第 27 卷第 10 期
2010 年 10 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 27 No. 10
Oct. 2010

文章编号: 1000−8152(2010)10−1435−06

聚类中心初始化的新方法

李春生[1,2], 王耀南[2]

(1. 广东商学院 数学与计算科学学院, 广东 广州 510320; 2. 湖南大学 电气与信息工程学院, 湖南 长沙 410083)

摘要: k–均值聚类算法易受初始聚类中心的影响而陷入局部最优解. 现有聚类中心初始化方法尚未得到广泛认可. 本文依据每个类内至少有一个数据稠密区, 且处于不同类的数据稠密区比处于同一类的数据稠密区相距更远的假设, 在数据集合上构造一棵最小支撑树, 应用根树原理在其上搜索数据稠密区并估计其密度, 从中选出密度大且足够分离的数据稠密区, 以其内的点作为初始聚类中心, 得到了一个聚类中心初始化的新方法. 将此方法与现有的方法进行比较, 仿真实验表明, 本文方法性能更优越.

关键词: 最小支撑树; 聚类中心初始化; k-means算法

中图分类号: TP39　　**文献标识码:** A

New initialization method for cluster center

LI Chun-sheng[1,2], WANG Yao-nan[2]

(1. Department of Mathematics and Computational Science, Guangdong University of Business Studies, Guangzhou Guangdong 510320, China;
2. College of Electrical and Informational Engineering, Hunan University, Changsha Hunan 410083, China)

Abstract: The k-means clustering algorithm is prone to be trapped into local optima by inappropriate initial cluster centers. For this reason, the existing initialization methods for the cluster center have not been widely accepted. We assume that there is at least one dense subset of data in a cluster; and the dense subsets between different clusters are more distant than those in the same cluster. A minimum spanning tree is built for the given data set. The dense subsets can be found through the search from root trees, and their densities are obtained by the estimation technique for data density. The initial cluster centers are picked out from the dense subsets that are dense enough and distant enough from each other. The comparisons between the proposed method and current methods show that the performance of the proposed method is promising.

Key words: cluster center initialization; minimum spanning tree; k-means algorithm

1 引言(Introduction)

数据聚类是机器学习、数据挖掘和模式识别的基础. k-means聚类算法因其简单易行而广为应用[1]. 其原理是: 应用迭代方法将数据聚为k类, 使得每个数据到其最邻近的类中心的距离平方之和达到最小[2]. k-means聚类算法是一种局部优化策略, 容易陷入局部最优解, 而且对初始中心非常敏感[3]. 好的初始中心有助于k-means算法避免陷入局部最优解. 因此, 如何为k-means算法提供初始聚类中心吸引了国内外学者的广泛关注. 自1965年Forgy首次为k-means算法提出了一个聚类中心初始化方法以来[4], 国内外学者提出了各种不同的聚类中心初始化方法. 早期的有McQueen提出的随机筛选法[2]、Tou和Gonzales提出的simple cluster seeking method(简易中心搜索法)[5]、Linde et al.提出的Binary Splitting (二分法)[6]. 其共同的不足在于时间复杂度高, 不适合大规模的数据. 近期的有Huang and Harris提出的direct search binary splitting(直接搜索二分法)[7]、Katsavounidis et al.提出的KKZ法[8]. 最近, Khan and Ahmad提出了一个基于数据压缩原理的聚类中心初始化方法, 简记为CCIA[9]. 这种方法内置了k-means算法, 其时间复杂度随着数据维数的增大而增加, 不适合高维数据[9]. Stephen J. Redmond提出了一个基于*kd*-tree的聚类中心初始化方法, 简记为*kd*-tree[10]. 这种方法利用*kd*-tree, 从待划分的数据集中筛选密度大而又相互分离的数据作为初始中心. 其基本原理相对合理, 但其估算数据密度的方法有待改进. 其他聚类中心初始化方法见文献[11], 各种聚类中心初始化方法的比较研究可参看文献[2,12].

收稿日期: 2008−09−02; 收修改稿日期: 2010−01−03.
基金项目: 国家“863”计划重点资助项目(2007AA04Z224); 国家自然科学基金重点资助项目(60835004).

