一种基于多属性权重的分类数据子空间聚类算法

庞 宁 [1] 张继福 [1] 秦 啸 [2]

摘 要 采用多属性频率权重以及多目标簇集质量聚类准则, 提出一种分类数据子空间聚类算法. 该算法利用粗糙集理论中的等价类, 定义了一种多属性权重计算方法, 有效地提高了属性的聚类区分能力; 在多目标簇集质量函数的基础上, 采用层次凝聚策略, 迭代合并子簇, 有效地度量了各类尺度的聚类簇; 利用区间离散度, 解决了使用阈值删除噪音点所带来的参数问题; 利用属性对簇的依附程度, 确定了聚类簇的属性相关子空间, 提高了聚类簇的可理解性. 最后, 采用人工合成、UCI 和恒星光谱数据集, 实验验证了该聚类算法的可行性和有效性.

关键词 分类数据聚类, 多属性频率, 多目标簇集质量, 属性相关子空间, 区间离散度

引用格式 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. 自动化学报, 2018, **44**(3): 517−532

DOI 10.16383/j.aas.2018.c160726

A Subspace Clustering Algorithm of Categorical Data Using Multiple Attribute Weights

PANG Ning[1] ZHANG Ji-Fu[1] QIN Xiao[2]

Abstract In this paper, we propose a subspace clustering algorithm using frequencies of multiple attributes of categorical data. An attribute-value weight is calculated on the basis of equivalence class in rough set theory by using frequencies of multiple attributes. Attribute-value weights offer ample opportunities to improve classiffication ability of clustering. The well-known parameter problem, which is caused by using a threshold to delete noise points, is solved by the virtue of interval dispersion degrees. By adopting the hierarchical clustering method to iteratively merge sub-clusters, we effectively measure various scale clusters on the basis of a multi-objective clusters quality function. An attribute subspace is determined with the relevance degree of dimension to a cluster, so as to improve the cluster′s interpretability. Finally, we validate the feasibility and effectiveness of our algorithm through extensive experiments using synthetic data as well as UCI and stellar spectral data sets.

Key words Categorical data clustering, multiple attribute frequency, multi-objective cluster quality, attributes related subspace, interval dispersion degree

Citation Pang Ning, Zhang Ji-Fu, Qin Xiao. A subspace clustering algorithm of categorical data using multiple attribute weights. *Acta Automatica Sinica*, 2018, **44**(3): 517−532

聚类是根据数据对象之间的相似程度, 将数据集合理划分成若干数据子集的过程, 每个子集称为一个簇. 聚类分析的目标是使得划分后的数据点在簇内彼此相似, 在簇间彼此相异. 经典的聚类算法主要包括划分聚类算法[1−3]、层次聚类算法[4−6]、基于密度[7] 和网格[8] 的聚类算法以及谱聚类[9] 等, 但没有任何一种聚类算法可以普遍适用于揭示各种多维数据集所呈现出来的多种多样的结构[10]. 随着高维海量数据的广泛应用, 数据类型也日益复杂化和多样化, 以处理数值型数据为主的传统聚类算法已无法满足高维分类数据的聚类需求. 而分类数据广泛分布在各应用领域中, 分类数据的聚类分析已成为目前数据挖掘领域中极具挑战性的研究方向之一[11−16].

目前, 分类数据聚类研究需要解决以下两个问题: 1) 由于高维属性空间的稀疏性, 大量冗余属性或不相关属性使得全属性空间下的聚类结果毫无意义[17], 因此, 属性权重的度量成为提高高维分类数据聚类效果的关键问题之一; 2) 由于分类数据缺乏数值型数据固有的几何特征, 不能进行数值运算, 同时, 也无法使用衡量数值型数据的距离方法去判定分类数据的相似性[18], 因此, 适合分类数据的簇集质量判断标准会直接影响聚类精度. 本文针对分类数据, 定义了一种基于多属性频率的属性权重计算新方法, 并在此基础上, 采用多目标聚类准则, 结合

收稿日期 2016-10-19 录用日期 2017-03-09
Manuscript received October 19, 2016; accepted March 9, 2017
国家自然科学基金 (61572343) 资助
Supported by National Natural Science Foundation of China (61572343)
本文责任编委 曾志刚
Recommended by Associate Editor ZENG Zhi-Gang
1. 太原科技大学计算机科学与技术学院 太原 030024 中国 2. 奥本大学计算机科学与软件工程学院 奥本 36849 美国
1. School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China 2. Department of Computer Science and Software Engineering, Auburn University, Auburn 36849, USA

层次聚类思想, 提出了一种适合分类数据的子空间聚类算法. 该算法无需设定初始参数, 可以识别和处理噪音点, 并根据数据自身的结构特征, 在相关属性子空间上聚类. 本文的主要贡献如下:

1) 定义了一种基于多属性频率的属性权重计算方法;

2) 给出了一种基于多目标簇集质量的聚类准则;

3) 提出了一种基于多属性权重的分类数据子空间聚类算法.

1 相关工作

传统分类数据聚类算法[1, 19−20] 认为各属性维在聚类过程中的作用是无差别的, 全属性空间下的聚类算法虽然可以有效解决低维数据的聚类问题, 但面对高维数据时, 有效性会显著降低. 为了克服传统聚类算法存在的不足, 各种降维技术被用于降低高维属性空间维度, 主要的解决思路包括: 特征转换[21] 和特征选择[22−23]. 文献 [21] 总结对比了几种主要的特征转化技术, 如主成分分析 (Principal components analysis, PCA)、核主成分分析 (Kernel principal component analysis, KPCA) 和独立分量分析 (Independent component analysis, ICA) 在高维空间向低维空间映射过程中的效果差异, 上述技术均采用将高维属性组合生成新属性的思路, 存在新组合属性的解释性差, 数据簇不易理解的缺陷. 文献 [22] 在粗糙集理论的基础上, 提出一种贪婪的混合属性约简算法, 该算法利用所推导出的属性重要性度量规则, 可以有效降低属性空间的维度; 文献 [23] 提出一种基于对偶图特征选择聚类算法, 该算法使用自表示特征保存数据空间和属性空间的几何信息, 在数据空间的自表示系数矩阵上加入稀疏约束以分析属性重要性, 选取重要属性提高聚类质量. 特征选择的方法可有效约简属性空间, 但这种将原属性空间中的若干属性提取出来构成新属性空间的方法, 会造成部分信息丢失. 因此, 针对高维数据, 基于特征转换和特征提取技术的聚类效果仍无法满足实际需求.

近几年, 在传统降维技术研究的基础上, 基于属性子空间的聚类研究受到了广泛关注. 大量文献[24−35] 均验证了在高维数据背景下, 全属性空间上无法形成有意义的簇集, 类簇存在于不同的相关属性子空间中, 属性权重的度量不再以整个属性维为单元, 同一属性在不同数据点上的重要度不同, 因此, 属性子空间算法是目前解决高维数据聚类问题的有效途径之一[24]. 子空间聚类算法不仅可以将相似数据聚合成若干类簇, 同时针对不同类簇会给出各自独立的相关属性子空间. 子空间的思想最初用于解决数值型数据的聚类问题[24−27]. 随着针对分类数据挖掘需求的不断增长, 各种高维分类数据的子空间聚类算法应运而生[28−35], 典型算法有: 文献 [28] 采用最小化目标函数的思想, 提出了一种面向分类数据近似最优划分的子空间算法 (Subspace clustering for categorical data, SUBCAD), 该算法奠定了与聚类相关子空间的理论基础, 但该算法所给出的目标函数无法确保在迭代过程是递减的; 文献 [29] 提出了一种基于图论的分类数据子空间聚类算法 (Mining subspace clustering algorithm, CLICKS), 利用密度参数将聚类问题转化成加权图的思想, 以属性值之间的共现频次作为权值, 反复搜索 k 个最大分离的类, 该算法以属性值的共现次数为基础提高了聚类精度, 但聚类效果依赖于初始密度参数, 同时基于加权图的权重计算增加了时间成本; 文献 [30] 基于属性值在本地属性维上出现的次数, 计算该值在单属性上的频率, 提出一种通过质量判断函数反复迭代判断数据归属的方法 (Projected clustering algorithm for categorical data, PROCAD), 但该方法在判断多类别和属性值域元素个数较少的数据集时, 会出现聚类效果变差的现象; 文献 [31] 提出了一种分阶段的无参层次聚类算法 (Top-down clustering of high-dimensional categorical data, AT-DC), 该算法以属性值在数据簇中的条件概率作为权重, 判断数据点对类簇的影响, 利用全局聚类质量函数, 将簇迭代分裂为子簇, 虽然该算法给出了类簇的划分, 却没有给出与簇集相关的属性子空间, 而且聚类结果与数据的输入顺序相关; 文献 [32] 提出了一种层次分裂聚类算法 (Divisive hierarchical clustering of categorical data, DHCC), 该算法采用多元对应分析思想, 将数据点对应到一维空间中并初步分为两个子簇, 再利用数据点与子簇相似度调整子簇内的数据点, 不断迭代上述分裂过程直至整个簇集总质量无法提高为止, 该算法同样可以避免参数对聚类结果的干扰, 但该算法的时间复杂度取决于多元对应分析的效率, 同时该算法在低维属性空间上的聚类效果会降低; 文献 [33] 在文献 [34] 的基础上, 提出一种双加权 K-Modes 聚类算法 (A mixed attribute weighting k-modes algorithm, MWKM), 利用最小化簇内误差平方和的思想, 自动调整属性在各数据簇中的重要度, 属性权值与该属性在簇内分布的离散程度相关, 该算法计算速度快, 但算法需要设定参数; 文献 [35] 提出一种熵加权聚类算法 (An entropy weighting k-means algorithm, EWKM), 该算法同样采用拉格朗日乘子优化单目标函数, 推导属性权值公式, 将表现属性对于聚类不确定性的熵引入权重计算中, 属性权值与该属性在簇内的熵成反比, 该算法也

需要设定大量初始参数, 同时文献 [33−35] 均是在 K-Modes 算法的基础上提出的, 这类算法在处理类簇尺寸差异较大的数据集时, 往往会出现 “均匀效应” 现象[36], 数据常常会聚合成尺寸相对均匀的簇.

综上所述, 子空间聚类算法思想是基于各属性在聚类过程中权重的差异, 在重要属性组成的相关属性子空间中, 根据聚类目标函数将数据点划分到不同类簇内. 多数子空间聚类算法[30−35] 在选取构成子空间的相关属性时, 仅以属性取值的单属性权重作为衡量标准, 没有考虑到其他属性对于该属性值权重的影响, 即属性取值在整个属性空间上的分布特征. 文献 [29] 在考虑多属性之间的关联对属性权重的影响时, 需遍历整个数据空间以统计属性值的同现次数, 从而增加了聚类过程的时空复杂性. 作为判断聚类质量的关键, 数据点与簇的相似度计算是整个聚类算法的核心, 仅以类内数据点紧凑作为聚类目标会影响到簇集质量[33−35], 聚类结果容易受到非平衡数据分布的影响, 同时, 参数也会干扰聚类算法的稳定性. 因此, 要有效地解决高维分类数据背景下的子空间聚类问题, 需要解决的关键问题为: 利用属性值在多维属性上的分布特征, 获取高权值属性以构建各数据点的相关属性子空间; 在属性子空间的基础上, 将类内紧凑与类间分离相结合作为判断簇集质量的聚类准则, 以解决基于单目标函数聚类算法无法有效处理非平衡数据集的问题.

