文章编号:1672-3961(2011)05-0037-09

基于信息熵的高维分类型数据子空间聚类算法

孙浩军,杜育林,姜大志
(汕头大学计算机系,广东 汕头 515063)

摘要:由于分类型数据相异度度量的局限性以及分类型数据在高维空间中的稀疏性,使得传统的相异度度量在高维分类型数据聚类中失效,针对上述问题,本研究提出了一个基于信息熵的理论高维分类型数据聚类算法。该算法综合考虑对应子空间和噪声空间的维度信息熵设计了一个高效、无监督的子空间搜索对高维数据进行有效降维,同时提出了基于整体数据的平均信息熵的全局优化方法对聚类结果进行迭代寻优。通过用人工数据和 Votes、Mushroom 和 Soybean 3 个典型的真实分类数据集试验,与其他分类型聚类算法相比,新算法在聚类准确性、熵值、CU(category utility)以及类个数等指标上有明显提高。

关键词:分类型数据;信息熵;子空间聚类;高维数据
中图分类号:TP301 **文献标志码**:A

ESCHCD: entropy-based algorithm for subspace clustering with high dimensional categorical datasets

SUN Hao-jun, DU Yu-lin, JIANG Da-zhi
(Department of Computer Science, Shantou University, Shantou 515063, China)

Abstract: In high dimensional categorical data datasets, the lack of exact measurement of similarity between data and the distributions of the data are usually sparse. This makes most of those traditional clustering algorithms which work well on low-dimensional data invalid for high-dimensional categorical data datasets. Focusing on these problems, a new high dimensional categorical clustering algorithm was proposed, called ESCHCD (entropy-based algorithm for subspace clustering with high dimensions categorical datasets). An effective and unsupervised objective function was designed to determine the subspace associated with each cluster by considering the entropies of the matched subspace and the noise subspace. At the same time, an average entropy-based global optimization method was also proposed to find the best clustering results. By comparing with other categorical clustering algorithms, the results demonstrated the advantage of the new algorithm on efficiency, entropy measure, category utility(CU) and the number of cluster on synthetic data sets and real data sets, such as Votes、Mushroom and Soybean.

Key words: categorical datasets; entropy; subspace clustering; high dimensional data

0 引言

将物理或抽象的对象的集合分成相似的对象的类的过程称为聚类分析,又称为数据分割[1]。聚类分析已经广泛地用于许多运用领域,包括市场研究、医学、金融等方面。对于数值型数据聚类问题,现在已经有很多优秀的聚类算法,如 k-means[2-3]、PROCLUS[4]、CLIQUE[5]等。但是对于分类型数据聚类问题,由于人们关注的时间较晚,基于分类型数据聚类算法的提出相对较少。然而,分类型数据聚类在实际领域中有着重要的应用价值[6-8],如在社会学、统计学以及心理学等领域中。与数值型聚类相比,分类型数据聚类的难度更大。分类型数据的属性值

收稿日期:2011-04-15
基金项目:广东省自然科学基金资助项目(8151503101000016)
作者简介:孙浩军(1963 -),男,河北衡水人,教授,博士,主要研究方向为模式识别与数据挖掘等. E-mail:haojunsun@stu.edu.cn

是离散的[9],仅仅是一个表示符号,不同数据相应的属性值之间没有大小之分,只有相同或不相同之分,因此,很多传统的距离度量公式,如欧氏距离、Manhattan 距离[4]等,都不能很好地描述数据间的相似程度,因而基于传统距离下的聚类算法并不适用于分类型数据聚类。于是为了度量分类型数据之间的差异,分类型数据聚类通常采用"相异度"作为分类型数据之间的距离度量,如 Jaccard/Dice 系数、汉明距离等[10]。但另一方面,因为分类型数据属性的值域(相对于数值型数据来说)十分狭小,所以即使低维的分类型数据聚类也会经常出现"维数灾"问题[5,11-12](即在全空间上数据间的差异度趋于一个相同的值),在高维数据中就更加突出。

针对以上问题,本研究提出了一种基于信息熵的高维分类型数据子空间聚类算法(entropy-based algorithm for subspace clustering with high dimensions categorical datasets, ESCHCD)来处理高维分类型数据聚类问题。算法中设计了新的子空间识别方法,可以快速、有效、无监督的查找到各类对应的子空间,同时也设计了全局的优化方法进行迭代得到最优的聚类结果。为验证算法的性能,在人工数据和真实数据集上与 k-modes[13]、DHCC[14]、SUBCAD[15]等处理分类数据的算法进行对比实验。实验结果表明,该算法在准确度、信息熵和 CU(category utility)[16]值上均优于其他算法。

1 相关概念

1.1 信息熵

熵,最初作为一个热力学概念,用来度量热量的退化或进化[17]。熵是系统无序的度量,熵越大,无序性越高[18]。香农在信息论中引入熵的概念,用来度量信息系统结构的不确定性。COOCAT[19]、PCC[20]等算法都用信息熵作为聚类优化的度量。ESCHCD 算法也基于信息熵理论的基本原理:越是有序排列的数据(如有聚类特征的数据),熵越小;越是无序的、混沌的数据,熵越大。

