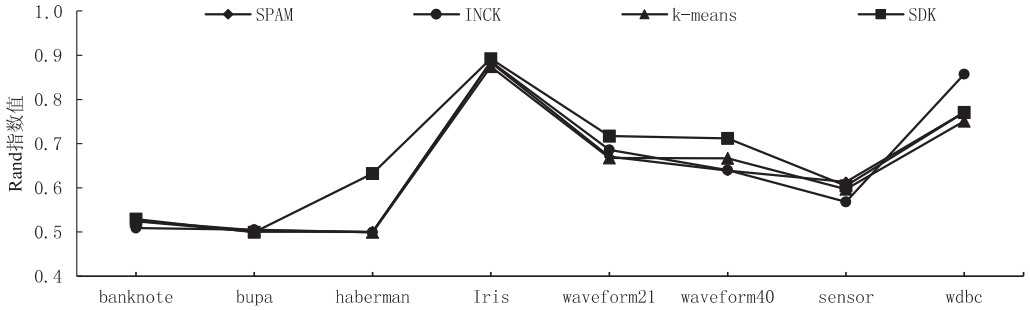

图 1 真实数据集上 Rand 指数的值

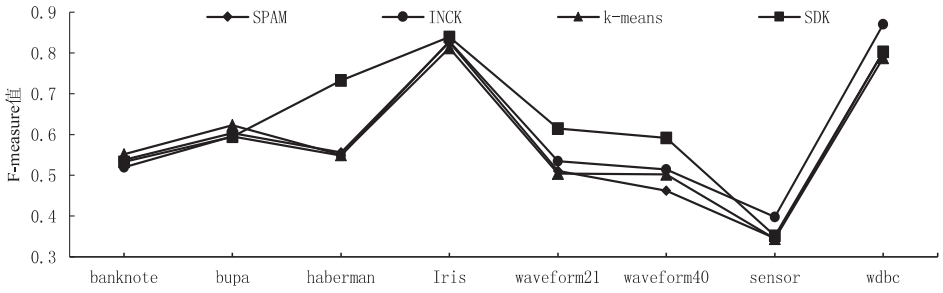

图 2 真实数据集上 F-measure 的值

4.4 聚类效率

采用表 1 所示的 UCI 数据集,与同样是 K-medoids 算法的 SPAM 聚类算法和 INCK 聚类算法进行实验对比,验证 SDK 聚类算法的效率。实验结果如表 5 所示,其中取 10 次运行结果的平均值作为实验结果。

由表 5 可知,SDK 聚类算法效率是最高的,而 SPAM 聚类算法效率是最差的,INCK 聚类算法的效率居中。其主要原因是 SDK 聚类算法采用了初始中心点候选集的方式,确定中心点的时候不必从全部样本中选择,在更新中心点时和 INCK 聚类算法一样采用了快速 K-medoids 的方式,而 SPAM 聚类算法采用的是 PAM 聚类算法的中心点更新方式,从全部数据样本中查找中心点,所以 SDK 聚类算法在计算量上相对减少。除此之外,SDK 聚类算法和 INCK 聚类算法采用了存储样本之间距离的方式,之后用到直接调用即可。SPAM 聚类算法虽然也采用存储距离的方式,但是只存储样本点到其中心点的距离和中心点之间的距

离,在之后用到其他两个样本之间的距离时都要重新计算。因此 SDK 聚类算法和 INCK 聚类算法都减少了不必要的距离的重复计算,而 SPAM 聚类算法需要重复计算距离。所以整体上 SDK 聚类算法和 INCK 聚类算法的效率都要比 SPAM 聚类算法要高,而文中 SDK 聚类算法的效率是最高的。

表 5　聚类算法运行时间　　s

数据集	SPAM	INCK	SDK
Banknote	2.422 4	0.800 9	0.774 6
Bupa	0.664 87	0.342 6	0.316 5
Haberman	0.293 9	0.216 7	0.195 8
Iris	0.231 7	0.090 4	0.079 8
wavaform21	335.155 6	13.398 8	12.486 8
waveform40	709.034 2	24.553 9	22.577 8
sensor	1 055.596 5	17.599 3	16.727 4
wdbc	2.615 6	1.066 4	0.991 7

4.5　聚类鲁棒性

为了验证 SDK 聚类算法的鲁棒性,采用表 3 所示的人工数据集,将 SDK 聚类算法,SPAM 聚类算法,INCK 聚类算法和 k-means 聚类算法进行了实验对比分析,其中实验结果选取了各算法运行 10 次结果的平均值。表 6 是 Rand 指数的实验结果,表 7 是 F-measure 的实验结果。

表 6　人工数据集上的 Rand 指数

数据	SPAM	INCK	k-means	SDK
15%	0.933 20	0.8366 2	0.856 78	0.933 20
20%	0.922 06	0. 828 99	0.924 72	0.920 30
25%	0.909 46	0.825 22	0.857 03	0.909 75
30%	0.897 74	0.889 78	0.864 9	0.897 74
35%	0.890 06	0.818 77	0.863 11	0.890 06

表 7　人工数据集上的 F-measure

数据	SPAM	INCK	k-means	SDK
15%	0.857 10	0.701 32	0.725 08	0.857 10
20%	0.833 25	0.690 86	0.839 44	0.830 55
25%	0.808 48	0.684 03	0.723 38	0.811 11
30%	0.787 18	0.774 35	0.727 36	0.787 18
35%	0.777 26	0.678 13	0.736 61	0.777 26