2　问题描述与求解策略

分类数据 (Categorical data) 是指数据属性值是分类型的数据, 分类属性又称标称属性, 分类属性取值都是有限无序的, 且不可比较大小, 也无法进行数值运算. 参考文献 [30] 与表 1 定义的符号, 问题描述如下:

分类数据的子空间聚类目标是将数据集 U 内的 n 个数据点划分为若干个类簇 C_s 以及噪音集 NOS, 数据点 x_k 由 d 个分类属性值组成, 可形式化表示为 $x_k = \{x_{k1}, x_{k2}, \cdots, x_{kd}\}$, 在属性子空间 SA_s 下, 类簇 C_s 由数据点集 SD_s 构成, C_s 可表示为二元组 (SD_s, SA_s). 例如图 1, U 共有 10 个数据点, 每个数据点由 8 个分类属性值组成, 数据集可划分为三个类簇和噪音集, 分别表示为 C_1, C_2, C_3 和 NOS.

从图 1 可以发现, 各簇对应的相关子空间规模不一定相同, 例如, 簇 C_1 的子空间为 $\{a_1,a_2\}$, 而 C_2 和 C_3 的子空间分别为 $\{a_2,a_3,a_4\}$, $\{a_4,a_5,a_6\}$, 同时各簇之间没有交集 $(SD_i \cap SD_j = \emptyset)$, 即数据点的归属具有唯一性; 属性值分布过于稠密或过于稀疏, 均无法反映类簇的特征, 例如, 属性 a_7 和 a_8 不会出现在任一类簇的属性子空间中; 数据 x_{10} 明显与数据集的其他点不同, 服从不同的分布, 可判定为噪音点.

表 1　符号表示及含义

Table 1　Symbol and notation

符号表示	符号含义
U	数据集
x_k	第 k 个数据点
a_j	第 j 维属性
NOS	由噪音点组成的集合
SA_s	簇 C_s 的相关属性子空间
A	属性集
x_{ki}	数据 x_k 在第 i 维属性上的属性取值
C_s	第 s 个类簇
SD_s	簇 C_s 的数据集合
V_{a_i}	属性 a_i 的值域
S	一个分类信息系统
$[x_k]_{a_i}$	包含数据点 x_k 的 a_i 等价类
n	数据点的总数
d	属性空间的总维数
SC	子簇集

		a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
C_1	x_1	A	C	Q	B	U	Q	Q	A
	x_2	A	C	E	Y	I	W	Q	S
	x_3	A	C	R	H	K	E	Q	D
C_2	x_4	C	S	R	V	L	R	Q	E
	x_5	W	S	R	V	J	S	Q	R
	x_6	E	S	R	V	M	D	Q	T
C_3	x_7	P	S	M	X	T	M	Q	B
	x_8	C	W	Q	X	T	M	Q	N
	x_9	C	T	E	X	T	M	Q	M
NOS	x_{10}	Y	Y	M	C	A	B	Q	I

图 1　聚类示例图

Fig. 1　Clustering sample

针对上述子空间聚类问题, 由第 1 节的相关工作分析可知, 现有分类数据聚类方法在计算属性权重时存在的问题: 1) 尽管单属性权重计算方法有利于提高时间效率, 但仅以单属性评价属性权值会造成属性值聚类区分度下降, 例如图 1, 属性 a_3 不属于 C_1 的属性子空间, 在 C_1 的形成过程中, 尽管具有相同取值 (R), 属性值 x_{33} 的聚类作用应远低于 x_{43} 的作用, 若仅以 R 值在 a_3 上的出现频率来衡量属性值权重, 则无法反映出 x_{33} 与 x_{43} 聚类能力的差异, 因此, 区分同频率属性值的权重需要借助其他相关属性; 在依靠多属性计算属性值权重时, 若以遍历全数据空间为代价, 计算基于多属性的属性值权

重会降低算法的时间效率. 2) 单目标聚类准则虽然可以根据聚类过程中数据点的分布特征自动调整各属性权值, 但仅以数据点与簇中心的距离作为聚类准则, 对于广泛分布的非平衡数据, 边缘数据很容易被误判到其他类中, 影响最终的聚类效果, 同时, 单目标聚类算法需要最优化聚类目标函数, 存在设置参数等问题.

采用多属性频率计算属性权重, 有效地体现了属性值在属性空间中的分布特征, 可解决单属性计算权重所带来的聚类区分度下降等问题, 例如, 从属性空间分布上分析, 由于属性值 x_{43} 与 $x_{42}(RS)$, x_{43} 与 $x_{44}(RV)$ 在数据集上同时出现的频率均为 30 %, 而 x_{33} 与其他属性的同现频率仅为 10 %, 因此, 利用属性 a_2 与 a_4, 基于多属性频率计算属性值 x_{43} 与 x_{33} 的权重, 可以有效区分两属性值的权重差异. 在簇内距离最小化目标的基础上, 结合簇间分离最大化目标作为聚类准则, 有助于边缘数据的正确划分; 单个类中心无法反映大类特征也是造成非平衡数据均匀化的原因之一[37], 因此, 采用层次凝聚聚类思想, 利用子簇代表大类的多个类中心, 通过迭代合并子簇, 将大类的子簇聚集在一起, 从而解决非平衡数据的聚类问题. 子空间聚类问题的求解步骤为: 1) 属性权值的计算. 基于粗糙集利用多属性频率计算各属性值的权重, 该方法不仅考虑到多属性维之间的关联对属性值权重的影响, 同时, 基于粗糙集理论, 利用数据点在不同属性下等价类之间的交集, 统计多属性上属性值的同现次数, 无需遍历整个数据空间, 提高了时间效率; 2) 基于多目标聚类准则的层次凝聚聚类. 利用簇内紧凑和簇间分离的聚类目标, 给出簇集质量函数, 在聚类过程中采用一种自底向上的凝聚聚类策略, 将聚类过程分为初聚类和合并聚类两个阶段, 初聚类阶段利用多目标簇集质量函数先将最相似的数据点生成子簇, 在合并聚类阶段, 迭代合并各子簇以提高整个簇集质量, 形成最终的类簇; 3) 噪音点检测. 在聚类之前, 利用最小化区间离散度的方法, 滤掉那些低权值的点, 即噪音点以提高聚类效果; 4) 属性相关子空间识别. 根据簇内各属性对簇的依附度, 确定各簇的相关属性子空间.

3 基于多属性权重的分类数据子空间聚类

由第 2 节可知, 分类数据聚类步骤主要由四个阶段构成, 即: 分类属性权重计算, 基于多目标聚类准则的层次凝聚聚类, 噪音点识别和识别属性相关子空间.

3.1 分类属性权重

3.1.1 分类属性权重计算依据

解决第 2 节所提出的权重计算问题, 需要考虑两个因素: 1) 属性取值在本地属性上的出现频率, 该值反映了属性值的局部重要程度; 2) 用其他相关属性度量该属性值与相关属性值的共现频率, 该值刻画了属性值的全局重要性. 为了避免计算共现频率而增加的时间成本, 可按照属性值将各属性划分为若干个数据集合, 为了表述方便, 用符号 $S_i(x_k)$ 表示在属性 a_i 上与 x_k 具有相同属性值的数据集合. 统计属性值 x_{ki} 与 x_{kj} 的同现次数, 相当于求解集合 $S_i(x_k)$ 与 $S_j(x_k)$ 交集的基数, 即 $|S_i(x_k) \cap S_j(x_k)|$. 例如图 1, 按照属性值, 属性 a_1 可分为集合: $\{x_1, x_2, x_3\}$, $\{x_4, x_8, x_9\}$, $\{x_5, x_{10}\}$, $\{x_6\}$和 $\{x_7\}$, a_2 可分为 $\{x_1, x_2, x_3\}$, $\{x_4, x_5, x_6, x_7\}$, $\{x_8\}$, $\{x_9\}$和 $\{x_{10}\}$, 求 $S_1(x_3) \cap S_2(x_3)$ 得 $\{x_1, x_2, x_3\}$, 可知属性值 x_{31} 与 x_{32} 同时出现了 3 次.

3.1.2 分类属性权重

在粗糙集理论[38] 中, R 表示论域上的等价关系, $[x]_R$ 表示包含 x 的 R 等价类, 等价类内的所有对象在关系 R 中等价. 可以将上述统计属性值同现次数的问题转换为求解属性等价类交集的问题. 同时, 本文采用文献 [39] 给出的计算等价类算法 (参见文献 [39] 中的算法 1), 利用快速排序思想, 根据属性值预先对数据集进行排序处理, 文献 [39] 理论推导出该算法的时间复杂度降至 $\mathrm{O}(|A|\ |U| \lg |U|)$, 并实验证明了该算法可有效地提高计算等价类的运行效率.

对于任意 $a_i \in A$, 设 $x_{ki} \in V_{a_i}$, 从本地属性 a_i 的角度度量属性值 x_{ki} 单属性权重, 则可以定义为

$$W_{a_i}(x_{ki}) = \frac{|[x_k]_{a_i}|}{n} \times \lg\left\{|[x_k]_{a_i}| \times \left(1 - \frac{|[x_k]_{a_i}|}{n}\right) + 1\right\} \tag{1}$$

其中, $[x_k]_{a_i}$ 内的数据点在属性 a_i 上的取值与 x_k 相同, $|[x_k]_{a_i}|$ 反映了属性值 x_{ki} 在 a_i 上出现的次数. 参考文献 [30] 采用取对数运算, 可避免图 1 中属性 a_7 和 a_8 对聚类结果的干扰, 利用式 (1) 可得, $W_{a_8}(x_{18}) = 0.09$, 而属性 a_7 上任意属性值, 均有 $W_{a_7}(x_{k7}) = 0$.

从相关属性 a_j 的角度度量 x_{ki} 的多属性权重, 可以定义为

$$W_{a_j}(x_{ki}) = \frac{|[x_k]_{a_j} \cap [x_k]_{a_i}|}{|[x_k]_{a_i}|} \tag{2}$$

其中, $[x_k]_{a_j}$ 表示包含数据 x_k 的 a_j 等价类, $|[x_k]_{a_j} \cap [x_k]_{a_i}|$ 表示两等价类交集的元素个数, 即属性值 x_{ki} 与 x_{kj} 的同现次数, $0 < W_{a_j}(x_{ki}) \le 1$, $W_{a_j}(x_{ki})$ 的定义表明属性值 x_{kj} 与 x_{ki} 同现次数占 $[x_k]_{a_i}$ 的比例越大, 从 a_j 的角度上所反映的 x_{ki} 聚类作用越

大. 根据式 (2), 可证明多属性权值 $W_{a_j}(x_{ki})$ 的三条性质, 证明过程见附录 A.