设一个信源 X,其概率空间为

$$\begin{bmatrix} X \\ p(x) \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \cdots & x_q \\ p(x_1) & p(x_2) & \cdots & p(x_q) \end{bmatrix},$$

$$\sum_{i=1}^{q} p(x_i) = 1。$$

其信息熵计算公式为

$$E(X) = -\sum_{i=1}^{q} p(x_i)\log(p(x_i)), \qquad (1)$$

从直观上来看,若上述概率空间中 x_i 的概率 $p(x_i)=1$,而 $p(x_1)=p(x_{i-1})=p(x_{i+1})=\cdots=p(x_q)=0$,则每次试验结果 x_i 一定出现,也就是说试验结果的不确定性为零,此时信息熵的值为零。若 $p(x_1)=p(x_2)=\cdots=p(x_q)=\frac{1}{q}$,则每次实验哪个 x_i 会出现是最不确定的,这时不确定性是最大的,信息熵的值为 $\log(q)$。这样在确定子空间和优化迭代过程中采用香农的信息熵作为空间选择和数据归类不确定性的量度。

1.2 符号定义及有关公式介绍

设所给的分类数据集 D,$D=\{X_1,X_2,\cdots,X_N\}$,包含 N 个数据。数据 $X_t=(x_1,x_2,\cdots,x_d)$,$t=1,2,\cdots,N$,每个数据都有 d 个属性。将数据集归类到 k 个类 $C=\{C_1,C_2,\cdots,C_k\}$ 中,$C_i\subseteq D$,$C_i\cap C_{i'}=\Phi$,$i,i'=1,2,\cdots,k$,$i\neq i'$,各类对应的子空间 $P=\{P_1,P_2,\cdots,P_k\}$,$Q=\{1,2,\cdots,d\}$ 是全空间,$P_i\subseteq Q$。x_j 是数据的第 j 个属性的值,$x_j\in\{x_{j,1},x_{j,2},\cdots,x_{j,V_j}\}$,$j=1,2,\cdots,d$,$x_{j,m}$是第 j 个属性的域中的第 m 个值,$m=1,2,\cdots,V_j$。$f_i(x_{j,m})$是 $x_{j,m}$在第 i 类 C_i 中的频数。第 i 类的第 j 个属性域 $x_{j,\cdot}$ 中各个值的概率为

$$\rho_i(j)=(f_i(x_{j,1}),f_i(x_{j,2}),\cdots,f_i(x_{j,m}),\cdots,f_i(x_{j,V_j}))/|C_i|=(\rho_i(x_{j,1}),\rho_i(x_{j,2}),\cdots,\rho_i(x_{j,m}),\cdots,\rho_i(x_{j,V_j}))。 \qquad (2)$$

由信息熵公式(1)可得第 i 类的第 j 个属性的信息熵为

$$E_i(j)=-\sum_{m=1}^{V_j}[\rho_i(x_{j,m})\times\log(\rho_i(x_{j,m}))]。 \qquad (3)$$

全数据集 D 在第 j 个属性的信息熵为

$$E(j)=-\sum_{m=1}^{V_j}[\rho(x_{j,m})\times\log(\rho(x_{j,m}))], \qquad (4)$$

其中,$\rho(x_{j,m})$是属性值 $x_{j,m}$在 D 的第 j 维的概率。

由 COOLCAT[19] 知,数据的属性之间相互独立,则第 i 类的信息熵:

$$\tilde{E}_i=\frac{1}{|P_i|}\sum_{j\in P_i}E_i(j),\ P_i \text{ 是第 } i \text{ 类的子空间。}$$

由上式可得,此次聚类的平均信息熵为

$$\tilde{E}=\sum_{i=1}^{k}\left(\frac{|C_i|}{|D|}\tilde{E}_i\right)。 \qquad (5)$$

公式(5)就是在 2.2 节优化过程中用到的目标函数,用它来决定迭代继续或者终止。

1.3 信息熵与分类数据聚类

对于分类型数据聚类,传统的相异度量度公式已经不再适合,下面给出了一个简单的分类数据集 $D=\{X_1,\cdots,X_6\}$,每个数据包含 9 个维度,分别取值为 A、B、C、D,数据如下所示:

X_1:	A	A	A	B	A	C	C	D	B
X_2:	A	A	A	A	B	A	B	C	D
X_3:	A	A	A	C	C	B	D	A	A
X_4:	D	A	C	B	D	C	B	B	B
X_5:	B	C	B	D	A	B	B	B	B
X_6:	C	D	B	D	B	A	B	B	B

显然,数据包含两个明显的子空间聚类,一类 $C_1 = \{X_1, X_2, X_3\}$,对应子空间 $P_1 = \{1, 2, 3\}$;另一类 $C_2 = \{X_4, X_5, X_6\}$,对应的子空间 $P_2 = \{7, 8, 9\}$。由公式(5)可得,此时聚类的平均信息熵 $\tilde{E} = 0$。如果采用传统的 Jaccard/Dice 系数或汉明距离[10],由于 X_1 与 X_4 的相异度小于他们与其他数据的相异度,则 X_1 和 X_4 将会归并到同一类中,得到另一种聚类结果:$C'_1 = \{X_1, X_2, X_3, X_4\}$,$P'_1 = \{1, 2, 3, 4, 7, 9\}$,$C'_2 = \{X_5, X_6\}$,$P'_2 = \{3, 4, 7, 8, 9\}$,$\tilde{E}' = 0.925$。显然,前一次的聚类结果要比后一次的聚类结果要好得多。这是因为传统的度量方法都是基于数据与数据两者之间的相异度来度量的,而没有考虑到整体数据的概率分布。所以传统度量方式的分类型数据聚类在这样的低维数据下也容易出现"维数灾"问题[11],而基于信息熵的子空间聚类就不会出现上述问题。