聚类中心初始化方法各式各样、各有特色, 但没有一个获得广泛的认可[13]. 本文应用图论中的最小支撑树原理, 为k-means算法设计一个聚类中心初始化方法, 简记为MST. 该方法的基本思想是, 从稠密而又足够分离的k个数据区域中各选出一个数据作为k-means的初始中心, 即选择密度大而又足够分离的k个点作为k-means的初始中心. 将本文提出的方法与Khan and Ahmad[9]和Stephen J. Redmon[10]的方法进行比较. 实验显示, 本文提出的方法具有明显的优越性. 由于本文提出的方法是对Stephen J. Redmon[10]方法的改进, 下面简要介绍它.

2 基于*kd*-trees的聚类中心初始化方法(A method for initialising the K-means clustering algorithm using kd-trees)

记$X=\{x_1,x_2,\cdots,x_n\}\in\mathbb{R}^m$为一个待划分的数据集, 其中$x_j=(x_{1j},x_{2j},\cdots,x_{mj})^{\mathrm{T}}$为$m$维向量. 在数据集$X$上构造*kd*-tree的方法如下: 首先视$X$为一个母数据箱(box or bucket), 计算X的每一维属性的长度l, 其中l为该维属性值的最大值与最小值之差. 找出X的长度最大的属性, 称为最长维, 以最长维的属性值的均值或中位数为分界点, 将X的最长维的属性值一分为二, 最长维中处于同一组的属性值所在的数据模式构成一个子数据箱, 由此得两个子数据箱X_1,X_2. 例如, 若X的最长维为第k维, 记第k维的属性值为$A_k=\{x_{k1},x_{k2},\cdots,x_{kn}\}$, 计算$A_k$的均值或中位数$a$, 以$a$为分界点, 将$A_k$一分为二, 得两个子集合$A_1$, A_2, 其中:

$$A_1=\{x_{kj}^{(1)}|x_{kj}^{(1)}\in A_k,x_{kj}^{(1)}\leqslant a\},$$
$$A_2=\{x_{kt}^{(2)}|x_{kt}^{(2)}\in A_k,x_{kt}^{(2)}>a\}.$$

则A_1中的所有元素$x_{kj}^{(1)}$所在的数据模式x_j构成一个子数据箱X_1, A_2中的所有元素$x_{kt}^{(2)}$所在的数据模式x_t构成另外一个子数据箱X_2. 然后, 分别视X_1与X_2为母数据箱, 按照上述方法一分为二, 如此反复, 直至所有子数据箱中的数据个数少于预设值α而成为叶子数据箱. Stephen J. Redmon设定$\alpha=20$[10].

设X的*kd*-tree共有q个叶子数据箱, 记为$L_1,L_2,\cdots,L_q$, $A_i^{(t)}$为L_t的第i维属性值构成的集合, $\lambda_{\max}^{(i)}$和$\lambda_{\min}^{(i)}$分别为$A_i^{(t)}$的最大值与最小值, 定义$l_i=\lambda_{\max}^{(i)}-\lambda_{\min}^{(i)}$为$L_t$的第$i$维属性的长度, 则$L_t$的体积为

$$V_t=\prod_{i=1}^{m}l_i. \tag{1}$$

若L_t含有N_t个数据模式, 则其密度为

$$\rho_t=N_t/V_t. \tag{2}$$

用L_t中数据的均值y_t代表$L_t(t=1,2,\cdots,q)$, 则ρ_t表示y_t代表的数据箱L_t的密度, 也称为y_t的密度. 基于*kd*-tree的聚类中心初始化方法就是从$y_1,y_2,\cdots,y_q$中选出k个密度大而又相互分离的点作为k-means算法的初始中心. 具体筛选过程为: 首先将$y_1,y_2,\cdots,y_q$的密度$\rho_1,\rho_2,\cdots,\rho_q$按照由小到大的顺序排列, 并用每个密度的序号替换它, 由此将密度ρ_t转换成序号密度$\hat{\rho}_t$. 从$y_1,y_2,\cdots,y_q$中选出一个序号密度最大的点作为第一个初始中心v_1; 然后计算余下的每个候选点y_i与已经入选的所有初始中心的最小距离$d_j=\min\limits_{c}\{d(y_j,v_c)\}$, 其中$d(y_j,v_c)$为欧氏距离. 再从中选出$\hat{\rho}_jd_j$达到最大的候选点作为初始中心, 重复此过程, 直至选出k个初始中心.

