

一种高维数据集的子空间聚类算法

乐耀佳, 许建华

(南京师范大学 计算机科学与技术学院, 江苏 南京 210097)

[摘要] 提出了一个基于密度和网格的子空间聚类算法. 该算法运用启发式的密度连通思想来确定一维空间初始簇的生成, 使用自底向上的搜索策略来发现存在子空间中的簇. 实验结果表明, 在处理高维数据时, 在不牺牲算法的其他性能的同时提高了聚类的有效性, 降低了对输入数据顺序及噪音数据的敏感性.

[关键词] 聚类算法, 子空间聚类, 基因芯片

[中图分类号] TP 391.4 [文献标识码] A [文章编号] 1672-1292(2009)03-0055-09

A Subspace Clustering Algorithm for High-dimensional Data

Yue Yaojia, Xu Jianhua

(School of Computer Science and Technology, Nanjing Normal University, Nanjing 210097, China)

Abstract A new subspace clustering algorithm based on grid and density is proposed in this paper. The algorithm makes use of heuristic density-connected idea to generate the initial clusters in the first dimension, and applies bottom-up strategy to search the subspace clusters. With the experiments on real-world gene expression data, the results show that our algorithm is effective without sacrificing other performances and reduces the sensitivity to the data order and to the noise data in dealing with high-dimensional data.

Key words clustering algorithm, subspace clustering, gene CMOS chip

随着基因芯片技术的迅速发展, 积累了大量的实验数据, 它们的特点是样本数量少、特征维数高. 聚类算法是常用的分析手段之一, 但是传统的聚类算法遇到了一系列难题, 如距离度量在高维空间中难以衡量两个样本之间的差异. 因此, Rakesh Agrawal 等提出了针对高维数据的子空间聚类算法 (CLIQUE)^[1], 它是在高维数据空间中对传统聚类算法的一种扩展, 其思想是将搜索局部化在相关维中进行, 尝试在相同数据集的不同子空间上发现聚类. 现有的子空间聚类算法根据搜索的方向的不同, 可以分成两大类: 自底向上的搜索策略和自顶向下的搜索策略.

在自底向上的搜索策略中, CLIQUE 算法^[1]是最早尝试在数据子空间中查找聚类的算法. 该算法采用了基于密度和基于网格的聚类算法的思想, 使用 Apriori 策略来查找和合并某个度量大于给定阈值的单元, 产生候选子空间, 并且将这些候选子空间按其规模即子空间中点的数量进行排序; 随后利用最小描述长度 (Minimum Description Length^[2], MDL) 准则将规模较低的子空间剪枝. 该算法能自动标识高维空间的子空间, 所有搜索限制在原始空间的子空间中, 而不是引入新的维度. 随后, ENCLU 算法^[3]和 MAFIA 算法^[4]都在 CLIQUE 算法基础上进行了改进, 大大提高了运行速度. SUBCLU 算法^[5]采用的是基于密度的聚类算法策略, 是在 DBSCAN 算法基础上发展而来的. DOC 算法^[6]是基于网格的算法, 采用了反复改善簇质量的策略. IBUSCA 算法^[7]采用的是基于网格的聚类算法策略, 该算法是在 CLIQUE 和 MAFIA 的基础上发展而来的. 自底向上的算法通过对数据集的全面搜索, 不会失去任何一个簇, 但是大部分该类算法策略很少考虑数据的分布, 通常设定全局密度阈值, 随着维度的增加, 数据集将被打散, 其密度也将随之降低, 从而容易导致聚类质量的降低, 很容易导致重叠的簇产生.

在自顶向下的搜索策略中, 初始将整个数据集划分为 k 个部分, 并赋给每个簇相同的权值, 然后重复采用某种策略对这些初始簇不断改进, 并更新这些簇的权值. 在大数据集中, 这个重复过程所需的代价相

收稿日期: 2008-12-20

通讯联系人: 许建华, 教授, 研究方向: 模式识别、神经网络、机器学习、信号处理等. E-mail: xujianhua@njnu.edu.cn

当高,因此大部分该类算法都采用某些策略选择实际数据的一部分作为数据样本来提高性能. PROCCLUS 算法^[8]和 ORCLUS 算法^[9]采用的策略就是将数据库分成多个子集,将高维空间分成多个子空间,形成子集-子空间对,子集在子空间中的映射形成紧凑的映射类.其中 PROCCLUS 算法^[8]是最早也是最典型的自顶向下算法,该算法选择实际数据的一部分作为数据的样本,然后从样本中选择 k 中心点并反复改进簇的数量,适合查找超球面形状的簇.之后提出的 ORCLUS 算法^[9]较之 PROCCLUS 算法更稳定、更精确. Meta-Cluster 算法^[10]采用的是基于网格和密度的聚类算法策略,该算法采用相似度测量的策略来挖掘存在于重叠网格之间的真实簇.自顶向下的算法为数据的每个部分都建立簇,这意味着不会有重复的簇产生,一个点只能赋给一个簇,但是在大数据集中,不断重复改进初始簇的过程需要的代价相当高,时间开销通常随着数据维度和子空间维度的增加呈指数级增长.

本文提出的高维数据集的子空间聚类算法,首先将每个样本点看作单独的簇,扫描一遍数据集一次性地构造出数据集各维特征上的所有合理的聚类,然后在自底向上的簇合并过程中生成所有合理的聚类,最后用 3 个数据来验证算法的有效性与效果.