用 $W(x_{ki})$ 表示综合权重, 并做归一化处理, 使得 $0 \le W(x_{ki}) < 1$, 计算公式定义为

$$W(x_{ki}) = \frac{W_{a_i}(x_{ki}) \times \left[\sum\limits_{j=1,\, j\neq i}^{d} W_{a_j}(x_{ki})\right]^2}{\sum\limits_{k=1}^{n}\left\{W_{a_i}(x_{ki}) \times \left[\sum\limits_{j=1,\, j\neq i}^{d} W_{a_j}(x_{ki})\right]^2\right\}} \tag{3}$$

基于粗糙集的多属性度量方法可以有效提高属性的聚类能力, 解决仅依靠单属性无法区分同频次属性值权重的问题. 例如图 1, 使用式 (1) 计算得, $W_{a_3}(x_{33}) = W_{a_3}(x_{43})$, 无法区分属性值 x_{33} 和 x_{34} 聚类能力的差别; 利用式 (3), 可得 $W(x_{33}) = 0.13$, $W(x_{43}) = 2 \times W(x_{33}) = 0.25$, 可解决第 2 节提出的单属性权重聚类区分度下降的问题.

3.2　基于多目标聚类准则的层次凝聚聚类

聚类准则是聚类过程中判断数据点划分的主要依据, 通常采用簇集质量函数表示. 簇集质量是指聚类结果的质量, 该值代表数据划分的合理程度, 簇集质量可以采用单目标和多目标两种方式, 多目标簇集质量更有利于挖掘数据集的内部结构.

3.2.1　多目标聚类准则依据

基于单目标准则的聚类算法, 常用最小化类内误差平方和的方法聚类, 该方法容易导致大类边缘的数据点被误划到其他类中, 因此, 在簇内数据紧凑的基础上, 采用簇间数据分离的原则可避免边缘数据误判, 有效处理非平衡数据的聚类问题.

综合簇内紧凑和簇间分离两个目标来评价簇集质量, 1) 簇集质量应取决于簇内数据是否紧凑, 簇内紧凑程度与簇投影到重要属性上的属性值相关, 该属性值权值越大, 出现的频率越高, 则簇集质量越好; 2) 簇集的质量也与簇间数据分离程度相关, 为了使不同的簇尽可能分离, 簇投影在重要属性上的属性值应尽量集中, 不同的簇在高权值属性维上的取值应尽可能不同.

3.2.2　多目标簇集质量判断

为了计算簇集总质量, 需要先分别计算各簇质量. 参照文献 [20] 和文献 [31], 基于属性值采用如下定义的 $Q(C_s)$ 来描述簇 C_s 的质量, 其质量值可分别从簇内紧凑度和簇间分离度来度量, $Com(x_{ki})$ 用于表示用属性值 x_{ki} 所度量的簇内紧凑度, $Sep(x_{ki})$ 是用属性值 x_{ki} 度量的簇间分离度.

$$Q(C_s) = \sum_{x_k \in C_s} \sum_{i=1}^{d} \left\{ [Com(x_{ki})]^2 \times Sep(x_{ki}) \right\} \tag{4}$$

$$Com(x_{ki}) = [P(a_i = x_{ki}) \times P(a_i = x_{ki}|C_s)] \times W(x_{ki}) = \frac{count(x_{ki}, a_i, C_s)}{n} \times W(x_{ki}) \tag{5}$$

$$Sep(x_{ki}) = \frac{P((a_i = x_{ki}) \wedge (x_k \in C_s))}{P(a_i = x_{ki})} = \frac{count(x_{ki}, a_i, C_s)}{count(x_{ki}, a_i)} \tag{6}$$

簇内紧凑度 $Com(x_{ki})$ 可以从两个方面评价: 属性值 x_{ki} 在簇 C_s 内的分布, 可由属性值 x_{ki} 在属性 a_i 上的概率 $P(a_i = x_{ki})$ 以及该值在簇 C_s 内的概率 $P(a_i = x_{ki}|C_s)$ 的乘积来度量, 该值主要体现了属性值 x_{ki} 在 C_s 上的集中程度; 属性值 x_{ki} 对于类簇的重要性, 由该值的权重 $W(x_{ki})$ 表示. 簇间分离度 $Sep(x_{ki})$ 则取决于属性值 x_{ki} 专属于簇 C_s 的程度, 可以用 x_{ki} 出现在簇 C_s 属性 a_i 上的概率 $P((a_i = x_{ki}) \wedge (x_k \in C_s))$ 与整个数据集上 x_{ki} 出现的概率 $P(a_i = x_{ki})$ 的比例表示, 该值越大, 说明 a_i 上的属性值 x_{ki} 越集中地出现在 C_s 中. 经推导, $Q(C_s)$ 可由以下符号组成的表达式表示: $count(x_{ki}, a_i, C_s)$ 表示在类簇 C_s 内, 投影在 a_i 上的值为 x_{ki} 的数据点数目; n 代表数据集的数据总量; $W(x_{ki})$ 是属性值 x_{ki} 的权值; $count(x_{ki}, a_i)$ 是指在属性 a_i 上 x_{ki} 出现的总次数.

假设簇集 $C = \{C_1, C_2, \cdots, C_k\}$, 采用 $Q(C)$ 描述簇集整体质量, 可用下式来刻画簇集质量.

$$Q(C) = \sum_{s=1}^{K} [P(C_s) \times Q(C_s)] \tag{7}$$

$Q(C)$ 反映了在簇集 C 的划分方式下, 数据分布的整体质量. 该值越大, 意味着这种划分方案越合理, $Q(C_s)$ 则表示簇 C_s 的质量, $P(C_s)$ 代表 C_s 中的数据点占整个数据集的比例, $P(C_s)$ 的主要作用是协调各簇之间的相互作用以达到最佳的整体聚类效果.

3.2.3　基于多目标聚类准则的层次凝聚聚类

在多目标聚类准则的基础上, 层次凝聚聚类过程可分为初聚类和合并聚类两个阶段. 初聚类阶段, 随机挑选一个数据点作为第一个子簇, 使用式 (7), 依次计算其他待聚类数据点与现有子簇的簇集质量, 根据簇集质量最大化原则, 决定该数据点分配到已有的某个子簇中或者生成的新子簇. 初聚类的目标是将数据中最相似的数据点划分为若干子簇, 子簇

具有纯度高、规模小的特点, 可以代表大类的多个类中心, 子簇的形成不仅可以有效避免非平衡数据的均匀化, 同时也为迭代合并阶段提高了时间效率; 初聚类阶段利用簇集质量函数, 按照簇集质量最大化原则, 将每个数据点依次选择分配到已有的子簇中或者生成的新子簇. 合并聚类的任务是找到同属一个类簇的所有子簇, 迭代合并各子簇以提高数据集的聚类质量, 直到整个迭代过程中没有产生合并子簇的操作为止; 该过程采用层次凝聚思想, 以式 (7) 度量的簇集质量函数最大化为基础, 将最相近的两个子簇合并. 该阶段充分利用了高纯度子簇, 来迭代合并最相似的子簇, 有效地加速了聚类形成的过程.

3.3 噪音点检测

噪音点是指那些与正常数据点有显著区别的特殊点. 对于各属性权重都相对偏低的数据点, 可认为投影到任何维上都无法找到相似点, 即为噪音异常点. 为了判断噪音点, 可引入如下定义的聚合度概念 $Os(x_k)$.

$$Os(x_k)=\sum_{i=1}^{d}W(x_{ki}) \tag{8}$$

$Os(x_k)$ 体现了数据点 x_k 在各属性维上的聚类能力. $Os(x_k)$ 值越小, 说明 x_k 越有可能是噪音点.

具体识别步骤: 按照聚合度 $Os(x_k)$, 先将所有数据点升序排序; 利用区间离散度 Ds_i^j, 找到最佳的划分点 x_m, m 的取值使满足 $\min(Ds_1^m+Ds_{m+1}^n)$, 即, 该划分点使得区间 $[x_1,x_m]$ 和区间 $[x_{m+1},x_n]$ 的离散度之和在所有划分点中最小; 以最小化区间离散度为目标, 采用类似 k-means 的聚类思想, 将数据点划分为两类, 其中, 低聚合度区间 $[x_1,x_m]$ 中的数据点就是聚类能力较差的噪音点.

上述步骤中提到的区间离散度用 Ds_i^j 表示, 采用方差来描述, 主要反映区间 $[x_i,x_j]$ 数据的分离程度:

$$Ds_i^j=\frac{1}{|j-i|}\times\sum_{k=i}^{j}[Os(x_k)-\overline{Os(x_k)}]^2 \tag{9}$$

$\overline{Os(x_k)}$ 表示区间 $[x_i,x_j]$ 上聚合度的平均值. $|j-i|$ 表示 j 与 i 差的绝对值. Ds_i^j 值越小, 说明区间内的点越接近.

3.4 属性相关子空间

属性相关子空间是由最能反映类簇特征的一组属性组成. 计算各属性相对于簇的依附程度是确定属性子空间的关键, 采用 $R(a_i,C_s)$ 刻画属性 a_i 对于簇 C_s 的依附程度. 参照式 (4), 依附程度可用如下表达式描述:

$$R(a_i,C_s)=\sum_{x_k\in C_s}\left\{\left[\frac{count(x_{ki},a_i,C_s)}{n}\right]^2\times[W(x_{ki})]^2\times\frac{count(x_{ki},a_i,C_s)}{count(x_{ki},a_i)}\right\} \tag{10}$$

属性子空间的识别方法与噪音点识别类似, 根据属性依附度 $R(a_i,C_s)$, 以区间离散度最小化为目标, 将降序属性区间 $[a_1,a_d]$ 分为区间 $[a_1,a_m]$ 和 $[a_{m+1},a_d]$, 高依附度区间 $[a_1,a_m]$ 就是属性子空间.