Stephen J. Redmond指出, 在实际应用中有两个问题值得关注: 其一, 叶子数据箱的体积可能为0, 例如, 若叶子数据箱L_t中所有数据的第j维属性值相等, 则第j维属性的长度为0, 导致$V_t=0$, 此时ρ_t无意义. 在这种情形下, 他建议用长度不为0的其他属性的长度的几何平均值作为长度为0的属性的长度. 其二, ρ_jd_j的值可能被ρ_j或d_j所左右, 若被ρ_j左右, 则选出的初始中心密度大但分离度不够, 反之, 选出的初始中心足够分离但密度不够大, 有可能选出噪音点作为初始中心. 为避免上述两种不利情形, Stephen J. Redmond建议将ρ_j按照由小到大的顺序排列, 用ρ_j对应的序号取代ρ_i的真实值. 为避免噪音的干扰, 他同时提出, 去掉20%的低密度数据子箱, 重新筛选一组初始中心.

3 基于最小支撑树的聚类中心初始化算法(A cluster center initialization method based on minimum spanning tree)

Stephen J.Redmond指出的上述问题确实存在, 但其建议的处理方法值得商榷. 比如, 用密度的序号替换密度的真实值能否削弱密度的真实值对ρ_jd_j的影响, 要视具体情况而定. 在不同叶子数据箱的密度差异不是很大的情况下, 用密度的序号替换密度的真实值反而会强化密度在选择初始中心时的作用. 例如, 假设总共有100个叶子数据箱, 其密度的真实值差异不大, 不妨设最大密度与最小密度满足$\rho_{\max}=40\rho_{\min}$, 但是$\hat{\rho}_{\max}=100\hat{\rho}_{\min}$, 此时, 叶子数据箱的密度普遍被人为放大, 依据最大化$\hat{\rho}_jd_j$的原则选择初始中心更加强化了密度的作用, 更容易选择密度大而分离度不够的叶子数据箱的中心点为初始中心. 在数据聚类时, 由于数据的相关信息, 如数据的分布、稠密度、噪音等等, 难以获得, Stephen J. Redmond建议的方法只有在满足其假设条件的时候才有效.

仔细分析可以发现, 上述两个问题的根源在于叶子数据箱的体积的定义. 若将体积定义为数据各维长度的平方和的算术平方根, 即$V_i = \sqrt{\sum_{j=1}^{m} l_j^2}$, 则体积总是大于0, 上述第1个问题就不会出现. 上述第2个问题是d_j与V_j的定义导致它们的数量级不一致. d_j是m个数的平方和的算术平方根, V_j是m个数的乘积. m个数的乘积的数量级与这m个数的平方和的算术平方根的数量级有时候相差较大. 例如, 3个数的乘积$0.1\times0.3\times0.5 = 0.015$远小于这3个数的平方和的算术平方根$\sqrt{0.1^2+0.3^2+0.5^2} = 0.59$, 而3个数的乘积$10\times30\times50 = 15000$远大于这3个数的平方和的算术平方根$\sqrt{10^2+30^2+50^2} = 59.16$. 加上各个数据子箱内含的数据量几乎相等, 导致$\rho_j d_j = N_j d_j/V_j$ 的值可能被d_j左右, 选出足够分离但密度小的噪音点为初始中心, 也可能被V_j左右, 选出密度大但处于同一类的点为初始中心. 例如, 三维数据模式x与已入选的某个初始中心处于同一类, 不妨设$d_x = \sqrt{1^2+2^2+3^2} = \sqrt{14}$, 按照Stephen J. Redmond定义的体积估计数据密度, 不妨设$V_x = 1\times0.6\times0.5 = 0.3$, 其数量级大概是$d_x$的十分之一, 则$\rho_x = n_x/0.3$, $\rho_x d_x = n_x\sqrt{14}/0.3$; 另外一个三维数据模式$y$与已入选的任意初始中心都不同类, 其与已入选的初始中心的距离大于d_x, 不妨设$d_y = \sqrt{3^2+4^2+5^2} = 5\sqrt{2}$, $V_y = 1\times2\times0.5 = 1$, 则$\rho_y = n_y$, 由于$n_x \approx n_y$, 所以, $\rho_y < \rho_x$表明x所在的数据区域比y所在的数据区域更稠密. $\rho_y d_y = n_y 5\sqrt{2}/1 < n_x\sqrt{14}/0.3 = \rho_x d_x$表明$x$比$y$更有可能被选为初始中心, 导致选出的初始中心处于同一类. 若将体积定义为数据各维长度的平方和的算术平方根, 则