从表 6 和表 7 可知,在 4 个聚类算法中,SDK 聚类算法和 SPAM 聚类算法的鲁棒性表现相近且表现较好,INCK 聚类算法的鲁棒性是最差的。SDK 聚类算法的鲁棒性保持良好的主要原因是 SDK 聚类算法采用了初始中心点候选集,在选取中心点的时候,尽量避免不合适的数据被选为中心点。

5　结束语

利用了标准差反映数据分布离散程度的原理,定义了初始中心点候选集,从初始中心点候选集中选取初始中心点,避免孤立点或者密集度较低的样本点被选为初始中心点,同时也使初始中心点选为密集度较大的样本点。在 UCI 数据集及人工数据集上进行了实验验证,其实验结果验证了该算法的有效性。下一步的工作主要是对 SDK 聚类算法的并行化。

参考文献:

[1] 甘月松,陈秀宏,陈晓晖. 一种 AP 算法的改进:M-AP 聚类算法[J]. 计算机科学,2015,42(1):232-235.

[2] 王卫卫,李小平,冯象初,等. 稀疏子空间聚类综述[J]. 自动化学报,2015,41(8):1373-1384.

[3] 张顺龙,库　涛,周　浩. 针对多聚类中心大数据集的加速 K-means 聚类算法[J]. 计算机应用研究,2016,33(2):413-416.

[4] ZHANG Chaowei, GUPTA A, KAUTEN C, et al. Detecting fake news for reducing misinformation risks using analytics approaches[J]. European Journal of Operational Research, 2019, 279(3): 1036-1052.

[5] 杨辉华,王 克,李灵巧,等. 基于自适应布谷鸟搜索算法的 K-means 聚类算法及其应用[J]. 计算机应用,2016,36(8):2066-2070.

[6] 宋飞豹,贾瑞玉. 精英遗传 K-medoids 聚类算法[J]. 计算机工程与应用,2018,54(22):144-149.

[7] 金建国. 聚类方法综述[J]. 计算机科学,2014,41(11A): 288-293.

[8] PANG Ning, ZHANG Jifu, ZHANG Chaowei, et al. Parallel hierarchical subspace clustering of categorical data[J]. IEEE Transactions on Computers, 2019, 68(4): 542-555.

[9] PANG Ning, ZHANG Jifu, ZHANG Chaowei, et al. PUMA: parallel subspace clustering of categorical data using multi-attribute weights[J]. Expert Systems with Applications, 2019, 126: 233-245.

[10] 庞 宁,张继福,秦 啸. 一种基于多属性权重的分类数据子空间聚类算法[J]. 自动化学报,2018,44(3):517-532.

[11] 谢娟英,鲁肖肖,屈亚楠,等. 粒计算优化初始聚类中心的 K-medoids 聚类算法[J]. 计算机科学与探索,2015,9(5): 611-620.

[12] 马 箐,谢娟英. 基于粒计算的 K-medoids 聚类算法[J]. 计算机应用,2012,32(7):1973-1977.

[13] 谢娟英,屈亚楠. 密度峰值优化初始中心的 K-medoids 聚类算法[J]. 计算机科学与探索,2016,10(2):230-247.

[14] YU Donghua, LIU Guojun, GUO Maozu, et al. An improved K-medoids algorithm based on step increasing and optimizing medoids[J]. Expert Systems With Applications, 2018, 92: 464-473.

[15] CHIANG C, CHU Shuchuan, RODDICK H F, et al. New search strategies and new derived inequality for efficient K-Medoids-based algorithms[J]. Chinese Journal of Electronics, 2007, 16(1): 82-87.

[16] 颜宏文,周雅梅,潘 楚. 基于宽度优先搜索的 K-medoids 聚类算法[J]. 计算机应用,2015,35(5):1302-1305.

[17] 宋红海,颜宏文. 基于优化粒计算下微粒子动态搜索的 K-medoids 聚类算法[J]. 智能计算机与应用,2016,6(2):9-13.

[18] 余冬华,郭茂祖,刘 扬,等. 基于距离不等式的 K-medoids 聚类算法[J]. 软件学报,2017,28(12):3115-3128.

[19] JIANG Yaobin, ZHANG Jiongmin. Parallel K-Medoids clustering algorithm based on Hadoop[C]//2014 IEEE 5th international conference on software engineering and service science. Beijing: IEEE, 2014: 649-652.

[20] ZHAO Yonghan, CHEN Bin, LI Mengyu. Parallel K-Medoids improved algorithm based on MapReduce[C]//2018 sixth international conference on advanced cloud and big data (CBD). Lanzhou: IEEE, 2018: 18-23.

[21] 赖向阳,宫秀军,韩来明. 一种 MapReduce 架构下基于遗传算法的 K-Medoids 聚类[J]. 计算机科学,2017,44(3):23-26.

[22] 王永贵,戴 伟,武 超. 一种基于 Hadoop 的高效 K-Medoids 并行算法[J]. 计算机工程与应用,2015,51(16):47-54.

[23] 张雪萍,龚康莉,赵广才. 基于 MapReduce 的 K-Medoids 并行算法[J]. 计算机应用,2013,33(4):1023-1025.

[24] CUI Xiaoli, ZHU Pingfei, YANG Xin, et al. Optimized big data K-means clustering using MapReduce[J]. The Journal of Supercomputing, 2014, 70(3): 1249-1259.