4 基于多属性权重的分类数据子空间聚类算法 SAC

依据第 2 节和第 3 节, 基于多属性权重的子空间聚类算法 SAC 基本思想, 可归纳为: 首先利用粗糙集概念分别计算属性值权重, 由式 (1) ∼ (3) 完成, 计算过程可由函数 WAC 来实现; 其次利用式 (8) 计算各数据点的聚合度, 采用式 (9) 划分数据区间, 判断并去除噪音点, 此过程由函数 NAI 完成; 利用式 (7), 实现将数据集到簇集的转化过程, 该过程由初聚类阶段和合并聚类阶段构成, 分别由两个函数 INC 和 MEC 实现; 最后基于式 (10) 给出的属性相对簇的依附度, 确定与簇相关的属性子空间, 具体实现由函数 SUI 完成. 具体算法描述如下:

1) 函数 WAC: 计算属性权重函数
输入. 属性取值 x_{ki};
输出. 权值 $W(x_{ki})$;
begin
保存 x_k 在各属性 a_i 上的等价类 $[x_k]_{a_i}$;
分别根据式 (1)、式 (2), 式 (3), 计算 $W_{a_i}(x_{ki})$, $W_{a_j}(x_{ki})$, $W(x_{ki})$, 并输出 $W(x_{ki})$;
end

2) 函数 NAI: 噪音点识别
输入. 数据集 U;
输出. 噪音集 NOS;
begin
根据式 (8) 计算各点聚合度 $Os(x_i)$, 并按照聚合度对 x_i 升序排序;
$minind=0$;
for $i=1$ to n
{将数据序列分割为 $L[x_1,x_i]$ 和 $H[x_{i+1},x_n]$;
根据式 (9), 分别计算区间 L 和 H 的离散度 $Ds(L)$ 和 $Ds(H)$;
$minind=\min((Ds(L)+Ds(H)),minind)$;}
$NOS\leftarrow L[x_1,x_{\text{minind}}]$, 并输出噪音集 NOS;
end

3) 函数 INC: 初聚类
输入. $\acute{U}=U-NOS$, $W(x_{ki})$;
输出. 子簇集 SC;

begin
$SC = \{\{x_1\}\}$;
for $i = 2$ to $|\acute{U}|$
{$Q_{\max} = 0$; $id_{\max} = 0$;
for $j = 1$ to $|SC|$
{$sum_1 = Q(\{x_i\} \cup sc_j)$;
$sum_2 = Q(\{x_i\}) + Q(sc_j)$;
if $(sum_1 > sum_2)$ && $(sum_1 > Q_{\max})$
{$Q_{\max} = Q(\{x_i\} \cup sc_j)$; $id_{\max} = j$;}
if $(id_{\max} > 0)$ $sc_{\text{idmax}} \leftarrow sc_{\text{idmax}} \cup \{x_i\}$;
else {x_i}作为新簇;}
输出子簇集 SC;}
end

4) 函数 MEC: 合并聚类
输入. 子簇集 SC;
输出. 最终的簇集 C;
begin
repeat
$maxind = 0$;
for $i = 1$ to $|SC|$
for $j = 1$ to $|SC|$
{利用式 (7), 计算 SC_i 与 SC_j 合并后的簇集质量 $Q(SC(t))$;
$s = Q(SC(t)) - Q(SC(t-1))$, $SC(t-1)$ 是上次迭代的簇集质量值;
if $(maxind < s)$
{$maxind = s$; $\max_i = i$; $\max_j = j$;}}
合并 $SC_{\max_i}$ 和 $SC_{\max_j}$;
until $(maxind == 0)$;
$C \leftarrow SC$, 并输出簇集 C;
end

5) 函数 SUI: 相关子空间识别
输入. 簇集 C;
输出. 各簇 C_s 的属性相关子空间 SA_s;
begin
用式 (10), 计算 C_s 内各属性维 a_j 的依附度 $R(a_j, C_s)$, 并将簇内各属性降序排序;
$minind = 0$;
for $i = 1$ to d
{将属性序列分割为 $L[a_1, a_i]$ 和 $H[a_{i+1}, a_d]$;
$minind = \min$(区间 L 离散度 + 区间 H 离散度, $minind$)};
$SA_s \leftarrow H[a_1, a_{\text{minind}}]$, 输出 SA_s;
end

上文描述的 SAC 算法, 运行时间随数据量的增加呈线性增长. 该算法主要由三个函数组成, 分别为属性权重计算、初步聚类和合并聚类, 因此, 算法 SAC 的时间复杂度分析主要包含以下三个阶段: 设 $n = |U|$, $d = |A|$, $s = MAX_{a_i \in A}|V_{a_i}|$, k 代表子簇的数量, $c = MAX_{sc_s \in SC}|sc_s|$ (sc_s 是初聚类形成的子簇), t 为最大迭代次数.

1) 计算属性值权重的时间复杂度. 该阶段的时间消耗主要集中在相关属性对属性值权重的度量上, 由于算法 SAC 采用粗糙集的等价类概念, 统计同现属性值的比较范围由 n 缩小至 s, 该阶段的时间复杂度为 $\mathrm{O}(n \times s \times d^2)$.

2) 初步聚类的时间复杂度. 该阶段的主要操作是计算子簇内数据属性值对簇集质量的影响, 该阶段的时间复杂度为 $\mathrm{O}(n \times k \times c \times d)$.

3) 合并聚类的时间复杂度. 该阶段的时间消耗主要集中在寻找可合并子簇的操作上, 该阶段的时间复杂度为 $\mathrm{O}(t \times c \times d \times k^2)$.

综合上述三个阶段的分析, 算法 SAC 时间复杂度为 $\mathrm{O}(n \times s \times d^2 + n \times k \times c \times d + t \times c \times d \times k^2)$. 由于真实海量数据, $n \gg d$, $n \gg s$, $n \gg c$, 相对于 n 而言, 算法 SAC 所消耗的时间是线性增长的.

5　实验及分析

实验环境: 3.2 GHz Intel Core i5 处理器, 2 GB 内存, windows 7 的操作系统, ECLIPSE 1.0 的编辑环境, 并采用 Java 7.0 语言实现了本文 SAC 算法、CLICKS[29] 算法、PROCAD[30] 算法以及 EWKM[35] 算法, AT-DC[31]、DHCC[32] 算法代码由作者提供. 上述 5 种算法用于进行对比实验, 这些算法均可以解决高维分类数据的聚类问题. 其中, CLICKS 算法是一种基于图论的聚类算法, 该算法以属性值同现信息作为聚类依据; PROCAD 算法是针对分类数据的子空间聚类算法, 该算法以属性值在单属性上的频率计算属性权重; AT-DC 和 DHCC 是无参数的层次聚类的典型算法, 其中, AT-DC 算法以属性值在簇中的条件概率作为属性权重, DHCC 算法采用多元对应分析思想决定各属性在聚类过程中的作用; EWKM 算法是基于经典 K-Modes 算法的改进算法, 利用最小化簇内误差平方和的思想计算属性权重. SAC 算法是一种分类数据的子空间聚类算法, 算法基于粗糙集, 利用多属性维取值的频次衡量各属性权重, 采用层次凝聚策略聚类. SAC 算法、PROCAD 算法、AT-DC 算法和 DHCC 算法均是无参算法, 无需设置参数; CLICKS 算法的主要参数包括 k, $\boldsymbol{\alpha}$ 和 min_{sup}, 其中, k 设置为实际簇数, $\boldsymbol{\alpha}$ 和 min_{sup} 从 0.1 ∼ 0.9 按步长 0.1 取值实验, 实验取平均值作为实验结果; 算法 EWKM 的参数主要包括 k, m, $\boldsymbol{\alpha}$ 和 $\boldsymbol{r}$, 其中, k 为实际簇数, m 和 $\boldsymbol{\alpha}$ 设为 1, $\boldsymbol{r}$ 分别设为 1, 2 和 5. 数据集主要分为人工合成数据集、UCI 数据集以及真实光谱数据三大类, 表 2 详细描述了实验数据集的具体信息.

聚类算法的评价指标可以分为内部和外部两类, 外部评价指标通常适用于有人工标示聚类结果的数据集, 使用聚类算法所得到的聚类结果与根据人的先验知识给出的结果进行对比, 可以有效地评价聚类算法是否适合于该数据集. 参照文献 [30−31], 分别采用调整兰德系数 ARI (Adjusted rand index)、

表 2 数据集
Table 2 Dataset

类别	数据集	数据量	维数	簇数	平衡数据	实验目的
合成数据	Type 1	10 000	50	6	是	抗噪性对比实验
	Type 2	10 000	200	6	是	维度可扩展性对比实验
	Type 3	60 000	50	6	是	数据量可扩展性对比实验
	Type 4	10 000	50	6	否	非平衡数据集聚类效果对比实验
UCI	Voting	435	16	2	否	属性值域元素少对聚类效果的影响
	Splice	3 190	60	3	否	高维 UCI 数据集的效果对比实验
	Mushroom	8 124	22	2	是	大数据量聚类效果对比实验
	Zoo	101	17	7	否	多簇数据集性能实验; 属性子空间实验
真实数据	光谱数据	8 315	208	7	否	真实数据聚类效果对比实验

纯度 Purity、雅克比系数 Jaccard、兰德指数 RI (Rand index) 四个外部评价指标, ARI 主要测试聚类结果和真实类之间的相似度. Purity 评测聚类结果中各簇起支配作用的数据比例. Jaccard 系数主要判断隶属于同一类的数据对在同一簇的比例. RI 主要评测同类数据对是否被聚合以及异类数据对是否可以分开. 其中, ARI, Jaccard 和 RI 三个指标不仅注重对比两种结果中各簇内元素的分布状况, 更偏重两种聚类结果在类簇数目上的差距; 而指标 Purity 更侧重于衡量簇内占主导地位数据点的比例.

5.1 人工合成数据集

为了全面测试算法的聚类效果, 结合文献 [40] 给出的合成数据方法, 构造了图 2 所示的四种合成数据集.

如图 2 (a) 所示, 数据集 Type 1 各簇之间的数据点以及属性相关子空间独立存在, 互不相交; Type 2 代表簇间数据点有交集, 如图 2 (b) 所示, 从不同的子空间分析, x_3 可以分属于两个不同的簇; Type 3 代表各簇属性相关子空间有交集, 如图 2 (c), 属性 a_2 既属于簇 C_1 的相关子空间, 又属于 C_2 的子空间; 如图 2 (d), Type 4 与 Type 1 数据分布特征一致, 不同的是, Type 4 是簇尺寸差异较大的非平衡数据集.

5.1.1 可扩展性

在数据集 Type 3 上, 主要测试各算法在数据容量上的可扩展性, 分别选取 6 000, 10 000, 20 000, 40 000 和 60 000 作为数据量测试节点, 实验结果如图 3. 从图 3 可以发现, 随着数据量的增加, 所有算法的聚类时间都呈现近似线性增长的趋势, SAC 算法所消耗的时间在所对比的算法中仅比 PROCAD 少. 在数据集 Type 2 上, 主要测试各算法在属性维度上的可扩展性, 维度测试节点分别选择 50, 100, 150 和 200 维, 结果如图 4. 如图 4 所示, 随着维度的增加, 除 CLICKS 以外, 其余算法的聚类耗时呈现线性增长的态势, SAC 算法消耗的时间虽然比算法 AT-DC 和 DHCC 多, 但远低于 CLICKS 算法, 略低于 PROCAD 算法.