$$\begin{aligned}
&V_x = \sqrt{1^2+0.6^2+0.5^2} = \sqrt{1.31},\\
&\rho_x = n_x/\sqrt{1.31},\\
&V_y = \sqrt{1^2+2^2+0.5^2} = \sqrt{5.25},\\
&\rho_y = n_y/\sqrt{5.25},\\
&\rho_x d_x = n_x\sqrt{14}/\sqrt{1.31} <\\
&\qquad 5\sqrt{2}n_y/\sqrt{5.25} = \rho_y d_y,
\end{aligned}$$

此时, y比x更有可能被选为初始中心, 导致选出的初始中心处于不同类. 此例表明, 在相同的情况下, 用Stephen J.Redmond定义的体积估算数据密度, 由于距离d_j与体积V_j的数量级相差较大, 导致$\rho_j d_j$的值可能被ρ_j左右, 选出的初始中心可能处于同一类, 若将体积定义为数据各维长度的平方和的算术平方根, 由此估计数据密度, 由于距离d_j与体积V_j的数量级相近, $\rho_j d_j$的值不再被ρ_j左右, 而是由ρ_j与d_j共同确定, 导致选出的初始中心处于不同类.

基于*kd*-tree的聚类中心初始化方法的思想基础值得借鉴, 即选择密度大且足够分离的叶子数据箱内的点的均值作为初始中心是可取的. 众所周知, 一个数据集内可能存在多个区域, 不同区域的形状、大小及其密度由数据自身结构决定, 可能相差较大. 用叶子数据箱界定区域的范围能否反映数据的真实结构, 这一点依然值得商榷. 况且, Stephen J.Redmond说*kd*-tree产生的不同叶子数据箱内的数据量几乎相等[10], 用叶子数据箱内数据的数量与其体积之比度量叶子数据箱的密度, 其结果完全取决于叶子数据箱的体积. 一旦体积的定义不合理, *kd*-tree就无法准确地估计数据的密度. 另外, *kd*-tree中的参数α 在应用中难以设定. 直观地讲, 设计一个聚类中心初始化方法应该同时考虑两个原则: 1) 每个类的初始中心应该处于该类中数据比较密集的区域; 2) 不同类的初始中心足够分离. 这两个原则可用数据密度与聚类中心间的距离加以量化, 其中, 数据密度是一个比较难估计的量, 尤其是高维数据. 本文用最小支撑树原理估计数据密度, 提出一个聚类中心初始化的新方法, 称为基于最小支撑树的聚类中心初始化方法. 该方法利用最小支撑树的连通性搜索不同的数据区域, 它们的形状、大小及其内含的数据量可能不同, 且由数据自身结构决定, 这不同于*kd*-tree的叶子数据箱界定的数据区域, 因为不同叶子数据箱内含的数据量几乎相等. 为避免Stephen J.Redmond定义的体积产生的两个问题, 本文重新定义了数据区域的体积. 新定义的体积能确保数据区域的体积总是大于0, 且保持数据区域的体积的数量级与两点间的距离的数量级一致. 用数据区域内的数据量与其体积之比度量数据区域的密度, 用不同数据区域的中心点间的距离度量它们之间的分离度, 依据最大化密度与分离度之积的原则, 选出初始中心. 下面详述本文的聚类中心初始化方法.