于是 ESCHCD 采用信息熵作为函数度量的子空间聚类算法,算法在子空间选择和优化过程中都用到了信息熵,这样在算法的整个过程中都考虑到了数据与数据之间和属性值与属性值之间的整体概率分布,最终可得到最优的聚类结果。

2　基于信息熵的高维分类数据子空间聚类算法

前面已经定义了目标函数(5),并且详细地介绍了目标函数的导出过程。ESCHCD 算法的目的就是将数据集中的数据归类到 k 个类中,找出各类对应的子空间,并且保证目标函数(5)的值最小。以下算法介绍如下。

第一步:将数据集 D 做初始的划分。可以采取多种方法对数据集进行初始划分,其中一种方法就是将数据随机的划分到 k 个类中,这种方法虽然在一定程度上可以避免陷入局部极小,但在优化阶段需要更多的迭代次数以达到最优的聚类结果。PROCLUS[4]、COOLCAT[19]等采用贪婪算法来选取相异度最大的 k 个数据作为中心点,然后再将数据集归类到 k 个类中,这种方法用较小的代价得到较优的初始归类,大大减少后续优化阶段的迭代次数,降低算法的时间消耗。基于此,ESCHCD 采用这种贪婪算法对数据做初始归类。详细过程如 2.1 节所示。

第二步:经过初始划分后,进入优化阶段,优化阶段的目的是最小化目标函数(5)。首先进行子空间选择,然后基于各个类的子空间来计算信息熵,并进行迭代寻优。若将一个类中的数据转移到其他类中,使得目标函数的值减小,则将该数据转移到另一个类;否则,该数据保持在原类中,继续遍历下一个个体。当所有类中的数据无法转移时退出迭代。详细过程如 2.2 节所示。

以下为算法的步骤。

输入:数据集 D,类的个数 k
输出:聚类 C,及其对应的子空间 P
Begin
　初始化——Initialization
　　若 $|D| > |S|$ 则从 D 中抽出样本集 S,否则 D 为样本集,即 $S = D$
　　用贪婪算法 Greedy(S,k)取出 k 个相异度尽量大的数据,存于 M
　将 D 中的数据归类到以 M 中的数据为中心点的类中 $C = \{C_1, C_2, \cdots, C_k\}$
优化——Optimization
　repeat
　　对归类得到的 C 确定各类的子空间,得到对应的子空间 $P = \{P_1, P_2, \cdots, P_k\}$
　　for $i = 1$ to $|D|$
　　数据 X_i 属于类(C_l, P_l),$l = 1, 2, \cdots, k$
　　for $m = 1$, $m \neq l$ to k do
　　if 不等式(8)为真
　　数据 X_i 由 C_l 类转移到 C_m 类中
　　break
　　end if
　　end for
　　end for
　until (直到所有类中的数据都不再转移)
　return C, P
End

算法的详细过程请看 2.1 节和 2.2 节。

2.1　初始化阶段

初始化阶段,用贪婪算法从数据集中选出相异度尽量大的 k 个数据,将数据集中的其他数据归类到以它们为中心点的 k 个类中。

然而用贪婪算法从一个大的数据集中选出 k 个

数据需要消耗大量的时间,于是 ESCHCD 采用抽样的方法来减小计算量,即从原数据集 D 中随机抽取样本 S ($|S| \ll |D|$),再从 S 中用贪婪算法选取差异度尽量大的 k 个数据。抽样方法在不损失精度的情况下,可大大减少计算量。

在 COOLCAT 中,确定样本集 S 的大小公式为

$$|S| = k\rho + k\rho\log\left(\frac{1}{\delta}\right) + k\rho\sqrt{\left(\log\left(\frac{1}{\delta}\right)\right)^2 + 2\log\left(\frac{1}{\delta}\right)}, \tag{6}$$

其中 $\rho = \frac{|D|/k}{m}$,m 为聚类所要求的类中数据成员的最小个数,δ 为置信度。由 PROCLUS[4] 知,可取 $m = 0.1 * (|D|/k)$,由 SUBCAD[15] 知 $\delta = 0.01$,于是(6)可化为

$$|S| \approx 10k + 46k + 55k \approx 111k。$$

当然,抽样只是针对于大数据集,而对于小数据集抽样就无关紧要,即当 $|D| > |S|$ 是才采取抽样策略,否则直接在数据集 D 上选择相异度尽量大的 k 个数据。

定义 1 分类型数据 X_i 与 X'_i 之间的距离方程为

$$\mathrm{Dist}(X_i, X'_i) = \frac{\sum_{j=1}^{d} \| x_j - x'_j \|}{d},$$

若 x_j 与 x'_j 相同则 $\| x_j - x'_j \| = 0$,反之 $\| x_j - x'_j \| = 1$。以下为贪婪算法的具体步骤。

贪婪算法: Greedy(S,k)

{Dist(,)是定义 1 中定义的距离函数}

```
Begin
  M = {m1} {m1 是 S 中随机抽取的一个数据}
  for each x  S/M
    d(x) = Dist(x, m1) {计算 S 中的每个数据到 m1 的距离}
  end for
  for i = 2 to k {再选出 k-1 个距离最大的数据}
    {选择与 M 中的数据距离最大的数据 mi}
    如果 mi  S/M 使得
      d (mi) = max(d(x) | x  S/M)
    for each x  S/M
      d (x) = min(d (x), Dist(x, mi))
    end for
  end for
return M
End
```