从图 3 可知, SAC 算法与 PROCAD 算法时间成本高于其他三种算法, 原因是这两种算法在衡量属性权重时, 需要计算数据点所有属性值的重要度, 当数据量增加时, 会比以整个属性为度量单位的算法费时. 从图 4 可知, CLICKS 算法的时间效率明显受限于维数, 原因是该算法将聚类问题转化为有权图划分问题, 而图上的权值取决于数据点之间的同现次数, 当属性维增多或者属性值域元素增多时, 统计各维度上所有属性值的同现次数非常耗费时间. 尽管 SAC 算法基于属性值间的同现频次计算权重, 随着数据集属性规模的增长, 会增加计算成本, 但该算法在聚类过程中采用粗糙集缩小属性查找范围, 缩短权重计算时间, 同时基于子簇的合并过程可以大大提高聚类迭代速度, 因此, 图 3 与图 4 所示的扩展性实验结果表明, SAC 算法的运行时间随数据量和维度呈线性增长.

5.1.2 抗噪性

抗噪性实验主要包括两方面: SAC 算法在合成数据上对比不同维度之间的抗噪性能变化以及在真实光谱数据上的抗噪性能; SAC 算法与其他 5 种聚类算法在合成数据集上的抗噪性对比实验.

在 Type 1 数据集上, 数据总量为 10 000, 分别包含 900, 1 800, 3 150 和 4 500 个噪音点作为测试节点, 并观察在 15, 30, 40 和 50 维属性空间上算法 SAC 抗噪性能的变化. 图 5 (a) 和图 5 (b) 分别反映了在不同维度空间下, 噪音点数量的变化对算法

		a_1	a_2	a_3
C_1	x_1	a		
	x_2	a		
C_2	x_3		b	c
	x_4		b	c

(a) 数据集 type 1 示例
(a) Type 1 sample

		a_1	a_2	a_3
C_1	x_1	a	b	
	x_2	a	b	
C_2	x_3	$\underline{a}$	$\underline{b}$	$\underline{c}$
	x_4			c

(b) 数据集 type 2 示例
(b) Type 2 sample

		a_1	a_2	a_3
C_1	x_1	a	$\underline{b}$	
	x_2	a	$\underline{b}$	
C_2	x_3		$\underline{b}$	c
	x_4		$\underline{b}$	c

(c) 数据集 type 3 示例
(c) Type 3 sample

		a_1	a_2	a_3
C_1	x_1	a		
C_2	x_2		b	c
	x_3		b	c
	x_4		b	c

(d) 数据集 type 4 示例
(d) Type 4 sample

图 2　合成数据集示例图

Fig. 2　Synthetic data sets sample

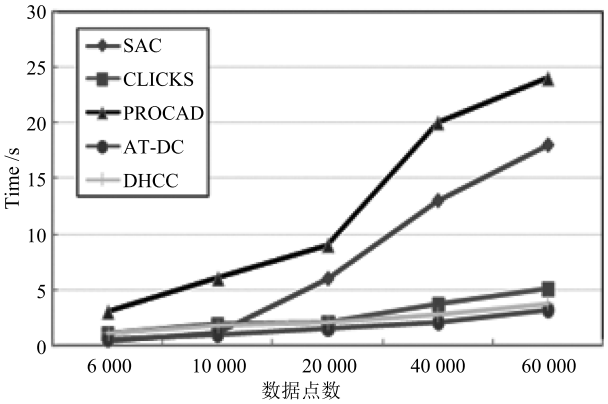


图 3　数据量可扩展性

Fig. 3　Scalability with data

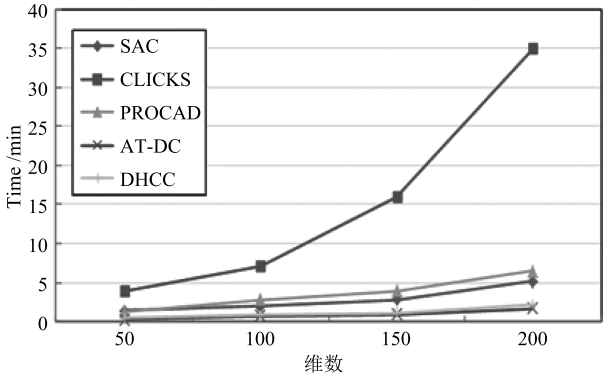


图 4　维度可扩展性

Fig. 4　Scalability with dimendionality

SAC 聚类指标 ARI 和 Purity 的影响. 从图 5 可知, SAC 算法在噪音点的影响下, 性能较稳定, ARI 与 Purity 指标均高于 90 %, 尤其是 50 维的属性空间, 所有指标均在 96 % 以上, 说明 SAC 算法具有良好的抗噪性能. 由图 5 (a) 和图 5 (b) 可知, SAC 算法的抗噪能力与属性空间规模, 噪音点比例有关. 在加入相同噪音点的数据环境下, 抗噪性能与属性空间的维数成正比. 当属性空间越小、噪音点越多, 聚类效果所受影响越大, 其主要原因是 SAC 算法是基于属性之间的同现概率获取属性权重, 当属性空间过小时, 不利于挖掘簇的聚类模式.

为了测试 SAC 算法在真实数据上的抗噪能力, 选择包含 8 315 条 208 维恒星光谱记录的数据集, 分别加入 1 000, 2 000 和 3 000 条噪音记录. 表 3 显示在不同噪音环境下, SAC 去除噪音点的数目以及各性能指标的变化, 表明噪音点对真实数据的聚类影响以及 SAC 算法的抗噪能力. 从表 3 可知, 与无噪音点时 SAC 性能指标相比, 所有指标值均下降, 其中, ARI 下降最多, Purity 最少; 同时, 聚类性能下降的程度与噪音数目成正比. 算法 SAC 可以识别并去除大部分噪音点, 少量残留的噪音基本上会聚为若干类簇, 这些类簇仅包含噪音点, 这样的聚类结果对 Purity 影响最小, 但增加的类簇数目使得其他性能指标显著下降.

在 Type 1 数据集上, 选取 50 维属性, 10 000 个数据点构成测试数据, 包含噪音点数分别为 900, 1 800, 3 150 和 4 500. 随着噪音点数的增加, 对比 SAC 算法和其他五种算法的聚类效果. 图 5 (c) 反映了噪音点数量的变化对不同算法聚类指标 ARI 的影响. 从图 5 (c) 可知, 算法 SAC 和 PROCAD 在抗噪性上的表现均优于其他 4 种算法, 原因是算法 SAC 和 PROCAD 都具有检测噪音点的能力, 聚类之前已去除了噪音点的干扰, 保证聚类性能的稳定. 同时, 由于多属性权重有助于更精准地反映出数据点各属性的聚类能力, 可以更准确地检测到噪音点, 因此, 基于单属性权重算法 PROCAD 的抗噪性比算法 SAC 略低.

5.1.3　不同类型数据集的聚类性能对比

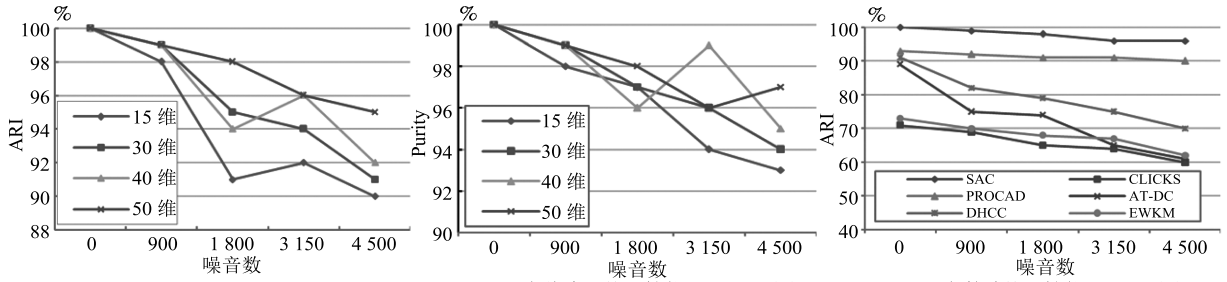
采用图 2 所描述四个合成数据集 Type 1, Type 2, Type 3, Type 4 作为实验数据集. 四个数据集规模相同, 即 10 000 个数据点, 50 维属性, 可聚为 6 个簇; Type 1, Type 2, Type 3 是平衡数据, 各簇尺寸基本一致, Type 4 簇的尺寸差异较大, 其中, C_1 和 C_2 有 3 000 点, 是其他簇的三倍.

图 6 显示在四种合成数据集上, 随维度变化, SAC 聚类指标 ARI、Purity 的变化情况. 由图 6 可知, 尽管 Type 1 和 Type 4 数据分布相同, 但簇的尺寸有差异, 特别是 Type 4 各簇规模差距很大. SAC 算法在 Type 1 和 Type 4 上的聚类效果良好, 各指

表 3 SAC 在光谱数据上性能指标变化

Table 3 Noise immunity on the spectral data for SAC

加入噪音数	去除噪音数	ARI (%)	Purity (%)	Jaccard (%)	RI (%)
1 000	993	2.56↓	0.13↓	2.28↓	2.08↓
2 000	1 984	2.94↓	0.57↓	2.72↓	2.21↓
3 000	2 971	3.25↓	0.92↓	3.01↓	2.52↓



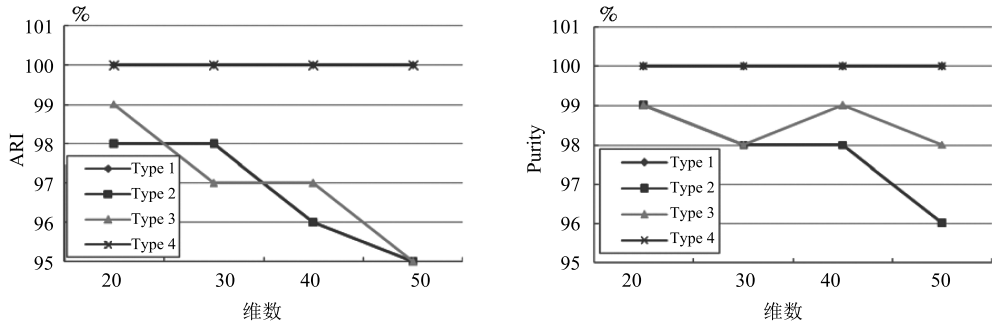
(a) 各维度下抗噪性能 (ARI) 对比 (a) ARI for various dimensions

(b) 各维度下抗噪性能 (Purity) 对比 (b) Purity for various dimensions

(c) 各算法抗噪性能 (ARI) 对比 (c) Experiment (ARI) for all algorithms

图 5 抗噪性实验

Fig. 5 Immunity to noise



(a) ARI 值 (a) ARI

(b) Purity值 (b) Purity

图 6 SAC 算法在四种数据集上的对比实验

Fig. 6 Contrast experiment on four data sets for SAC

标均可达到 100 %, 说明 SAC 算法的聚类效果与簇尺寸是否平衡无关; 而 Type 2 和 Type 3 的实验结果显示, SAC 聚类效果与数据分布紧密相关, 随着维度的增加, Type 2 和 Type 3 数据分布的变化可能会造成最终聚类性能的下降.