以X中任意两点x_i, x_j间的距离$d(x_i, x_j)$为连接这两点的边的权, 创建一个带权完全图$G(X, E, D)$, 其中$E = \{\{x_i, x_j\}\}|x_i, x_j \in X\}$, $D = \{d(x_i, x_j)|\{x_i, x_j\} \in E\}$. 应用Prim算法[16]求出图$G$的一棵最小支撑树$T(X, E_{\rm T}, D_{\rm T})$, 其中$E_{\rm T} \subset E$, $D_{\rm T} \subset D$. 在最小支撑树$T(X, E_{\rm T}, D_{\rm T})$上搜索初始中心, 其过程如下:

首先, 确定初始中心的所有候选点. 计算最小支撑树上各点的度$\deg x$, 平均度$\theta = \sum_{x\in X} \deg\ x/|X|$. 以最小支撑树上度大于平均度的点为初始中心的候选点.

然后计算每个候选点所在区域的密度. 这个过程有点复杂, 先以例说明, 再在算法1中给出伪代

码. 若$\deg\ x > \theta$, 将它存入B_x, 以x为父节点, 在最小支撑树上搜索其所有子节点$y_1, y_2, \cdots, y_p$, 将它们存入B_x, 若$\forall i \in \{1, 2, \cdots, p\}$,都有$\deg\ y_i > 1$, 则将子节点$y_1, y_2, \cdots, y_p$升级为父节点, 在最小支撑树上搜索每个父节点的所有子节点, 并将它们存入B_x, 否则, 终止搜索, 由此产生一棵以x为根的子树, 此树上的所有结点构成一个数据区域, 称之为以x为中心的数据区域, 记为B_x. 记$\lambda_{\max,j}, \lambda_{\min,j}$分别为$B_x$中所有数据的第$j$维属性值的最大值与最小值, 定义第$j$维属性的长度$l_j$为

$$l_j = \lambda_{\max,j} - \lambda_{\min,j},\ j = 1, 2, \cdots, m. \tag{3}$$

定义数据区域B_x的体积$V(B_x)$为

$$V(B_x) = \sqrt{\sum_{j=1}^{m} l_j^2}. \tag{4}$$

定义稠密区域B_x的密度, 也称为候选点x的密度为

$$\rho(B_x) = |B_x|/V(B_x) = |B_x|/\sqrt{\sum_{j=1}^{m}(\lambda_{\max,j} - \lambda_{\min,j})^2}. \tag{5}$$

算法 1 估算数据密度.

Step 1 将x分别存入集合F与B_x;

Step 2

while $\min\limits_{y\in F}\{\deg\ y\} > 1$

for $i = 1$ to $|F|$

在最小支撑树上搜索F中第i个结点的所有邻接点, 存入集合C与B_x;

end

用C更新F, 即$F = C$;

清空集合C;

end

Step 3 用式(5)计算点x的密度.

最后筛选初始中心. 选择密度最大的点为第1个初始中心, 如果密度最大的点有多个, 则从中任选一个, 存入集合 V 中. 从筛选第2个初始中心开始, 先计算每个候选点到集合 V 的距离 $d(x, V) = \min\limits_{v\in V}\{d(x, v)\}$, 再计算 $\rho_x d(x, V)$, 依据最大化 $\rho_x d(x, V)$的原则, 选择下一个初始中心, 重复这个过程, 直至选出所有初始中心. 下面给出基于最小支撑树的聚类中心初始化算法的伪代码, 简记为MST.

算法 2 基于最小支撑树的聚类中心初始化算法.

Step 1 创建一个带权完全图$G(X, E, D)$, $E = \{\{x_i, x_j\}|x_i, x_j \in X\}$, $D = \{d(x_i, x_j)|x_i, x_j \in X\}$.