1 算法描述

1.1 基本概念

为方便后续的讨论,对有关术语作一些说明.

定义 1 对标准化数据集,给定一个阈值 $0 < \alpha < 1$ 若 $|x_j - y_j| \leq \alpha$ ($1 \leq j \leq d$),则称点 $x = [x_1, x_2, \dots, x_i, \dots, x_d]$ 和 $y = [y_1, y_2, \dots, y_i, \dots, y_d]$ 是关于第 j 维相似.

本文为简化查找相似点的过程,首先将样本点按照每个属性的属性值大小进行排序(每个维度 j 有一个排序的序列 S_j).根据定义 1 通过顺序扫描 S_j 可以得到所有与样本点 x 第 j 维相似的点 y .

定义 2 给定一个三元组 (x_i, x_j, f) 来存储样本点在每一维上的近邻关系,其中 $(x_i, x_j, 1)$ 表示样本点 x_i 和样本点 x_j 在当前指定空间维度中是相似的, $(x_i, x_j, 0)$ 表示样本点 x_i 和样本点 x_j 在当前指定空间维度中是不相似的.

引理 1^[1] (单调性原理) 如果一个样本点集 S 是 k 维空间的一个密集簇,那么将 S 映射到 $k-1$ 维空间得到 S' ,则 S' 将是 $k-1$ 维空间某个密集簇的子集.

1.2 算法描述

1.2.1 数据标准化与排序

由于芯片原始数据集的表达多样性,首先标准化原始数据集.数据标准化处理以数据的最大值和最小值的差距进行数学计算,其结果介于 $0 \sim 1$ 之间.具体计算公式为: $z_i = \frac{x_i - x_i^{\min}}{x_i^{\max} - x_i^{\min}}$.式中, x_i 为原始样本值, z_i 为标准化处理后的样本值, x_i^{\min} 为样本集中某个属性特征的最小值, x_i^{\max} 为样本集中某个属性特征的最大值.

经过上述标准化处理,原始数据均转化到同一个数量级别上,消除了各个属性分量之间数值范围大小对计算的影响.下一步,对每个属性下的样本值排序,便于后面进一步的聚类工作.

1.2.2 初始簇的生成

将每个样本点看作单独的簇,然后在自底向上的簇合并过程中生成所有合理的初始聚类.在一维空间中,本文采用密度连通的策略,通过扫描一次样本一次性构造出初始簇.一维空间上初始簇的形成采用密度连通的思想:定义的簇为密度相连的点的最大集合,是将密度连通的区域划分为簇,可以完全避免对输入数据顺序的依赖性,并可以有效降低噪声数据的干扰,利于发现任意形状的聚类.本文提出用一个三元组 (x_i, x_j, f) 来存储样本点在每一维上的近邻关系,来降低查找相似点的复杂性.关于密度阈值如果限定唯一的值,那么对各个属性的评判是不公平的,所以本文根据每个属性空间下样本的分布方式确定相对应属性空间的密度阈值,由此得到自适应网格.与等间距网格划分方法相比,自适应网格划分方法避免了对密集簇边界的破坏.

算法 1 生成初始簇的算法描述.

输入: 已标准化数据集

输出: 一维初始簇集

1) for $k = 1$ to d do

依据第 k 维属性下的样本值对样本排序; 计算第 k 维属性下三元组矩阵 S

2) for $k = 1$ to d do

for $x \in S_j$ do

如果 (x_i, x_j, f) 中的 $f = 1$, 其中 $i \neq j, x_i \in$ 一维初始簇 C , 那么 x_i 和 x_j 在属性 k 下是相似的, 一维初始簇 $C \leftarrow C \cup \{x_j\}$;

3) 算法结束, 得到所有一维初始簇。

其中关于三元组 (x_i, x_j, f) , 图 1 举例描述了一个包含 5 个样本的数据集在某个属性下的三元组 (x_i, x_j, f) 。从图 1 很显然看出有 2 个簇的存在, 簇 1 = {1, 2, 3}, 簇 2 = {4, 5}。如果遇到样本 x_k 与样本 x_i 和 x_j 都比较靠近的情况, 就按照先到的原则, 把 x_k 和 x_i 划为一类, 这对实验的结果可能有一定的影响。

	1	2	3	4	5
1	-	1	1	0	0
2		-	1	0	0
3			-	0	0
4				-	1
5					-

图 1 近邻三元组

Fig 1 Example of neighbor trip les

1. 2. 3 簇集合并

当一维初始簇完全生成后, 从一维空间到二维空间将产生数量庞大的候选簇, 所以在一维到二维密集簇的聚类过程中, 本文通过类别标签的策略首先来查找二维密集簇可能存在的子空间, 然后通过对相关度较高的属性之间的一维密集簇进行搜索, 合并得到二维密集簇。对二维空间中的密集子空间的寻找, 通过计算基于二维联合熵的函数来确定。

在计算二维联合熵时, 参考文献 [3] 中计算一维属性熵的方法, 首先将每个属性划分为 $1/\Delta$ 个等长的区间, 设 X 为所有区间, 密度 $d(x)$ 定义为落入 x 内的样本点个数占数据集样本总数的百分比, 属性 A 的熵值计算公式为: $H(A) = - \sum_{x \in X} d(x) \log d(x)$ 。间隔 Δ 的大小需要小心选择, 如果选取过小, 将划分出很多的区间, 导致落入区间里的样本点过少; 相反, 如果选取的过大, 则很难得到有效的信息, 区分出区间的密度变化; 本文采用文献 [3] 所建议的取值 0. 1。