由于大簇的边缘数据容易受到小簇类中心的干扰, 单一类中心无法反映大簇特征等原因, 非平衡数据往往更容易聚成尺寸相等的类簇. 由图 6 Type 1 和 Type 4 上的聚类效果可知, SAC 算法没有受到类簇尺寸差异的影响. 原因是 SAC 算法利用多目标聚类准则, 强化大类边缘数据与小簇之间的类间分离作用; 同时分阶段的层次聚类过程采用子簇的形式, 利用多个类中心表示大类, 从而避免出现 "均匀效应" 的现象. 在 Type 2 和 Type 3 上, SAC 算法的 ARI 和 Purity 值, 随着维数增加而下降. 原因是随数据维度增加, Type 2 中数据交集和 Type 3 中的属性子空间交集范围均在扩大, 当交集比例过大时, 类簇之间的差异会随之减少, 在合并聚类阶段, 原本分属两个类的数据点会聚为一类.

图 7 显示在 50 维, 10 000 个数据点的数据规模上, SAC 与其他 5 种算法在四类数据集上的 ARI 性能指标差异. 从图 7 可见, SAC 算法在四类数据集上的聚类效果均优于其他算法, 并且该算法在不同数据集上的聚类性能较稳定.

文献 [15] 指出, CLICKS 算法在数据集 Type 2 和 Type 3 上聚类性能不佳的主要原因是该算法会产生大量冗余簇; 而算法 PROCAD、DHCC 和 AT-DC 在四类数据集上的性能差别不大, 尽管算法

DHCC 和 AT-DC 无法给出簇集的相关子空间, 但在高维属性空间下, 这三类算法对数据集结构不敏感; 算法 EWKM 是基于 K-Modes 提出的, 这类算法在识别尺寸差别较大的簇集时, 会出现 “均匀效应”, 因此, 该算法在数据集 Type 4 上的聚类性能会降低.

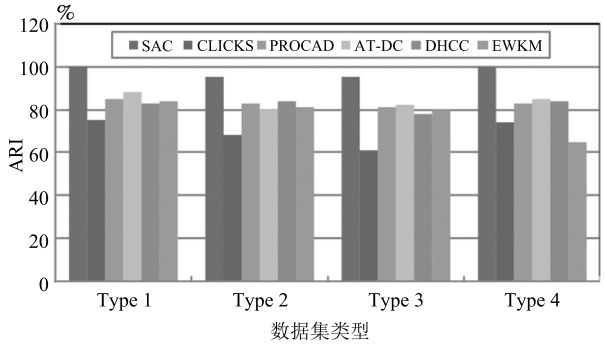


图 7　各算法在四种数据集上的 (ARI) 对比

Fig. 7　Contrast experiment on four data sets (ARI) for all algorithms

5.2　UCI 数据集

实验分别选取 Voting, Splice, Mushroom 和 Zoo 四个 UCI 数据集, 这些数据集均由分类数据组成, 其中, Zoo 中只含有一个数值型属性; 数据集均带有类别标识, 可以用外部评价指标, 更客观地度量各算法的聚类效果.

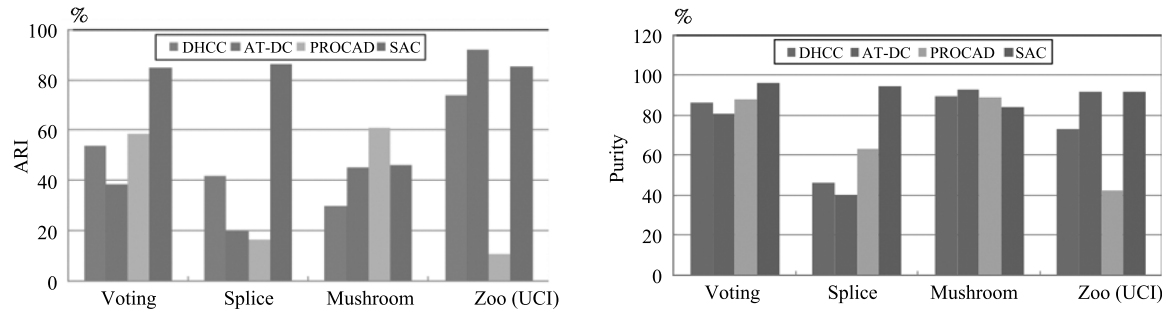
Voting 数据集共包含 435 条记录, 每条记录由 16 个议题的投票结果和一个类标签构成; 属性值域 $\{Y, N\}$; 所有的记录可以分为两个簇, 其中 267 条记录是 democrat 的投票结果, republican 的投票记录有 168 条. Splice 数据集共由 3 190 个数据点, 60 个属性维以及一个类标签构成; 每一个数据点是一组基因序列, 属性值对应该序列上的一个位置, 属性值域为 $\{A, T, G, C\}$; 数据点共分为三个簇, 分别是 EI (767 个数据点)、IE (768 个数据点) 和 Neither (1 655 个数据点). Mushroom 数据集包含 8 124 个数据点, 22 个属性; 各属性分别用来描述蘑菇的各种性状, $\max|V_{a_i}| = 10$; 数据集共分为两类, 共有 3 916 条 poisonous 记录表示该蘑菇有毒, 4 208 个带有 edible 标签的数据表示可食用. Zoo 数据集共有 101 个数据点, 17 个属性, 1 个类标签; 数据集可以分为 7 类, 分别为 Mammal (41 个数据点)、Bird (20 个数据点)、Reptile (5 个数据点)、Fish (13 个数据点)、Amphibian (4 个数据点)、Insect (8 个数据点) 和 Invertebrate (10 个数据点).

从表 4 和图 8 可知, SAC 算法在 Voting 和 Splice 上的聚类效果均优于其他三种算法; 在 Mushroom 上, PROCAD 算法 ARI、Jaccard 和 RI 指标均高于其他算法; 而 AT-DC 算法在 Zoo 上的实验效果最优. 其主要原因: 1) SAC 在 Voting 和

表 4　UCI 数据集上的算法性能

Table 4　Algorithm performance on UCI

算法	UCI	ARI (%)	Purity (%)	Jaccard (%)	RI (%)
DHCC	Voting	53.67	86.67	63.17	76.84
	Splice	41.75	46.08	38.96	75.11
	Mushroom	29.90	89.50	38.22	63.52
	Zoo	73.79	73.26	68.05	89.21
AT-DC	Voting	38.5	80.92	54.23	69.28
	Splice	19.78	40.22	35.31	61.28
	Mushroom	45.20	93.21	48.81	72.63
	Zoo	92.00	92.08	88.55	97.03
PROCAD	Voting	58.49	88.28	66.39	79.11
	Splice	16.57	63.29	34.31	59.30
	Mushroom	61.10	89.11	68.22	80.56
	Zoo	10.91	42.57	23.79	28.81
SAC	Voting	84.88	96.09	86.80	92.47
	Splice	86.26	94.76	84.23	93.54
	Mushroom	46.21	84.25	59.62	73.13
	Zoo	85.37	92.08	79.53	94.95



(a) ARI 值 (a) ARI (b) Purity 值 (b) Purity

图 8 UCI 数据集上的算法性能

Fig. 8 Algorithm performance on UCI

Splice 数据集上的效果最优, 原因是这两类数据集的共同特点是属性值域元素个数都较少, 其中, Congressional Voting 的属性值是布尔型; 数据集 Splice 的属性值仅有四个, 通常属性值的取值范围较小, 会造成单属性权值的区分度不明显, 进而影响到基于单属性聚类算法的最终聚类结果, 而 SAC 算法利用多属性的同现概率可以很好地解决这一问题, 进一步提高算法的聚类效果, 该结论可由表 5 和表 6 的聚类结果得到验证. 2) SAC 在 Mushroom 上的聚类性能仅次于 PROCAD, 而在 Zoo 上也是略低于 AT-DC, 原因是 SAC 算法用式 (3) 强化了多属性频率对属性权重的影响, 所以, 即使同一个类的数据, SAC 会根据子簇模式上的差异划分出若干纯度极高的子簇, 经过合并所形成的簇集纯度依旧非常高, 但同时也会造成簇划分较细的问题, 簇集细化会保证算法的聚类纯度, 但也影响了其他三项指标, 该现象在表 7 和表 8 中都有体现.

表 5 Voting 集上的实验结果

Table 5 Experimental results on Voting

算法	簇的总数	簇编号	democrat	republican
DHCC	2	0	49	159
		1	218	9
AT-DC	2	0	186	1
		1	81	166
PROCAD	2	0	226	10
		1	41	158
SAC	2	0	2	153
		1	265	15

5.3 属性相关子空间

Zoo 数据集规模较小, 但属性维较多, 有利于分析结果, 且该数据集所对应的类别较多, 可以更全面地反映出数据中不同结构的差异, 因此 Zoo 数据集有利于测试 SAC 算法在属性相关子空间上的识别能力.

表 6 Splice 集上的实验结果

Table 6 Experimental results on Splice

算法	簇的总数	簇编号	EI	IE	Neither
DHCC	6	0	15	10	419
		1	668	19	28
		2	40	3	393
		3	11	0	369
		4	28	728	34
		5	5	8	412
AT-DC	3	0	55	78	513
		1	151	644	1 058
		2	561	46	84
PROCAD	4	0	304	0	0
		1	462	747	687
		2	1	20	909
		3	0	1	59
SAC	7	0	765	82	37
		1	2	0	1
		2	0	682	45
		3	0	1	1 566
	4	0	2	0	
		5	0	1	4
		6	0	0	2

从表 9 可知, SAC 算法的聚类结果可以分为三类: 划分完全准确的簇, 例如 C_1 和 C_3 等; 应该分裂却聚在一起的簇, 例如 C_2 和 C_5 等; 应聚合为一个簇, 却分为两个簇, 例如 C_4、C_8 和 C_9. 针对上述三类情况, 分别从数据特点的角度, 对 SAC 算法聚类结果的合理性进行分析.

1) 被准确划分的数据点特征是: 该类数据间具有很强的结构特征, 同时, 簇内数据点在该结构特征上属性取值也非常一致. 例如 C_1 由 37 种哺乳动物构成, 其相关子空间上各属性的含义, 分别是有毛发、非蛋生、哺乳、无鳍和有 4 条腿, 这些特征是哺乳动物的共同典型特征.