Step 2 应用Prim[16]算法求出$G(X, E, D)$的一棵最小支撑树$T(X, E_{\rm T}, D_{\rm T})$, $E_{\rm T} \subset E$, $D_{\rm T} \subset D$;

Step 3 确定初始中心的所有候选点. 对于$T(X, E_{\rm T}, D_{\rm T})$上的任意一个结点$x$, 若它满足

$$\deg\ x \geqslant \frac{1}{|X|}\sum_{y\in T(X,E_{\rm T},D_{\rm T})} \deg\ y,$$

则x为初始中心的候选点, 它们构成一个候选点集S;

Step 4 按照算法1给出的方法估算每个候选点的密度;

Step 5 从候选点集S中选出k个初始中心.

选择$x_1 = \arg\max\limits_{x\in S}\{\rho_x\}$为第一初始中心$v_1$, 存入集合$V$;

for $i = 2$ to k

计算$d(x, V) = \min\limits_{v\in V}\{d(x, v)\}$, $\forall x \in S$;

选择$x_i = \arg\max\limits_{x\in S}\{\rho_x d(x, V)\}$为第$i$个初始中心$v_i$, 存入$V$;

end

输出所有选定的初始中心$v_1, v_2, \cdots, v_k$.

注 释 $\deg\ x$表明与x邻接的点的数量, 其值越大, 以x为中心的数据区域是稠密区域的可能性越大, 反之, 以x为中心的数据区域是稠密区域的可能性越小. 所以, 并非每个点都有可能成为稠密区域的中心, 比如, 叶子结点处于稠密区域的边缘的可能性很大. 为简化计算, 不需要从每个点出发, 搜索稠密区域, 只要从度相对较大的点出发, 搜索稠密区域即可. 本文认为, 度大于平均度的点是初始中心的候选点. 以候选点为根节点, 在最小支撑树上搜索一棵子树时, 为避免因最小支撑树的联通性而误将整棵最小支撑树当成一棵子树, 并确保搜索出的子树处于稠密区域, 本文设定子节点升级为父节点的条件, 只有当上一代所有父节点的所有子节点都有其子节点, 它们才能升级为父节点. 以不同候选点为根节点的子树所代表的数据区域, 其密度可能不同, 密度越大表明该数据区域越稠密, 其中心点成为初始中心的可能性越大, 在选择第一个初始中心时, 不需要考虑它与其它初始中心的分离度, 因而, 选择密度最大的区域的中心作为初始中心是恰当的、可行的.

4 实验(Experiments)

在下面的实验部分, 用MST, CCIA和*kd*-tree分别表示本文的、Shehroz S. Khan与Stephen J.Redmond的聚类中心初始化方法. 用它们为k-means算法提供初始中心. 为测试本文提出的聚类中心初始化方法的效果, 选用不同规模、不同结构的真实数据—sonar[15], svmguide2[15], zoo[14], segment[15], pendigit(test)[14]与模拟数据(见图1)对MST进行测试,

并与CCIA与*kd*-tree进行比较. 实验结果展示于图1和表1中. 图1(a)中的数据分为4个类, 有3个噪音点. MST和*kd*-tree准确地估算出数据的密度, 它们提供的4个初始中心分别位于4个类的稠密区中, 而CCIA由于内置了*k*-means算法而易受噪音干扰, 误将一个噪音点当作初始中心. 图1(b)中的数据有5个类, 形状和稠密度互不相同, 结构相对复杂. 只有MST准确地估计出数据的密度, 并在选择初始中心时准确地结合了密度与分离度, 提供的5个初始中心分别位于5个类的稠密区, CCIA提供的5个初始中心虽然相互分离, 但有两个初始中心落在类的范围之外. *kd*-tree虽然正确地估计出数据的密度, 但由于各类的密度相差不是很大, 尤其在椭圆形、长方形与线性3个类中, 它们的密度非常接近, *kd*-tree将真实密度转换为序号密度, 人为放大了这3个类的密度的差异, 反而导致数据密度在选择初始中心时处于支配地位, 选择密度大但分离度不够的数据区域内的点作为初始中心. 图1(b)显示, *kd*-tree提供的5个初始中心虽然位于类的稠密区中, 但相互间不够分离, 两个初始中心出现于同一个类中, 密度相对较大的类却被忽略了.