选出相异度尽量大的 k 个数据后,利用定义 1 中的距离公式将数据集归类到以这 k 个数据为中心点的类中,即如果某个数据到第 i 个的中心点的距离最小,就将它划归到 C_i 中,直到数据集 D 中的数据都划分到各个类中。

2.2 优化阶段

在初始化阶段得到了数据集 D 的初始化分 $C = \{C_1, C_2, \cdots, C_k\}$,优化阶段先根据数据集的初始划分 C 计算出各个类的子空间,然后在迭代寻优过程中根据各类中的数据成员是否变化,判断迭代寻优进程是继续还是结束。简言之,优化阶段的主要任务是子空间的查找和迭代优化。

2.2.1 确定子空间

ESCHCD 是自动确定子空间大小的聚类算法,因此在子空间的确定过程中无需预设子空间的大小,子空间大小由子空间的平均熵值和噪声空间(噪声属性的集合,也就是子空间的补集)的平均熵综合确定。

根据数据集的归类 $C = \{C_1, C_2, \cdots, C_k\}$ 和属性信息熵的计算公式(3),可计算出各类中各属性的信息熵,将各类中各属性的熵值存储在一个 $k \times d$ 的矩阵 $\boldsymbol{E}$ 中,如

$$\boldsymbol{E} = \begin{bmatrix} E_1(1) & E_1(2) & \cdots & E_1(d) \\ \vdots & \vdots & \vdots & \vdots \\ E_k(1) & E_k(2) & \cdots & E_k(d) \end{bmatrix}_{k \times d},$$

按行将矩阵 $\boldsymbol{E}$ 中的值标准化到[0,1]区间,即

$$\mathrm{std}E_i(j) = \frac{E_i(j) - \mathrm{Min}_i}{\mathrm{Max}_i - \mathrm{Min}_i},\ i = 1,2,\cdots,k, j = 1,2,\cdots,d。$$

其中 $\mathrm{Min}_i = \min\{E_i(1), E_i(2), \cdots, E_i(d)\}$,$\mathrm{Max}_i = \max\{E_i(1), E_i(2), \cdots, E_i(d)\}$,如果 $\mathrm{Min}_i = \mathrm{Max}_i = 0$ 则 $\mathrm{std}E_i(j) = 0$。

$\boldsymbol{E}$ 标准化后的熵矩阵 std$\boldsymbol{E}$ 为

$$\mathrm{std}E = \begin{bmatrix} \mathrm{std}E_1(1) & \mathrm{std}E_1(2) & \cdots & \mathrm{std}E_1(d) \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{std}E_k(1) & \mathrm{std}E_k(2) & \cdots & \mathrm{std}E_k(d) \end{bmatrix}_{k \times d}。$$

由信息熵的物理意义可知,当熵值越小说明联系越紧密、相异性越小,于是定义变量 MS (matched subspace)、NS (noise subspace)分别表示对应子空间的平均熵值和噪声空间(噪声属性的集合,也就是子空间的补集)的平均熵值(P_i 是第 i 类对应的子空间,R_i 是其噪声空间),这里规定每个类的子空间不小于 2。MS 和 NS 的公式定义如下:

$$\mathrm{MS}(C_i, P_i) = \frac{\sum_{j \in P_i} \mathrm{std}E_{i,j}}{|P_i|}, |P_i| \geqslant 2。$$

$$\mathrm{NS}(C_i, R_i) = \begin{cases} \dfrac{\sum_{j \in R_i} \mathrm{std}E_{i,j}}{|R_i|}, & \text{if } R_i \neq \phi, R_i = Q \backslash P_i; \\ 1, & \text{if } R_i = \phi。 \end{cases}$$

可知,MS (C_i, P_i) 和 NS(C_i, R_i) 的取值都在[0,1]区间,按照子空间聚类的特征,若在子空间 P_i 上存在类 C_i,则类中的数据投影到 P_i 上时是稠密的,而在 P_i 的余空间上的投影是稀疏的,从数学上体现就是 MS(C_i, P_i) 的取值较小,同时 NS(C_i, R_i)的取值较大。因此搜索子空间 P_i 和类 C_i 的过程转化为使 MS(C_i, P_i)最小化和 NS(C_i, R_i)最大化。显然,MS(C_i, P_i) 和 NS(C_i, R_i)有着内在的联系,一般情况下,当 P_i 和 C_i 接近于要找的子空间聚类时,MS(C_i, P_i)的值减小,同时余空间 R_i 的数据变得稀疏,NS(C_i, R_i)的取值增大(或 1-NS(C_i, R_i)减小),因此确定子空间就是要使 MS(C_i, P_i) 和 1-NS(C_i, R_i)之间平衡,并同时达到"极小",以得到各类的最优子空间。将 MS(C_i, P_i) 和 1-NS (C_i, R_i)分别映射到直角坐标系 X 轴和 Y 轴的点 x、y 上(a、b 分别是点 x、y 之间的连线和点 x_0、y_0 之间的连线),如图 1。