由此推广定义 $H(X, Y)$ 为二维属性联合熵函数。其中, 两个属性在二维空间中的联合分布可以采用类矩形网格单元中的样本分布来计算, 二维空间上的密度同样定义为落入单位网格内的样本点个数占数据集样本总数的百分比。若数据集在属性 X 上有 m 个簇, 在属性 Y 上有 n 个簇, 如果由属性 X 和属性 Y 组合的二维密集簇的个数比一维簇的个数增加较多, 则说明新属性的加入打散了原本的一维簇, 则属性 X 和属性 Y 相关度较低, 二维联合熵值会由于两个属性在分布上缺乏关联性而使熵值较大; 反之, 若二维密集簇的个数较之一维簇的个数变化不大, 则说明属性 X 和属性 Y 相关度较高, 其熵值较低。

本文的类别标签定义为两个属性之间是否有较高的相关性, 若相关性较高, 则认为这两个属性共同存在的子空间中存在密集簇的可能性较大。通过二维联合熵的计算, 对于熵值小于阈值 τ (本文取 0. 4) 的两个属性 X 和 Y , 作为二维候选子空间组合 (X, Y) , 并将对应的类别标签 $c(X, Y)$ 置为 1, 认为在这个候选空间中存在密集簇的可能性较大。因此, 在二维密集簇的搜索中不需要传统的自底向上的搜索方式, 仅仅对类别标签为 1 的两两属性之间的一维密集簇进行搜索, 合并得到二维密集簇。

在 k 维 $\rightarrow k+1$ 维簇的合并过程中, 把在 k 维上相似的样本点划分到 $k+1$ 维上的同一个集合, 直至没有新的划分产生, 完成簇的合并。如果 k 维上不再有满足条件的簇进行合并, 也就是说 k 维上没有任何 2 个簇拥有 $k-1$ 个相同的属性, 并且不同的 2 个属性有相同的类别标记时, 这个时候就不再产生新的划分了。

簇 C_1 和簇 C_2 均属于二维簇: $\text{card}(C_1) = \text{card}(C_2) = 2$

并且, 它们拥有相同的一个特征维 X_x : $\text{card}(\{c \mid c \in (C_1 \cap C_2)\}) = 1$

则将簇 C_1 和簇 C_2 合并为三维空间上的一个新簇: $C_3 = C_1 \cup C_2$ 。

从图 2 可以看出两个二维密集簇合并生成一个新的三维簇的过程。图中的空心点代表三维空间的样本点 (实心点) 在二维空间的两个密集簇 C_1 和 C_2 上的投影。当二维密集簇 C_1 和 C_2 在相同属性 X_2 下拥有共同的投影区域, 且满足相当数目的样本点时, 则能生成一个新的三维密集簇 C_3 。

本文的算法将类别标签策略与 CLIQUE 的单调性机制 (引理 1) 相结合, 对候选簇进行“剪枝”操作。

假设已经生成了 k 维子空间的所有密集簇, 对其搜索时, 首先两两比较 k 维密集簇所包含的属性位, 判断是否只有 1 维不同; 然后判断这两个不同的维类别标签 c ; 如果 c 为 1 的话, 则将这两个簇作为候选簇集. 两个 k 维簇, 若有 $k-1$ 个属性是相同的, 不同的只有 1 个, 那么合并时, 共同的 $k-1$ 个属性加上它们各自不同的那个属性, 就是 $k-1+1+1$ 到了 $k+1$ 维了; 记录下它们中相同的样本点, 生成 $k+1$ 维的新簇. 根据引理 1 如果这样的一个 k 维单元是密集的, 那么它在 $k-1$ 维空间上的投影也是密集的 (空间闭合性); 反过来, 如果给定的 $k-1$ 维的单元不密集, 则其任意的 k 维空间也是不密集的, 由此可以保证我们得到的高一维的簇也是密集有效的. 如此反复密集簇的合并过程直到没有新的更高维的密集簇产生为止.

- 算法 2 高维簇合并的算法描述:
- 输入: k 维簇集
- 输出: $k+1$ 维簇集
- 1) 如果 $k < d$ 且 k 维簇集中存在 2 个未作比较的簇;
 - 2) 选择 2 个彼此未作比较的簇;
 - 3) 如果这 2 个簇的 $k-1$ 维属性完全相同, 只有 1 维属性不同, 且这 2 个不同的 1 维属性的类别标签 c 为 1;
 - 4) 那么合并这 2 个 k 维密集簇, 保留 $k-1$ 维相同的属性, 加上 2 个不同的属性, 就得到了 1 个新的 $k+1$ 维密集簇, 并写入 $k+1$ 维簇集;
 - 5) 否则转向 (2);
 - 6) $k \leftarrow k+1$ 转向 (1);
 - 7) 算法结束, 得到所有子空间密集簇.

根据单调性原理, 它可以保证所有的密集簇都会被搜索出来, 不管它存在于几维的子空间. 本文的算法并不像 CLIQUE 等以前的算法一样, 初始在整个数据集上划定统一网格, 所以在簇集合并的过程中可以形成样本交叉的聚类结果, 使最后的聚类结果得到更好的解释.