表 7 Zoo 集上的实验结果

Table 7 Experimental results on Zoo

算法	簇的总数	簇编号	Mammal	Bird	Reptile	Fish	Amphibian	Insect	Invertebrate
DHCC	3	0	41	0	0	0	0	0	0
		1	0	0	5	13	4	0	8
		2	0	20	0	0	0	8	2
AT-DC	7	0	41	0	2	0	1	0	0
		1	0	0	0	0	0	0	7
		2	0	0	0	0	3	0	0
		3	0	0	0	0	0	8	2
		4	0	20	0	0	0	0	0
		5	0	0	3	13	0	0	0
		6	0	0	0	0	0	0	1
PROCAD	2	0	41	18	5	12	4	7	10
		1	0	2	0	1	0	1	0
SAC	9	0	37	0	0	0	0	0	0
		1	4	0	0	13	0	0	0
		2	0	20	0	0	3	0	0
		3	0	0	0	0	0	0	6
		4	0	0	0	0	0	8	1
		5	0	0	2	0	3	0	0
		6	0	0	3	0	1	0	0
		7	0	0	0	0	0	0	2
		8	0	0	0	0	0	0	1

表 8 Mushroom 集上的实验结果

Table 8 Experimental results on Mushroom

算法	簇的总数	簇编号	poisonous	edible
DHCC	10	0	24	32
		1	1 728	0
		2	1 296	0
		3	0	16
	4	736	2 880	
		5	72	808
		6	0	216
		7	44	0
		8	16	64
		9	0	192
AT-DC	8	0	360	3 536
		1	0	288
		2	1 734	0
		3	456	192
		4	1 296	0
		5	64	0
		6	0	129
		7	6	0
PROCAD	2	0	3 050	20
		1	866	4 188
SAC	3	0	1 258	4 185
		1	2 615	23
		2	43	0

2) 本属异类却聚合在一起的数据点特征是: 该类数据点在某种划分方式下可以分为两类, 而从其他分类角度上分析, 这些数据又具有一些共同的特殊特征, 可以归为一类. 例如, C_2 是由 76.5 % 的鱼类和 23.5 % 的哺乳动物组成, 其中作为哺乳动物的 Dolphin (海豚)、Porpoise (鼠海豚)、Seal (海豹) 和 Sealion (海狮) 有着明显水生动物的特征, 例如它们有鳍, 具有水生能力等, 所以从水生动物的角度上分析, 这 4 种动物和 fish 类的数据可以聚合在一起.

3) 本属同类却分为两个簇的数据点特征是: 该类数据点虽然同属一类, 但数据点仍具有各自特点. 例如, 同属于无脊椎动物的 Octopus (章鱼)、Scorpion (蝎子) 和 Starfish (海星), 根据腿数的不同也可以分为 C_8 和 C_9 两类.

从上述分析中可知, SAC 算法在识别属性相关子空间时, 可以识别出特征信息明显的关键属性子集作为属性子空间, 该属性子集不仅在簇内具有高权值的特点, 而且高权值的属性取值又具有专属性, 例如, 属性 Legs 在每个簇上的取值几乎都不同, 原因是多属性权重计算方法在挖掘重要属性时更注重该属性在整个属性空间中的分布特征, 并非仅依靠单属性的出现频率, 因此, 基于多属性频率的属性权重聚类方法所提取的属性子空间对聚类结果更具解释性.

表 9 Zoo 数据集中的相关子空间
Table 9 Relevant subspace on Zoo data set

簇号	簇内数据分布	子空间上的属性及其取值
C_1	100 % Mammal	hair = 1; eggs = 0; milk = 1; fins = 0; legs = 4
C_2	76.5 % Fish; 23.5 % Mammal	fins = 1; legs = 0; aquatic = 1
C_3	100 % Bird	feathers = 1; airborne = 1; legs = 2
C_4	100 % Invertebrate	backbone = 0; legs = 0
C_5	88.8 % Insect; 11.2 % Invertebrate	backbone = 0; legs = 6
C_6	60 % Amphibian; 40 % Reptile	hair = 0; predator = 1; toothed = 1
C_7	75 % Reptile; 25 % Amphibian	breathes = 1; legs = 4; tail = 1
C_8	100 % Invertebrate	legs = 8
C_9	100 % Invertebrate	legs = 5

5.4 天体光谱数据

根据国家天文台提供的恒星光谱数据, 选取了 8 315 条 208 维光谱记录作为实验数据, 该数据集按照恒星温度由低至高排序可以将恒星光谱分为 7 类 (分别用 A, B, F, G, K, M 和 O 表示)[41−42]. 由于光谱数据的大部分属性是数值型数据, 需要按照不同离散程度将光谱数据前 200 维属性做等距离散化处理.

表 10 给出各种等距离散化方法与 SAC 聚类性能指标之间的关系. 指标 Purity 与离散化程度是单调递增的关系, 而其他三项指标会呈先扬后抑趋势. 造成该现象的主要原因是 SAC 算法擅长处理具有独立不相关的属性子空间 (如 Type 1) 的数据集, 离散化程度越高, 对于簇的特征划分的越细致, 因此, 聚类结果中的大量高纯度簇, 会提高聚类结果的纯度, 但划分过细也会造成其他三项性能指标的下降.

表 10 各种离散化方法下的 SAC 聚类性能指标对比
Table 10 Clustering performance of different discretization methods

	ARI (%)	Purity (%)	Jaccard (%)	RI (%)
7 等距离散	45.18	67.55	36.86	85.20
11 等距离散	46.65	84.02	36.12	87.38
15 等距离散	41.75	91.34	31.06	87.13
18 等距离散	36.68	91.61	26.61	86.46
22 等距离散	31.40	92.71	22.22	85.78

使用 11 等距离散化方法预处理光谱数据, 并实验对比 SAC 算法与其他 5 种算法在真实数据上的聚类性能差异. 从表 11 可知, 算法 SAC 的各性能指标均高于其他算法, PROCAD、DHCC 和 AT-DC 性能无明显差异, 算法 EWKM 的性能最低. 光谱数据实验得到了与合成数据相似的实验结果, 主要原因是光谱数据是非平衡数据, 其中类 A 的规模是类 O 的 3 倍, 算法 EWKM 不擅长处理这类数据, 聚类性能较差; 尽管光谱数据所形成的类簇较多, 但各簇的值域元素较多, 同时具有高维度, 因此, 算法 PROCAD 和 DHCC 的聚类性能均较稳定.

表 11 各算法在光谱数据上的对比实验
Table 11 Algorithm performance on spectral data

	ARI (%)	Purity (%)	Jaccard (%)	RI (%)
SAC	46.65	84.02	36.12	87.38
PROCAD	43.41	80.18	32.41	82.72
DHCC	40.79	78.31	31.23	79.67
AT-DC	41.38	79.84	31.86	80.21
EWKM	33.47	70.08	30.11	71.64
CLICKS	40.65	71.66	30.83	75.79

5.5 结束语

采用属性相关子空间, 挖掘隐藏在高维属性空间中的簇是高维分类数据聚类领域中的研究难点和热点之一. 本文基于粗糙集采用多属性频率权重, 提出一种分类数据子空间聚类算法 SAC. 该算法基于粗糙集采用多属性频率计算属性权重, 解决了使用单属性计算权重所带来的问题, 同时借助粗糙集可以提高统计多属性频率的时间效率; 针对噪音点对聚类效果的影响, 使用区间离散度方法, 有效地删除了噪音点; 基于层次凝聚聚类思想, 利用多目标簇集质量函数迭代合并子簇, 解决单目标聚类函数聚类结果不精确的问题; 利用属性相对簇的依附度, 提取属性相关子空间, 提高了聚类簇的可理解性; 大量的实验结果验证了算法 SAC 的正确性和有效性. 下一步研究工作是将 SAC 算法由分类数据扩展到混合数据.

附录 A

性质 A1. 当 $x_k \in \underline{a_j}([x_k]_{a_i}) \cap \underline{a_i}([x_k]_{a_j})$ 时, 属性值 x_{ki} 的多属性权值最大, 其值为 1.

证明. 由下近似集的定义可知: $\underline{a_j}([x_k]_{a_i}) = \{x \in U | [x_k]_{a_j} \subseteq [x_k]_{a_i}\}$, 且 $\underline{a_i}([x_k]_{a_j}) = \{x \in U | [x_k]_{a_i} \subseteq [x_k]_{a_j}\}$.

因为 $x_k \in \underline{a_j}([x_k]_{a_i}) \cap \underline{a_i}([x_k]_{a_j})$, 可得, $[x_k]_{a_j} \subseteq [x_k]_{a_i}$, 又因为 $[x_k]_{a_i} \subseteq [x_k]_{a_j}$, 所以可得, $[x_k]_{a_i} = [x_k]_{a_j}$, 可得, $[x_k]_{a_j} \cap [x_k]_{a_i} = [x_k]_{a_i} \Rightarrow |[x_k]_{a_j}| = |[x_k]_{a_i}|$, 可得, $W_{a_j}(x_{ki}) = 1$.

该情况表明, 对于数据 x_k, 在属性 a_i 的取值为 x_{ki}, 则在属性 a_j 的取值一定为 x_{kj}. 用属性 a_j 度量属性值 x_{ki} 的权重应该取最大值. 例如图 1 中 x_{11}, 用属性 a_2 计算 x_{11} 权重, 可得 $W_{a_2}(x_{11}) = 3/3 = 1$. □

性质 A2. 当 $[x_k]_{a_j} \cap [x_k]_{a_i} = \{x_k\}$ 时, 属性值 x_{ki} 的多属性权重取值最小, 值为 $1/|[x_k]_{a_i}|$.

证明. 因为 $[x_k]_{a_j}$ 与 $[x_k]_{a_i}$ 的交集内只有一个数据点 x_k, 所以 $|[x_k]_{a_j} \cap [x_k]_{a_i}| = 1$, 可得,

$$W_{a_j}(x_{ki}) = \frac{|[x_k]_{a_j} \cap [x_k]_{a_i}|}{|[x_k]_{a_i}|} = \frac{1}{|[x_k]_{a_i}|}$$

当 $[x_k]_{a_j} \cap [x_k]_{a_i} = \{x_k\}$ 时, 说明属性值 x_{ki} 和 x_{kj} 仅同时出现一次. 即使属性值 x_{ki} 单属性权值较大, 但从属性 a_j 上看, x_{ki} 与 a_j 的相关性较低, 该属性值的全局重要性较低, 例如图 1, 属性值 x_{41} 在属性 a_2 下的多属性权重值, $W_{a_2}(x_{41}) = 1/3 = 0.33$.□

性质 A3. 当 $x_k \in \overline{a_j}([x_k]_{a_i}) - \underline{a_j}([x_k]_{a_i}) \cap \underline{a_i}([x_k]_{a_j})$ 时, 属性值 x_{ki} 的多属性权值介于 1 和 $1/|[x_k]_{a_i}|$ 之间.