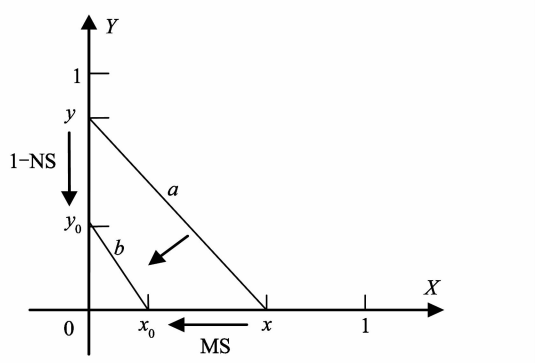

图 1　对应子空间和噪声空间的平均熵值的对应关系

Fig. 1　The relationship of average entropy of matched subspace ang noise subspace

从图 1 中可以看出,随着 MS(C_i, P_i) 和 1-NS (C_i, R_i)值减小,点 x、y 分别移动到点 x_0、y_0 处,x_0 和 y_0 间的距离逐渐减小,即线段 b 的长度小于线段 a 的长度。当 MS(C_i, P_i)和 1-NS(C_i, R_i)达到"平衡"时也就是 x_0 和 y_0 间的距离最短,此时得到的子空间 P_i 就是所求的子空间。于是将 x_0 和 y_0 间的距离可以定义为目标函数:

$$\text{Func}(C_i,P_i)=\sqrt{\text{MS}(C_i,P_i)^2+(1-\text{NS}(C_i,Q\backslash P_i))^2},\quad P_i\subseteq Q\ \text{且}\ |P_i|\geqslant 2。\tag{7}$$

因此确定子空间的过程中,使 Func(C_i, P_i)值最小的 P_i,就是 C_i 对应的子空间。于是就可得到归类后的数据 $C=\{C_1, C_2, \cdots, C_k\}$ 对应的子空间 $P=\{P_1, P_2, \cdots, P_k\}$。

显然,这里的子空间选择方法是不仅考虑到数据的相关子空间,而且也考虑了数据在余空间(噪声空间)的表现,同时子空间选择方法建立在各类数据成员的整体概率分布上的,因而采用无监督的方式得到各类对应的最优子空间。

2.2.2　迭代寻优

设数据 X 是类(C_l, P_l)的一个数据成员,如果将 X 转移到另一个类(C_t, P_t)中,其中 $l\neq t, l, t\leqslant k$,使得目标函数(5)的值减小,即未转移前的目标函数值 $\tilde{E}$ 大于移后的目标函数值 $\tilde{E}'$,也就是说如果 $\tilde{E}>\tilde{E}'$则将数据 X 转移到(C_t, P_t)类中。

由公式(5)$\tilde{E}=\sum_{i=1}^{k}\left(\frac{|C_i|}{|D|}\tilde{E}_i\right)$可得

$$\tilde{E}'=\sum_{i\neq l,t}\left(\frac{|C_i|}{|D|}\tilde{E}_i\right)+\frac{|C_l|-1}{|D|}E(C_l-X,P_l)+\frac{|C_t|+1}{|D|}E(C_t+X,P_t),$$

其中 $E(C_l-X,P_l)$ 和 $E(C_t+X,P_t)$ 分别表示第 l 类去除数据 X 后的平均熵值和第 t 类添加数据 X 后的平均熵值。由 $\tilde{E}>\tilde{E}'$可得,

$$\sum_{i=1}^{k}\left(\frac{|C_i|}{|D|}\tilde{E}_i\right)>\sum_{i\neq l,t}\left(\frac{|C_i|}{|D|}\tilde{E}_i\right)+\frac{|C_l|-1}{|D|}E(C_l-X,P_l)+\frac{|C_t|+1}{|D|}E(C_t-X,P_t),$$

上面公式可化简为

$$|C_l|E(C_l,P_l)-(|C_l|-1)E(C_l-X,P_l)>(|C_t|+1)E(C_t+X,P_t)-|C_t|E(C_t,P_t)。\tag{8}$$

因此在判断某个类中的数据是否能转移到另一个类时,只要判断是否满足不等式(8),满足则转移,否则不转移。也就是说,将迭代寻优过程中的目标函数(5)简化为不等式(8),这样可大大减少算法在迭代寻优过程中的计算量和时间消耗。

于是迭代寻优的过程简化为:对每个类的数据成员都扫描一遍作为一次遍历,根据不等式(8)判断扫描过程中数据是否可以转移到其他类中,若满足则转移,否则保持不变。如果某次遍历结束后有数据转移到其他类,则在该次遍历结束后得到优化的新归类 $C=\{C_1, C_2, \cdots, C_k\}$,再对新的归类重新进行子空间选择得到新的对应子空间 $P=\{P_1, P_2, \cdots, P_k\}$,然后进行再次遍历,直到某次遍历结束后没有数据转移到其他类,此时迭代结束,即得到最优归类 C 及其对应的子空间 P,此时目标函数(5)的值也最小。

由 2.2.1 节和 2.2.2 节的确定子空间和迭代寻优过程可知道,本算法的子空间选择是无监督的、迭代是全局优化的,而且聚类结果与数据排列顺序是

无关的，充分体现算法的顺序无关性。

3 实验及分析

通过对人工数据和真实数据两组数据进行实验来验证分析新算法的性能，并与 k-modes[13]、DH-CC[15]、COOLCAT[19] 和 SUBCAD[15] 算法进行比较。本研究中采用准确率、聚类的平均信息熵和 CU(category utility)[16]，这 3 项指标来对其聚类结果进行比较。平均信息熵也就是公式(5)的值，熵值越小聚类结果越好。准确率和 CU 的计算过程如下(人工数据和真实数据的分类情况是已知的)：