2 实验结果与分析

实验采用 3 个真实数据和 1 组人工合成数据集来测试算法的有效性. 3 个真实数据集都来自 UCI 机器学习数据库. 第一个数据集是威斯康星州乳腺癌数据集 (Wisconsin Breast Cancer Database)^[11], 第二个数据集是鸢尾科植物数据集 (Iris Plants Database)^[12], 第三个数据集是酵母细胞的基因表达数据集 (Yeast Gene Database)^[13]. 实验环境: Windows 系统, 1.83 G 的主频, 2.5 G 的内存, VC2005

2.1 实验数据

WBCD 数据集包含了 699 个样本、9 个属性特征和 1 个类别标签 (“良性的”和“恶性的”).

Iris 数据集包含了 3 种鸢尾科植物: 山鸢尾 *Setosa* 变色鸢尾 *Versicolour* 和维吉尼亚鸢尾 *Virginica* 的 150 个样本数据, 每一种植物有 50 个样本, 每个植物样本都有 4 个属性: 萼片长度、萼片宽度、花瓣长度、花瓣宽度.

基因表达数据通常来自于 DNA 芯片或者其他的微阵列技术. 本文采用的酵母细胞的基因芯片数据集包含了 17 个条件下的 2884 个酵母基因的数据. 每个数据表示一个基因的信使 RNA 在某一个特定条件下的相对值, 这个值是对基因芯片原始数据求对数得到的.

人工合成数据集是由参考文献 [14] 的合成方法编写的数据生成器随机生成的, 可以通过输入参数来获得不同的数据集, 参数包括数据集集中的样本数目、样本的取值范围、属性维数、子空间簇的个数、噪音数据的比例以及包含簇的子空间的维度.

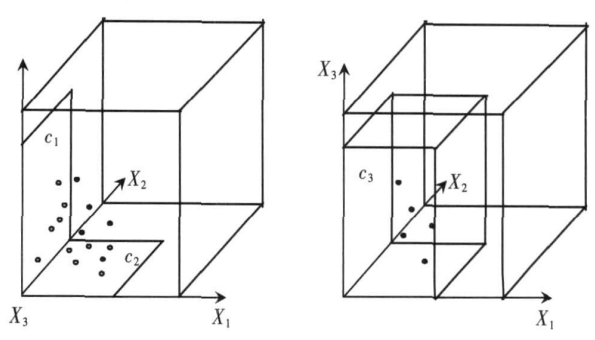


图 2 将两个 2 维相似簇合并到一个新的 3 维簇

Fig.2 Merger two 2-dimensional dense units into a new 3-dimensional dense unit

2.2 实验结果

2.2.1 WBCD 数据集

在对 WBCD 数据集分析时,本文排除了类别标签属性,所以实际分析的数据集只有 9 个属性,其中有 16 个样本没有标注,所以实际分析的样本只有 683 个.本文的方法成功地聚类出 32 个簇,并且所在最高的空间维度是 7 维,从表 1 中可以看到其中一部分实验分析结果.簇 1 和簇 2 分别包含了 100 和 104 个良性患者而没有一个恶性患者,簇 23 将恶性患者聚在一类而不含有良性患者. IBUSCA^[7] 算法在 WBCD 数据集上发现了较好的 3 个簇,如表 2 所示.

表 1 对 WBCD 数据集的聚类结果

Table 1 Clusters discovered in WBCD dataset with our algorithm

簇	子空间 维度	良性 样本数	恶性 样本数	样本 总数
# 1	7	100	0	100
# 2	7	104	0	104
# 23	2	0	128	128

与 IBUSCA^[7] 算法相比较,虽然本文的方法没有 BUSCA^[7] 算法发现的第一个簇属性空间高,但在错分性上,本文的方法都没有错分情况发生,这一点可以从图 3 中看出.图 3(a)显示的是一个 7 维空间中的簇 1,这个簇完全由 100 个良性患者组成,没有一个恶性患者.图 3(b)中显示出了另一个 7 维空间中的簇 2 同样这个簇也完全由良性患者组成.图 3(c)中显示的是存在于 2 维空间中的簇 23 其中包含 128 个恶性患者.

2.2.2 Iris 数据集

在一维空间中一共聚出 19 个簇,分属在 4 个属性特征下,如图 4 所示(因为单一的一维数据显示出来是点,不便于观察,所以在不影响聚类结果的情况下,添加了一维全为 0 值的属性,使得其便于显示与观察).在萼片长度属性下有 6 个簇,其中聚类效果较好的是簇 1、簇 2 和簇 3.簇 1 包含 38 个山鸢尾样本、3 个变色鸢尾样本(3 个错分样本);簇 2 包含 37 个变色鸢尾样本、1 个维吉尼亚鸢尾样本(1 个错分样本);簇 3 包含 35 个维吉尼亚鸢尾样本.在花萼宽度属性下聚类效果较差,一共聚出 4 个簇,簇 7 包含 30 个山鸢尾样本、2 个变色鸢尾样本(2 个错分样本);而簇 8 包含 95 个样本,将 3 类植物样本混淆在一起,效果非常差,这是由于 3 类植物的花萼宽度属性特征值非常相似导致的结果,这一点可以从图中看出.与之相比,在花瓣长度属性和花瓣宽度属性下的聚类效果较好,尤其是花瓣宽度属性.簇 11 和簇 16 都成功地将 50 个山鸢尾样本聚在一起,没有一个错分样本.簇 12 和簇 17 分别聚出 43 个和 42 个变色鸢尾样本,同样没有一个错分样本.效果稍差的簇 13 包含 7 个变色鸢尾样本;簇 14 和簇 15 各包含 14 个和 36 个样本,将 50 个维吉尼亚鸢尾样本分别聚在了两个簇中.簇 18 很好地将 50 个维吉尼亚鸢尾样本聚在一起,没有一个错分样本.在花瓣宽度属性下的聚类结果唯一不满意的是簇 19,它将变色鸢尾样本中 8 个表达较高的样本聚在了一起.