图中: “○”表示MST提供的初始中心; “*”表示CCIA提供的初始中心; “◇”表示*kd*-tree提供的初始中心.

用*k*-means算法对真实数据进行聚类时, 去除真实数据的类标号, 并分别用MST, CCIA和*kd*-tree为*k*-means算法提供初始中心. 就*k*-means算法的模式识别率而言, 表1说明在sonar, segment与pendigit 3个数据上, MST, CCIA和*kd*-tree没有太大的差别, MST略微优于 CCIA 和 *kd*-tree. 但在 svmguide 2[15]与zoo[14]两个数据上, MST明显优于CCIA和*kd*-tree, 大幅度地提高了*k*-mean算法的模式识别率(*k*-mean算法正确识别的模式数与模式总数之比). 就CPU的运行时间而言, 表1表明*kd*-tree最快、CCIA次之、MST最慢. 这是由于应用Prim算法[16]求最小支撑树, 其时间复杂度相对较高. 但在数据sonar上, CCIA比MST慢, 这说明CCIA随着数据维数的增加, 时间复杂度增大, 正如Shehroz S.Khan所说, CCIA不适合于高维数据.

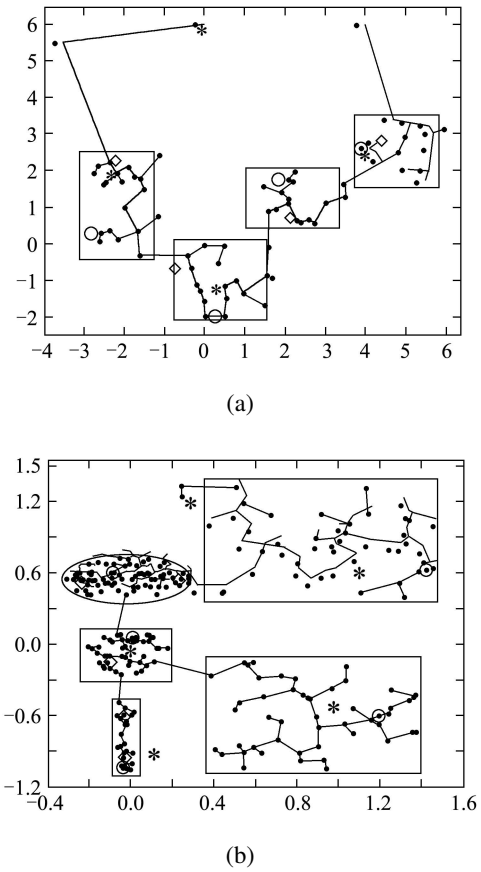

图 1　3种聚类中心初始化方法的效果对比图

Fig. 1　Comparisons among three cluster center initialization methods

表 1　3种初始化方法的比较结果

Table 1　Comparisons among three cluster center initialization methods

数据名称	数据维数	数据个数	聚类数	*k*-means的模式识别率			CPU Time/s		
				MST	CCIA	*kd*-tree	MST	CCIA	*kd*-tree
sonar	60	208	2	56.2500	55.2885	55.2885	0.578783	1.157092	0.033807
svmguide2	20	391	3	66.2404	41.9437	40.6650	2.896330	0.253951	0.038106
zoo	16	101	7	79.2079	71.2871	58.4158	1.258225	0.587573	0.035282
segment	19	2310	7	58.8745	54.8485	53.3333	29.430145	12.105683	0.589930
pendigit	16	3498	10	68.1532	64.4940	66.8382	30.990076	9.639070	0.476125

5 结论(Conclusions)

在Stephen J.Redmond的聚类中心初始化方法的启发下, 本文给出了设计聚类中心初始化方法的两个原则, 在此原则的指导下, 提出了一个基于最小支撑树的聚类中心初始化方法. 该方法克服了Stephen J.Redmond给出的方法在估计数据密度方面的缺陷, 但增加了时间复杂度. 应用真实数据与模拟数据对本文的聚类中心初始化方法MST进行测试, 并与CCIA与kd-tree进行比较. 模拟数据上的实验表明, 在CCIA或kd-tree不能选出处于不同类的数据作为初始中心的时候, MST能选出处于不同类的数据作为初始中心.