设第 i 类的准确率为 CR_i，第 i 类数据本身含有的数据个数 NUM_i，聚类结果中对应第 i 类的类中含有原第 i 类的数据个数为 A_i，则 $\mathrm{CR}_i = A_i/\mathrm{NUM}_i$。聚类的总正确率 $\mathrm{CR} = \dfrac{\sum_i^k A_i}{N}$，总正确率越大聚类结果越好。

CU 的计算公式为

$$\mathrm{CU} = \sum_{i=1}^{k} \frac{|C_i|}{N} \sum_{j=1}^{d} \sum_{v \in D_j} [p(x_j = v \mid C_i)^2 - p(x_j = v)^2],$$

CU 度量的是聚类的属性值在类内的概率与在全数据集上概率的平均平方差，由公式的物理意义可知 CU 的值越大说明聚类的结果越好。

3.1 人工数据

此实验中，生成了 150 × 10 的分类数据集，共 150 个数据，每个数据都有 9 个属性：A、B、C、D、E、F、G、H、I，所有数据分为 3 类，其中第 1 类所对应的子空间为 ABC 三维，其他维上是噪声数据；第 2 类所对应的子空间为 DEF 三维，其他维上是噪声数据；第 3 类所对应的子空间为 GHI 三维，其他维上是噪声数据。

ESCHCD 是用 Matlab 实现，在 Intel Core Quad CPU Q6600（时钟频率是 2.4 GHz）和 4 G 内存的联想台式电脑上运行。

实验结果显示，ESCHCD 能够准确的找到各个类的对应子空间，聚类结果的准确率都超过 98%。更进一步，对数据集中的数据顺序和整体属性列的顺序进行重组，实验的结果和时间消耗与之前是一样的，这就验证了 ESCHCD 是数据顺序无关的。因此 ESCHCD 在人工数据上的聚类效果与预期的聚类结果和聚类顺序无关性是一致的。

另一方面，从数据集大小和数据属性个数两方面验证了 ESCHCD 的可扩展性，通过变化人工数据集中的数据个数和属性个数，分别试验这两个因素对 ESCHCD 运行时间的影响。实验的结果是：当数据的属性个数 Dim 为 9、18、27、36、45、54、63 时，所对应的时间消耗 DimTime 分别为 3.94、7.26、10.29、13.54、16.42、19.04、21.6(s)，而当数据集的数据个数 DataSize 为 150、750、1 350、2 100、2 700、3 300、3 900、4 500、5 100 时，时间消耗 DataTime 分别为 3.94、25.79、50.54、83.43、110.8、138.86、167.49、198.43、228.24(s)。将 Dim 和 DimTime、DataSize 和 DataTime 分别作为数据对，在直角坐标系中对他们连线。如图 2、图 3 所示。

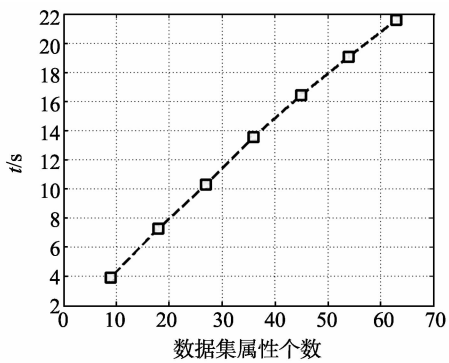

图 2 数据集属性与时间消耗关系

Fig. 2 The relationship between the number of features and running time

图 2 是数据集属性个数与时间消耗之间的关系图，横轴是属性个数，纵轴是消耗时间。图 2 显示，算法的运行时间与数据集的属性个数是线性关系，所以 ESCHCD 可有效地扩展到高维分类型数据聚类。

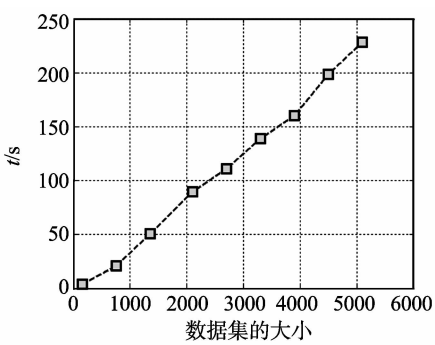

图 3 数据集数据个数与时间消耗关系

Fig. 3 The relationship between the number of data points and running time

图 3 是数据集大小与时间消耗的关系图，横轴是数据集的数据个数，纵轴是消耗的时间。通过对试验数据进行数据拟合，发现数据集的大小与时间消耗的函数关系是 $O(N\log_2 N)$，显然，对于大数据聚类这样的时间复杂度是可接受的，因此新算法可以扩展到大数据集和超大数据集聚类。

简言之，图 2 和图 3 显示 ESCHCD 在高维数据和大数据集上都有很好的扩展性。

3.2　真实数据

在这里选用了 Congressional Voting Records Data Set(Votes)、Mushroom Data Set (Mushroom)和 Soybean Data(Soybean)3 个典型的真实分类数据集来做试验对比,这 3 个数据集都来自 UCI 机器学习网站[21]。

3.2.1　数据描述

Votes 是 435 ×17 的数据集,第一列为类标识,数据分为 2 类,一类是民主党包含 267 个数据,另一类是共和党包含 168 个数据。数据集中有些数据的属性值缺失,将那些缺失的属性值补以“?”作为标识。