图 5 显示的是在二维空间的有效聚类结果.从(a)可以看出在萼片长度属性和花瓣长度属性组成的二维空间中有 3 个有效簇,簇 21、簇 22 和簇 23 分别包含 37 个山鸢尾样本、36 个变色鸢尾样本、32 个维吉尼亚鸢尾样本.从(b)可以看出在萼片长度属性和花瓣宽度属性组成的二维空间中有 3 个有效簇,簇 24、簇 25 和簇 26 分别包含 38 个山鸢尾样本、35 个变色鸢尾样本、35 个维吉尼亚鸢尾样本.由于 3 类植物在萼片长度属性下非常相似,所以聚类结果不佳.从(c)可以看出在花瓣长度属性和花瓣宽度属性组成的二维空间中有 3 个有效簇,簇 27、簇 28 和簇 29 分别包含 50 个山鸢尾样本、41 个变色鸢尾样本、36 个维吉尼亚鸢尾样本.

从图 6 中可以看到三维空间中的 3 个有效聚类结果:簇 31、簇 32 和簇 33.簇 31 聚类得到 37 个山鸢尾

表 2 IBUSCA 算法对 WBCD 数据集的聚类结果^[7]

Table 2 Clusters discovered in WBCD dataset with IBUSCA algorithm

簇	子空间 维度	良性 样本数	恶性 样本数	样本 总数
# 1	9	187	0	187
# 2	6	108	0	108
# 14	2	3	125	128

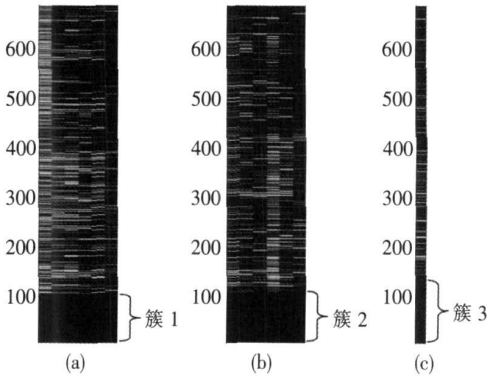


图 3 WBCD 数据集的聚类结果图
Fig.3 Results on WBCD dataset

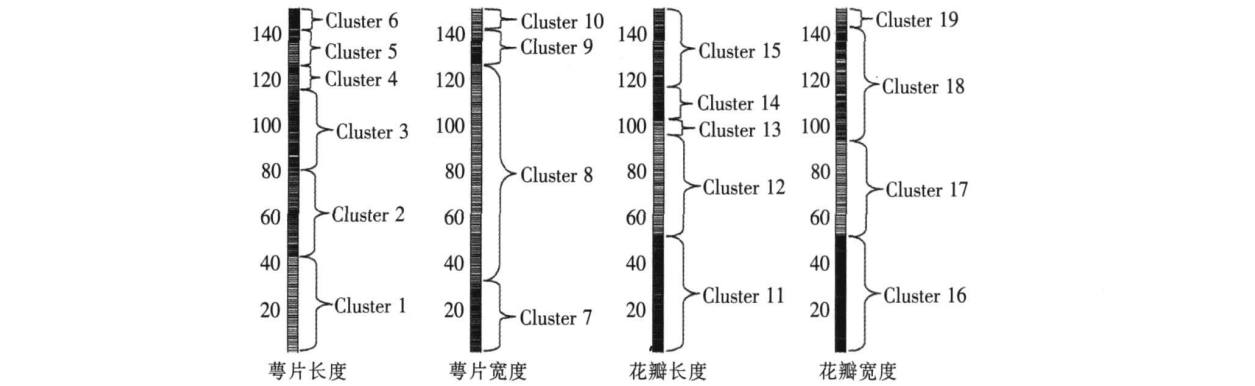


图 4 Iris 数据集的一维空间聚类结果

Fig.4 Results on the 1-dimensional subspace of Iris dataset

样本; 簇 31 聚类得到 35 个变色鸢尾样本; 簇 32 聚类得到 32 个维吉尼亚鸢尾样本. * 1、* 2 和 * 3 三个部分由于萼片长度属性表达差异较小的影响 (这点可以从图 6 左侧区域看出, 它们的颜色非常接近), 分别将 13 个山鸢尾样本、15 个变色鸢尾样本和 18 个维吉尼亚鸢尾样本聚在了一起, 造成样本错分, 影响到聚类效果.