真实数据上的实验表明, 应用MST为k-means算法提供初始聚类中心, 相对于应用CCIA和kd-tree提供的初始聚类中心而言, MST在一定程度上提高了k-means算法的模式识别率.

参考文献(References):

[1] 张军峰, 胡寿松. 基于聚类和支持向量机的非线性时间序列故障预报[J]. 控制理论与应用, 2007, 24(1): 64 – 68.
(ZHANG Junfeng, HU Shousong. Nonlinear time series fault prediction based on clustering and support vector machines[J]. *Control Theory & Applications*, 2007, 24(1): 64 – 68.)

[2] MACQUEEN J B. Some methods for classification and analysis of multivariate observation[M] //CAM L, NEYMAN L M J. *Berkeley Symposium on Mathematical Statistics and Probability*. California: University of California Press, 1967: 281 – 297.

[3] PENA J M, LOZANO J A, LARRANAGA P. An empirical comparison of four initialization methods for the k-means algorithm[J]. *Pattern Recognition Letter*, 1999, 20(10): 1027 – 1040.

[4] ANDERBERG M R. *Cluster Analysis for Applications*[M]. New York: Academic Press, 1973: 32 – 38.

[5] TOU J, GONZALES R. *Pattern Recognition Principles*[M]. New Jersey: Addison-Wesley, 1974: 54 – 57.

[6] LINDE Y, BUZO A, GRAY R M. An algorithm for vector quantizer design[J]. *IEEE Transactions on Communication*, 1980, 28(2): 84 – 95.

[7] HUANG C, HARRIS R. A comparison of several codebook generation approaches[J]. *IEEE Transactions on Image Process*, 1993, 2(1): 108 – 112.

[8] KATSAVOUNIDIS I, KUO C C J, ZHEN Z. A new initialization technique for generalized Lloyd iteration[J]. *IEEE Signal Processing Letters*, 1994, 1(10): 144 – 146.

[9] KHAN S S, AHMAD A. Cluster center initialization algorithm for k-means clustering[J]. *Pattern Recognition Letters*, 2004, 25(11): 1293 – 1302.

[10] REDMOND S J, HENEGHAN C. A method for initialising the K-means clustering algorithm using kd-trees[J]. *Pattern Recognition Letters*, 2007, 28(8): 965 – 973.

[11] 巩敦卫, 蒋余庆, 张勇, 等. 基于微粒群优化聚类数目的K-均值算法[J]. 控制理论与应用, 2009, 26(10): 1175 – 1179.
(GONG Dunwei, JIANG Yuqing, ZHANG Yong, et al. k-mean algorithm for optimizing the number of clusters based on particle swarm optimization[J]. *Control Theory & Applications*, 2009, 26(10): 1175 – 1179)

[12] SEINLEY D, BRUSCO M J. Initializing k-means batch clustering: a critical evaluation of several techniques[J]. *Journal of Classification*, 2007, 24 (1): 99 – 121.

[13] MEILA M, HECKERMAN D. An experimental comparison of several clustering methods[R] //*Microsoft Research Report MSR-TR-98-06*. Redmond, WA: [s.n.], 1998: 1 – 28.

[14] *The UCI machine learning repository, 1993*[EB/OL]. http://www.ics.uci.edu/ ∼ mlearn/MLRepository.html.

[15] CHANG C C, LIN C J. *LIBSVM: a library for support vector machines, 2001*[EB/OL]. http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[16] CHARTRAND G, ZHANG P. *Introduction to Graph*[M]. Beijing: Posts & Telecom Press, 2006: 98 – 99.

作者简介:

李春生 (1972—), 男, 博士, 主要从事模式识别、智能计算等领域的研究, E-mail: lcs5812084@sina.com;

王耀南 (1958—), 男, 教授, 博士生导师, 主要从事智能控制、智能信息处理、机器人等领域的研究, E-mail: yaonan@hnu.cn.