Soybean 是从“Soybean (Large) Data Set”中选择了其中的一部分数据组成的,其是 47 ×21 的数据集,第一列为类标识,数据分为 4 类:第一类是 diaporthe - stem - canker 包含 10 个数据,第二类是 charcoal-rot 包含 10 个数据,第三类是 rhizoctonia-root-rot 包含 10 个数据,第四类是 phytophthora-rot 包含 17 个数据。

Mushroom 是 8 124 ×23 的数据集,第一列为类标识,数据分为 2 类,一类是可食用的蘑菇包含 4 208个数据,另一类是有毒的蘑菇包含3 916个数据。数据集中有些数据的属性值缺失,将那些缺失的属性值补以“?”作为标识。

3 个数据集的数据信息,如表 1 所示。

表 1　数据集的数据信息
Table 1　The information of three data sets

数据集名称	特征(属性)	数据集的大小	类数
Soybean	21	47	4
Votes	17	435	2
Mushroom	23	8 124	2

3.2.2　实验结果及对比

本研究算法的实验结果,详见表 2。由表 2 可知,ESCHCD 能够准确的找出数据集的各个类,具有很高的准确率,而且熵值较小,CU 值较大,充分显示了 ESCHCD 的有效性。

表 2　ESCHCD 的聚类结果
Table 2　The clustering results of ESCHCD

数据集	准确率	熵值	CU 值	类数
Soybean	0. 957	0. 339	7. 104	4
Votes	0. 89	0. 524	2. 912	2
Mushroom	0. 883	0. 719	7. 131	2

将 ESCHCD 的实验结果与 DHCC、COOLCAT、*k*-modes 和 SUBCAD 算法的实验结果进行对比,表 3、表 4、表 5 和表 6 分别是 DHCC、COOLCAT、*k*-modes 和 SUBCAD 算法在 3 个数据集上的实验结果。

表 3　DHCC 的聚类结果
Table 3　The clustering results of DHCC

数据集	准确率	熵值	CU 值	类数
Votes	0. 832	0. 420	2. 22	2
Mushroom	0. 789	0. 264	6. 1	10

表 4　COOLCAT 的聚类结果
Table 4　The clustering results of COOLCAT

数据集	准确率	熵值	CU 值	类数
Votes	0. 618	0. 574	1. 040	2
Mushroom	0. 702	0. 219	6. 133	10

表 5　*k*-modes 的聚类结果
Table 5　The clustering results of *k*-modes

数据集	准确率	熵值	CU 值	类数
Soybean	0. 923	0. 425	6. 874	4
Votes	0. 805	0. 663	2. 838	2
Mushroom	0. 853	0. 738	4. 045	2

从表 1 可以看出,Mushroom 数据原本只有两个类,而表 3 和表 4 显示 Mushroom 被划分为 10 个类,显然 DHCC 和 COOLCAT 对 Mushroom 数据集聚类是失败的。此外,虽然 DHCC 和 COOLCAT 在 Votes 数据集上可以准确聚类,但是在准确率和 CU 值两个指标上,两个算法的结果都远逊于 ESCHCD。由此可知,ESCHCD 算法的聚类效果 DHCC、COOLCAT 算法要好。

表 5 与表 2 相比,*k*-modes 算法的聚类结果的准确率和熵值都显示效果完全不如 ESCHCD 的聚类结果,虽然 *k*-modes 在 Votes 数据集上的 CU 值的结果略比后者要好,但是总的来说 ESCHCD 的聚类结果明显优于 *k*-modes。

表 6 显示了 SUBCAD 算法的聚类结果在 3 个评价标准上面都不如 ESCHCD 的聚类结果。

表 6　SUBCAD 的聚类结果
Table 6　The clustering results of SUBCAD

数据集	准确率	熵值	CU 值	类数
Soybean	0. 934	0. 412	6. 985	4
Votes	0. 614	0. 688	2. 743	2
Mushroom	0. 857	0. 732	5. 368	2

为了更加直观地对比各个算法的效果,将各个算法在 Votes 数据集上的聚类结果用直方图的形式描述出来,如图 4 所示。

图 4 是 5 种算法在 Votes 数据集上的结果对比直方图,显然 ESCHCD 对 Votes 数据集的聚类结果在准确率、熵值和 CU 值 3 项指标上都要明显优于 *k*-modes、SUBCAD 和 COOLCAT 算法在 Votes 上的聚类结果,虽然 ESCHCD 的聚类结果在熵值上略高于 DHCC,但是其他两项指标都是 ESCHCD 明显占

优。因此 ESCHCD 在 Votes 数据集上的聚类结果优于其他 4 种算法。

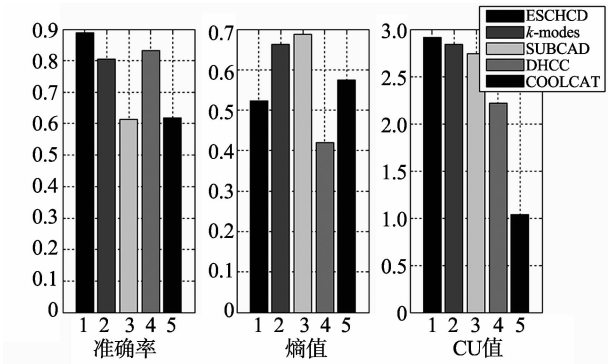

图 4 数据集 Votes 的对比图

Fig. 4 The comparison of five algorithms on Votes data

总之,上面的表格和直方图都说明了 ESCHCD 算法的试验结果要比 *k*-modes、SUBCAD、DHCC 和 COOLCAT 算法的结果要好,充分体现了 ESCHCD 的有效性和优越性。