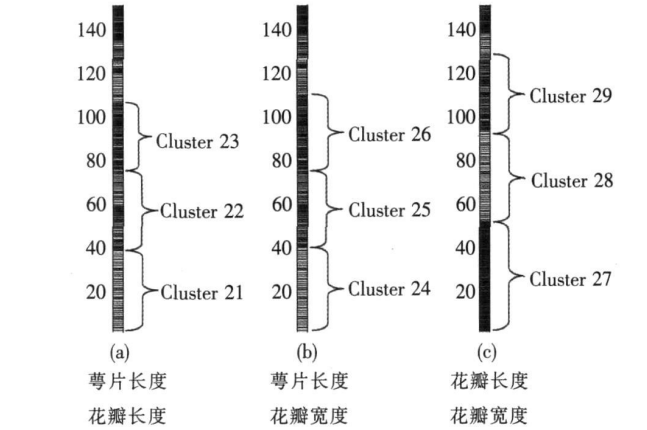


图 5 Iris 数据集的二维空间聚类结果

Fig.5 Results on the 2-dimensional subspace of Iris dataset

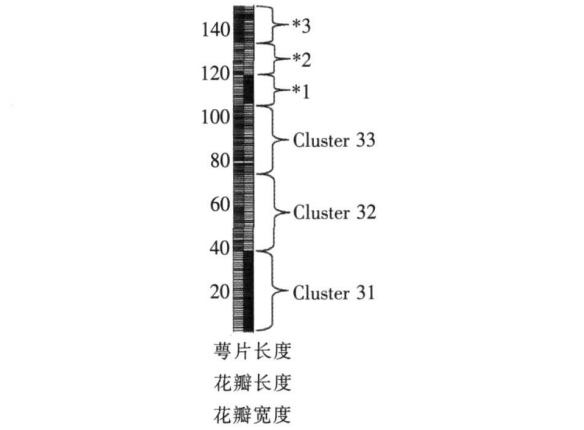


图 6 Iris 数据集的三维空间有效聚类结果

Fig.6 Results on the 3-dimensional subspace of Iris dataset

2.2.3 Yeast gene 数据集

酵母基因数据集共有 17 个属性、2 884 个样本, 原始数据如图 8 (a) 所示. 实验结果显示在一维空间中共聚出 497 个簇, 图 7 中从左至右依次显示的是分别在 17 个属性特征下的聚类结果, 由于簇的数量较多, 没有在图中一一标识出来.

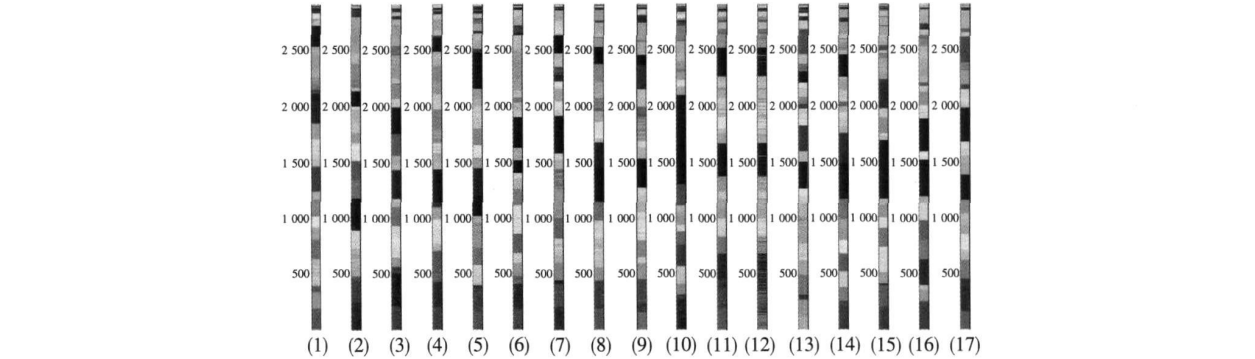


图 7 Yeast gene 数据集的一维空间聚类结果

Fig.7 Results on the 1-dimensional subspace of Yeast gene dataset

图 8 (b) 显示的是在 6 维空间中得到的一个有效簇, 这也是本算法在酵母基因数据集中挖掘出的最高

维的簇,含有 104个样本.在 (b)中的右上角,可以明显看到一个簇.在图 8(c)中较清楚的显示了这一点:在最右侧的 6个属性特征 (10 11, 14 16 17, 15)所指示的区域,有一个 104* 6的矩形区域.

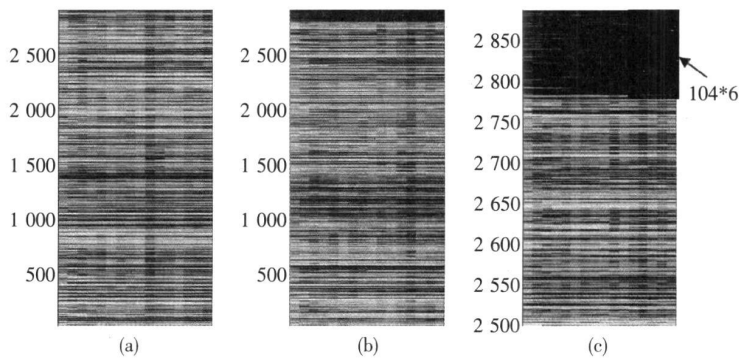


图 8 对 Yeast gene 数据集的聚类结果图示
Fig.8 Results on Yeast gene dataset

本文将本算法的聚类结果与文献 [15]的结果做了比较.文献 [15]将基因 NHP1Q DPB4 IES3和 TAF9 聚在一类,这 4个基因都与染色体重组与维持相关,本文在三维子空间的一个簇中得到了同样的结果.文献 [15]发现的另一个簇包含基因 CDC33 TEF4 EFB1和 NHP2,这 4个基因编码核糖体结构蛋白.在本文的聚类结果中基因 TEF4与前 3个基因却被分别聚在了四维子空间的两个不同的簇中,效果不如文献 [15]的方法.