证明. 由上近似集定义可知, $\overline{a_j}([x_k]_{a_i}) = \{x \in U | [x_k]_{a_i} \cap [x_k]_{a_j} \neq \emptyset\}$, 位于集合 $\overline{a_j}([x_k]_{a_i}) - \underline{a_j}([x_k]_{a_i}) \cap \underline{a_i}([x_k]_{a_j})$ 内的数据权值取决于 $[x_k]_{a_j}$ 与 $[x_k]_{a_i}$ 交集的大小, 该值反映了属性值 x_{ki} 和 x_{kj} 的同现次数. 交集元素越多, 表明属性值 x_{ki} 与属性 a_j 越相关, 该属性值在聚类过程中的作用越大, 例如图 1, 用 a_3 度量 x_{42} 的权重, $W_{a_3}(x_{42}) = 3/4 = 0.75$. □

References

1 Wu X H, Wu B, Sun J, Qiu S W, Li X. A hybrid fuzzy K-harmonic means clustering algorithm. *Applied Mathematical Modelling*, 2015, **39**(12): 3398−3409

2 Jiang Yi-Zhang, Deng Zhao-Hong, Wang Jun, Qian Peng-Jiang, Wang Shi-Tong. Collaborative partition multi-view fuzzy clustering algorithm using entropy weighting. *Journal of Software*, 2014, **25**(10): 2293−2311
(蒋亦樟, 邓赵红, 王骏, 钱鹏江, 王士同. 熵加权多视角协同划分模糊聚类算法. 软件学报, 2014, **25**(10): 2293−2311)

3 Bai L, Liang J Y. Cluster validity functions for categorical data: a solution-space perspective. *Data Mining and Knowledge Discovery*, 2015, **29**(6): 1560−1597

4 Bouguettaya A, Yu Q, Liu X, Zhou X, Song A. Efficient agglomerative hierarchical clustering. *Expert Systems with Applications*, 2015, **42**(5): 2785−2797

5 Zhou Chen-Xi, Liang Xun, Qi Jin-Shan. A semi-supervised agglomerative hierarchical clustering method based on dynamically updating constraints. *Acta Automatica Sinica*, 2015, **41**(7): 1253−1263
(周晨曦, 梁循, 齐金山. 基于约束动态更新的半监督层次聚类算法. 自动化学报, 2015, **41**(7): 1253−1263)

6 Jeon Y, Yoon S. Multi-threaded hierarchical clustering by parallel nearest-neighbor chaining. *IEEE Transactions on Parallel and Distributed Systems*, 2015, **26**(9): 2534−2548

7 Kim Y, Shim K, Kim M S, Sup Lee J. DBCURE-MR: an efficient density-based clustering algorithm for large data using MapReduce. *Information Systems*, 2014, **42**: 15−35

8 Mansoori E G. GACH: a grid-based algorithm for hierarchical clustering of high-dimensional data. *Soft Computing*, 2014, **18**(5): 905−922

9 Shang R H, Zhang Z, Jiao L C, Wang W B, Yang S Y. Global discriminative-based nonnegative spectral clustering. *Pattern Recognition*, 2016, **55**: 172−182

10 Sun Ji-Gui, Liu Jie, Zhao Lian-Yu. Clustering algorithms research. *Journal of Software*, 2008, **19**(1): 48−61
(孙吉贵, 刘杰, 赵连宇. 聚类算法研究. 软件学报, 2008, **19**(1): 48−61)

11 Park I K, Choi G S. Rough set approach for clustering categorical data using information-theoretic dependency measure. *Information Systems*, 2015, **48**: 289−295

12 Chen L F, Wang S R, Wang K J, Zhu J P. Soft subspace clustering of categorical data with probabilistic distance. *Pattern Recognition*, 2016, **51**: 322−332

13 Cao F Y, Liang J Y, Bai L, Zhao X W, Dang C Y. A framework for clustering categorical time-evolving data. *IEEE Transactions on Fuzzy Systems*, 2010, **18**(5): 872−882

14 Li M, Deng S B, Wang L, Feng S Z, Fan J P. Hierarchical clustering algorithm for categorical data using a probabilistic rough set model. *Knowledge-Based Systems*, 2014, **65**: 60−71

15 He X, Feng J, Konte B, Mai S T, Plant C. Relevant overlapping subspace clusters on categorical data. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM, 2014. 213−222

16 Ji J C, Pang W, Zheng Y L, Wang Z, Ma Z Q. A novel artificial bee colony based clustering algorithm for categorical data. *PLoS ONE*, 2015, **10**(5): e0127125

17 Gan G, Wu J, Yang Z. PARTCAT: a subspace clustering algorithm for high dimensional categorical data. In: Proceedings of the 2006 IEEE International Joint Conference on Neural Network Proceedings. IEEE, 2006: 4406−4412

18 Bouveyron C, Girard S, Schmid C. High-dimensional data clustering. *Computational Statistics and Data Analysis*, 2007, **52**(1): 502−519

19 Guha S, Rastogi R, Shim K. ROCK: a robust clustering algorithm for categorical attributes. In: Proceedings of the 15th International Conference on Data Engineering. Sydney, Australia: IEEE, 1999. 512−521

20 Barbará D, Li Y, Couto J. COOLCAT: an entropy-based algorithm for categorical clustering. In: Proceedings of the 11th ACM International Conference on Information and Knowledge Management. McLean, VA, USA: ACM, 2002. 582−589

21 Cao L J, Chu K S, Chong W K, Lee H P, Gu Q M. A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. *Neurocomputing*, 2003, **55**(1−2): 321−336

22 Hu Q H, Xie Z X, Yu D R. Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation. *Pattern Recognition*, 2007, **40**(12): 3509−3521

23 Shang R H, Zhang Z, Jiao L C, Liu C Y, Li Y Y. Self-representation based dual-graph regularized feature selection clustering. *Neurocomputing*, 2016, **171**: 1242−1253

24 Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: a review. *ACM SIGKDD Explorations Newsletter*, 2004, **6**(1): 90−105

25 Yip K Y, Cheung D W, Ng M K. HARP: a practical projected clustering algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 2004, **16**(11): 1387−1397

26 Müller E, Günnemann S, Assent I, Seidl T. Evaluating clustering in subspace projections of high dimensional data. *Proceedings of the VLDB Endowment*, 2009, **2**(1): 1270−1281

27 Bouguessa M, Wang S R. Mining projected clusters in high-dimensional spaces. *IEEE Transactions on Knowledge and Data Engineering*, 2009, **21**(4): 507−522

28 Gan G J, Wu J H. Subspace clustering for high dimensional categorical data. *ACM SIGKDD Explorations Newsletter*, 2004, **6**(2): 87−94

29 Zaki M J, Peters M, Assent I, Seidl T. CLICKS: an effective algorithm for mining subspace clusters in categorical datasets. *Data and Knowledge Engineering*, 2007, **60**(1): 51−70

30 Bouguessa M. Clustering categorical data in projected spaces. *Data Mining and Knowledge Discovery*, 2015, **29**(1): 3−38

31 Cesario E, Manco G, Ortale R. Top-down parameter-free clustering of high-dimensional categorical data. *IEEE Transactions on Knowledge and Data Engineering*, 2007, **19**(12): 1607−1624

32 Xiong T K, Wang S R, Mayers A, Monga E. DHCC: divisive hierarchical clustering of categorical data. *Data Mining and Knowledge Discovery*, 2012, **24**(1): 103−135

33 Bai L, Liang J Y, Dang C Y, Cao F Y. A novel attribute weighting algorithm for clustering high-dimensional categorical data. *Pattern Recognition*, 2011, **44**(12): 2843−2861

34 Chan E Y, Ching W K, Ng M K, Huang J Z. An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern Recognition*, 2004, **37**(5): 943−952

35 Jing L P, Ng M K, Huang J Z. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge and Data Engineering*, 2007, **19**(8): 1026−1041

36 Xiong H, Wu J J, Chen J. K-means clustering versus validation measures: a data-distribution perspective. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2009, **39**(2): 318−331

37 Liang J Y, Bai L, Dang C Y, Cao F Y. The k-means-type algorithms versus imbalanced data distributions. *IEEE Transactions on Fuzzy Systems*, 2012, **20**(4): 728−745

38 Qu Kai-She, Zhai Yan-Hui, Liang Ji-Ye, Li De-Yu. Representation and extension of rough set theory based on formal concept analysis. *Journal of Software*, 2007, **18**(9): 2174−2182
(曲开社, 翟岩慧, 梁吉业, 李德玉. 形式概念分析对粗糙集理论的表示及扩展. 软件学报, 2007, **18**(9): 2174−2182)

39 Liu Shao-Hui, Sheng Qiu-Jian, Wu Bin, Shi Zhong-Zhi, Hu Fei. Research on efficient algorithms for rough set methods. *Chinese Journal of Computers*, 2003, **26**(5): 524−529
(刘少辉, 盛秋戬, 吴斌, 史忠植, 胡斐. Rough 集高效算法的研究. 计算机学报, 2003, **26**(5): 524−529)

40 Kim M, Ramakrishna R S. Projected clustering for categorical datasets. *Pattern Recognition Letters*, 2006, **27**(12): 1405−1417

41 Zhang Ji-Fu, Jiang Yi-Yong, Hu Li-Hua, Cai Jiang-Hui, Zhang Su-Lan. A concept lattice based recognition method of celestial spectra outliers. *Acta Automatica Sinica*, 2008, **34**(9): 1060−1066
(张继福, 蒋义勇, 胡立华, 蔡江辉, 张素兰. 基于概念格的天体光谱离群数据识别方法. 自动化学报, 2008, **34**(9): 1060−1066)

42 Zhang J F, Zhao X J, Zhang S L, Yin S, Qin X. Interrelation analysis of celestial spectra data using constrained frequent pattern trees. *Knowledge-Based Systems*, 2013, **41**: 77−88

庞 宁 太原科技大学博士研究生, 副教授. 2007 年获得山西大学计算机与信息技术学院硕士学位. 主要研究方向为数据挖掘, 并行计算.
E-mail: pn529@126.com
(**PANG Ning** Ph. D. candidate and associate professor at Taiyuan University of Science and Technology. She received her master degree from the College of Computer Science and Technology, Shanxi University in 2007. Her research interest covers data mining and parallel computing.)

张继福 太原科技大学计算机科学与技术学院教授. 2005 年获得北京理工大学计算机学院博士学位. 主要研究方向为数据挖掘, 并行与分布式计算, 人工智能. 本文通信作者.
E-mail: jifuzh@sina.com
(**ZHANG Ji-Fu** Professor at the College of Computer Science and Technology, Taiyuan University of Science and Technology. He received his Ph. D. degree from the College of Computer Science, Beijing Institute of Technology in 2005. His research interest covers data mining, parallel and distributed computing, and artificial intelligence. Corresponding author of this paper.)

秦 啸 美国奥本大学计算机科学与软件工程系教授. 2004 年获得美国内布拉斯加州林肯大学计算机学院博士学位. 主要研究方向为并行与分布式系统, 存储系统, 容错和性能评估.
E-mail: qinxiao@gmail.com
(**QIN Xiao** Professor in the Department of Computer Science and Software Engineering, Auburn University, USA. He received his Ph. D. degree in computer science from the University of Nebraska-Lincoln, USA in 2004. His research interest covers parallel and distributed systems, storage systems, fault tolerance, real-time systems, and performance evaluation.)