4 结论

本研究提出了一种基于信息熵的高维分类数据子空间聚类算法,该算法通过以信息熵作为度量,在子空间的搜索中不仅考虑相关属性,也考虑了噪声属性对子空间选择的影响,并在两者之间找到平衡,从而无监督地确定了各类的最优子空间。算法在迭代优化过程中利用了信息熵的全局性,对聚类结果进行全局优化,最终得到最优的聚类结果。试验结果证明了算法对于高维分类数据聚类问题是有效的,而且有良好的可扩展性。

虽然 ESCHCD 确定子空间是无监督,无需输入预设子空间的大小,但是算法中依然要输入类的个数 *k*,参数 *k* 对聚类效率的影响很大。如何省略参数 *k* 进行问题求解将是一个研究重点。同时算法在执行过程中,内存使用量较大,对于当前内存无法容纳的超大数据集聚类问题,存在一定困难。因此,如何提高算法的内存使用率,以适应超大数据集聚类是算法的另一个改进方向。

参考文献:

[1] Jiawei Han, Micheline Kamber. 数据挖掘概念与技术[M]. 范明,孟晓峰,译. 北京:机械工业出版社,2008.

[2] BOUGUESSA M, WANG S, JIANG Q. A *k*-means-based algorithm for projective clustering [C]//Proceedings of 18th IEEE International Conference on Pattern Recognition. Hong Kong:[s. n.], 2006:888-891.

[3] HUANG Z. Exstensions to the *k*-means algorithm for clustering large data sets with categorical value[J]. Data Mining and Knowledge Discovery, 1998, 2(3): 283-304.

[4] AGGARWAL C C, PROCOPIUC C, WOLF J, et al. Fast algorithms for projected clustering [C]//Proceedings of ACM SIGMOD Conference on Management of Data. New York, USA:[s. n.], 1999:261-272.

[5] AGRAWAL R, GEHRKE J, GUNOPULOS D. Automatic subspace clustering of high dimensional data for data mining applications [C]//Proceedings of ACM-SIGMOD. Seattle, WA:[s. n.], 1998: 94-105.

[6] GUHA S, Rastogi R, SHIM K. ROCK: a robust clustering algorithm for categorical attributes [C]//Proceedings of ICDE'99. [S. l.]:[s. n.], 1999:512-521.

[7] Boris Mirkin. Reinterpreting the category utility function [J]. Machine Learning, 2001, 45(2): 219-228.

[8] ORDONEZ C. Clustering binary data streams with K-means[C]//Proceedings of ACM SIGMOD Workshop on Data Ming and Knowledge Discovery. San Diego, CA, USA:[s. n.], 2003:12-19.

[9] 王好芳,吴美,陈文艳. 模糊聚类分析在区域水资源承载能力评价中的应用[J]. 山东大学学报:工学版,2009,39(3):139-143.
WANG Haofang, WU Mei, CHEN Wenyan. Application of fuzzy cluster analysis to regional water resources carrying capacity evaluation[J]. Journal of Shandong University:Engineering Science, 2009, 39(3): 139-143.

[10] TAN P, STEINBACH M, KUMAR V. Introduction to Data Mining[M]. [S. l.]:Addison-Wesley, 2005.

[11] 杨分召,朱扬勇. 高维数据挖掘中若干关键问题的研究[D]. 上海:复旦大学,2003:4.
YANG Fenzhao, ZHU Yangyong. The research on a few key issues in high dimensional data mining[D]. Shanghai: Fudan University, 2003:4.

[12] Lance Parsons, Ehtesham Haque, Huan Liu. Subspace clustering for high dimensional data: a review [C]// Proceedings of SIGKDD Explorations. New York, USA:[s. n.], 2004:90-105.

[13] SAN O M, HUYNH V, NAKAMORI Y. An alternative extension of the *k*-means algorithm for clustering categorical data [J]. International Journal of Applied Mathematics and Computer Science, 2004,14(2): 241-247.

[14] Tengke Xiong, Shengrui Wang, André Mayers, et al. A new mca-based divisive hierarchical algorithm for clustering categorical data[C]//Proceedings of IEEE 9th ICDM Miami. FL, USA:[s. n.], 2009: 1058-1063.

[15] GAN G, WU J. Subspace clustering for high dimensional categorical data [J]. Southern Ontario Statistics Graduate Student Seminar Days, 2003,6(2):87-94.

[16] GLUCK A, Corter J. Information, uncertainty, and the utility of categories [C]//Proceedings of the Seventh

Annual Conference of the Cognitive Science Society. [S. l.]:[s. n.],1985:283-287.
[17] 王彬. 熵与信息[M]. 西安:西北工业出版社, 1994.
[18] 梁吉业,李德玉. 信息系统中的不确定性与知识获取[M]. 北京:科学出版社,2005.
[19] BARBARA D, LI Y, COUTO J. COOLCAT: an entropy-based algorithm for categorical clustering[C]//Proceedings of ACM CIKM. New York, USA:[s. n.], 2002: 582-589.
[20] KIM Minho, RAMAKRISHNA R S. Projected clustering for categorical datasets [J]. Pattern Recognition Letters, 2006, 27(12): 1405-1417.
[21] UCI data base[EB/OL]. (2007)[2010-12-20]. http://archive. ics. uci. edu/ml.

(编辑:胡春霞)