2. 2. 4 噪音分析

噪音分析分为两个部分.第一部分,本文在真实的 Iris数据集中加入了 3个野值数据,并进行实验.从图 9可以看出,得到的结果与在原 Iris数据集实验的结果完全相同 (图中色块的颜色差别,是由于噪音数据的加入),野值数据完全被排除在有效簇外.这是由于本文提出的算法在生成初始簇时采用了密度连通的策略,定义的簇为密度相连的点的最大集合,由此对野值数据的敏感性较弱,仍然可以得到较好的聚类效果.

第二部分,本文在含有高斯噪音的人工合成数据集上进行实验,该数据集共有 10维特征属性、1 000 个样本,噪音点比例为 10%,并在 3维空间和 5维空间中分别存在一个簇.本文在该数据集上分别运行 CLIQUE 算法和本文算法,结果如表 3所示.本文算法完整地挖掘出包含两个簇的子空间,而 CLIQUE算法却漏掉了第 2个簇中的一个属性,因此,本文算法在噪音数据集上的聚类性能优于 CLIQUE算法.

2. 2. 5 算法效率分析

为了测试本文提出的算法的实际效率,本文在一组人工合成数据集上分别测试了算法的运行时间与样本数、数据集属性维数以及包含簇的子空间维数之间的关系,并给出了本文算法和 CLIQUE 算法的比较.测试算法运行时间与样本数目的关系是在一组 10维属性的数据集上运行的,其中在 3维子空间有一个簇,样本数目为 500~ 3 000.实验结果如图 10中所示.随着数据集样本个数的增加,两种算法的运行时间均呈类线性增加.

测试算法时间与数据集属性维数的关系实验是在一组样本总数为 1 000的数据集上运行的,其中在 3维子空间有一个簇,属性维数为 5维 ~ 40维,实验结果如图 11所示.开始两种算法的时间开销相

表 3 噪音数据集搜索簇结果对比

Table 3 Comparison of quality of results obtained by CLIQUE and OUR Algorithm

簇所在子空间	
输入数据集	{ 1 7 8 } { 2 3 5, 6, 9 }
CLIQUE	{ 1 7 8 } { 2 3 6, 9 }
OUR	{ 1 7 8 } { 2 3 5, 6, 9 }

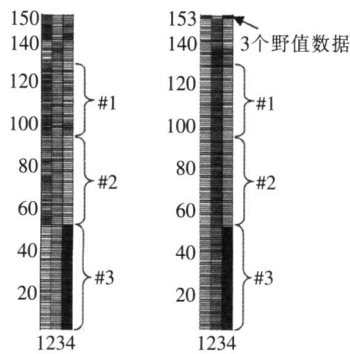


图 9 原 Iris 数据集与含野值数据的 Iris 数据集的聚类结果对比图

Fig.9 Comparison of quality of results obtained on Iris dataset

近,随着数据集属性维数的增加,到达 25 维以后,与 CLIQUE 算法相比较,本文算法的运行时间呈线性增长,这是由于计算一维三元组和二维联合熵的时间开销随着样本数的增加均呈线性增加的原因。

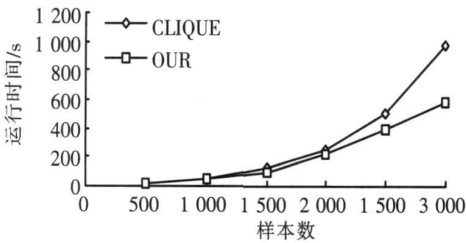


图 10 样本数目的可伸缩性
Fig.10 Scalability with datasets size

为了测试本文提出的类别标签策略对子空间簇的寻找能力的影响,本文还设计了算法运行时间与有效簇所在子空间的最高维度的关系实验,实验所用数据集为 1 000 个样本,20 维属性,有效簇所在子空间最高维度分别为 4 6 8 10 12 实验结果如图 12 所示.随着存在有效簇的最高子空间维度的增加,本文算法的运行时间增幅不大,由此可以看出类别标签策略在一定程度上降低了搜索空间,降低了时间开销。

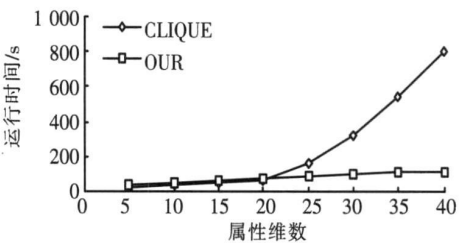


图 11 属性维数的可伸缩性
Fig.11 Scalability with attribute dimensionality

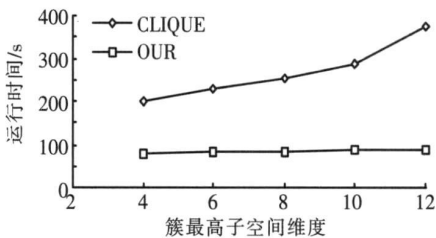


图 12 有效簇最高子空间维度的可伸缩性
Fig.12 Scalability with cluster dimensionality

3 结论

本文提出的聚类算法能有效地发掘有样本交叉的簇,即一个样本既能够在某些特征下归于一个簇,又能够在另外的特征下归于另一个簇,这能够有效地分析所给样本的特性。

实验结果表明,本文提出的子空间聚类算法在处理高维数据时,能有效地挖掘出研究数据集上的有效簇,在不牺牲算法的其他性能的同时提高了聚类的精度,有效地降低了对输入数据顺序及噪音数据的敏感性,并能形成任意形状的簇。

[参考文献] (References)

[1] Agrawal R, Gehrke J, Gunopulos D. Automatic subspace clustering of high dimensional data for data mining applications [C] // Proceedings of the 1998 ACM-SIGMOD International Conference on Management of Data Seattle Washington: ACM Press, 1998, 6: 94-105.

[2] Gunwald P D. Model selection based on minimum description length[J]. Journal of Mathematical Psychology, 2000, 44: 133-152

[3] Cheng C H, Fu A W-C, Zhang Y. Entropy-based subspace clustering for mining numerical data[C] // Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining San Diego, USA: ACM Press, 1999: 84-93.

[4] Nagesh H, Gail S, Choudhary A. Adaptive grids for clustering massive data sets[C] // Proceedings of SIAM International Conference on Data Mining SIAM, 2001: 477-493

[5] Kailing K, Kriegel H-P, Köger P. Density-connected subspace clustering for high-dimensional data[C] // Proceedings of the 4th SIAM International Conference on Data Mining Lake Buena Vista, FL, 2004: 246-257.

[6] Procopiu C M, Jones M, Agarwal P K, et al. A monte carlo algorithm for fast projective clustering[C] // Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data Madison: ACM Press, 2002: 418-427.

[7] Golub M, Urszula M-K. IBUSCA: a grid-based bottom-up subspace clustering algorithm[C] // Proceedings of the 6th International Conference on Intelligent Systems Design and Applications USA: IEEE Computer Society, 2006: 671-676.

[8] Aggarwal C C, Procopiu C, Wolf J L, et al. Fast algorithms for projected clustering[C] // Proceedings of the 1999 ACM SIG-

- MOD International Conference on Management of Data New York: ACM Press, 1999: 61-72
- [9] Aggarwal C C, PS Yu Finding generalized projected clusters in high dimensional spaces[C] // Proceedings of the 2000 ACM SIGMOD international conference on Management of data Oalaks, Texas: ACM Press, 2000: 70-81
- [10] Liu J, Strohmaier K, Wang W. Revealing true subspace clusters in high dimensions[C] // Proceedings of the 4th IEEE International Conference on Data Mining USA: IEEE Computer Society, 2004: 463-466
- [11] UCIMachine Learning Repository[EB/OL]. <http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/>.
- [12] UCIMachine Learning Repository[EB/OL]. <http://archive.ics.uci.edu/ml/machine-learning-databases/iris/>.
- [13] UCIMachine Learning Repository[EB/OL]. <http://archive.ics.uci.edu/ml/machine-learning-databases/yeast/>.
- [14] He J, Lan M, Tan C L, et al Initialization of cluster refinement algorithms: a review and comparative study[C] // Proceedings of IEEE International Joint Conference on Neural Networks USA: IEEE Computer Society, 2004: 297-302
- [15] Böhm C, Kailing K, Kriegel H P, et al Density connected clustering with local subspace preferences[C] // Proceedings of the 4th IEEE International Conference on Data Mining USA: IEEE Computer Society, 2004: 27-34

[责任编辑: 严海琳]

(上接第 17 页)

[参考文献] (References)

- [1] 李超, 陈玲. 广告牌风力卸载的卸载方法及抗强风广告牌: 中国, ZL2006 1 0086005. 6[P]. 2008-05-28
Li Chao, Chen Ling. The method of wind billboard uninstall and billboard anti-strong wind. China, ZL2006 1 0086005. 6[P]. 2008-05-28 (in Chinese)
- [2] 龙安国. 基于 AT89S51 的 PCB 制版喷雾腐蚀控制系统的设计[J]. 电子技术, 2008(5): 48-51.
Long Anguo. Design of PCB plate-making spray erosion control system based on AT89S51[J]. Electronic Technology, 2008(5): 48-51 (in Chinese)
- [3] 张迎新. 单片机原理与应用[M]. 北京: 电子工业出版社, 2005.
Zhang Yingxin. Principle and Application of SCM[M]. Beijing: Publishing House of Electronics Industry, 2005. (in Chinese)
- [4] ATMEL AT89S52 中文资料[EB/OL]. [2008-05-01]. <http://www.51.com>. 2008
ATMEL AT89S52 Chinese datasheet[EB/OL]. [2008-05-01]. <http://www.51.com>. 2008 (in Chinese)
- [5] 杨志忠, 卫桦林. 数字电子技术[M]. 北京: 高等教育出版社, 2004
Yang Zhizhong, Wei Hualin. Digital Electronic Technology[M]. Beijing: Higher Education Press, 2004 (in Chinese)
- [6] 田立, 田清, 戴方震. C 语言程序设计快速入门[M]. 北京: 人民邮电出版社, 2007.
Tian Li, Tian Qing, Dai Fangzhen. C Language Programming Design Quick Start[M]. Beijing: Posts and Telecom Press, 2007. (in Chinese)
- [7] 刘国汉, 韩根亮, 张建华. 嵌入式数据采集系统的研究[J]. 甘肃科技, 2004(10): 91-92
Liu Guohan, Han Genliang, Zhang Jianhua. Research of the embedded data collecting system[J]. Gansu Science and Technology, 2004(10): 91-92 (in Chinese)

[责任编辑: 刘 健]