一种基于聚类集成技术的混合型数据聚类算法

罗会兰[1,2] 危 辉[2]

(江西理工大学信息工程学院 赣州 341000)[1]

(复旦大学计算机科学技术学院 上海市智能信息处理重点实验室 上海 200433)[2]

摘 要 提出了一种基于集成技术和谱聚类技术的混合数据聚类算法CBEST。它利用聚类集成技术产生混合数据间的相似性,这种相似性度量没有对数据特征值分布模型做任何的假设。基于此相似性度量得到的待聚类数据的相似性矩阵,应用谱聚类算法得到混合数据聚类结果。大量真实和人工数据上的实验结果验证了CBEST的有效性和它对噪声的鲁棒性。与其它混合数据聚类算法的比较研究也证明了CBEST的优越性能。CBEST还能有效融合先验知识,通过参数的调节来设置不同属性在聚类中的权重。

关键词 聚类集成,混合型数据,相似性度量

中图法分类号 TP391.4 **文献标识码** A

Clustering Algorithm for Mixed Data Based on Clustering Ensemble Technique

LUO Hui-lan[1,2] WEI Hui[2]

(School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China)[1]

(Shanghai Key Laboratory of Intelligent Information Processing, School of Computer Science, Fudan University, Shanghai 200433, China)[2]

Abstract A clustering algorithm based on ensemble and spectral technique named CBEST that works well for data with mixed numeric and categorical features was presented. A similarity measure based on clustering ensemble was adopted to define the similarity between pairs of objects, which makes no assumptions of the underlying distributions of the feature values. A spectral clustering algorithm was employed on the similarity matrix to extract a partition of the data. The performance of CBEST was studied on artificial and real data sets. Results demonstrate the effectiveness of this algorithm in clustering mixed data tasks and its robustness to noise. Comparisons with other related clustering schemes illustrate the superior performance of this approach. Moreover, CBEST can infuse prior knowledge effectively to set the weights of different features in clustering.

Keywords Clustering ensemble, Mixed data, Similarity measure

1 国内外研究现状分析

传统的聚类算法都只考虑数据是数值型的,并在多维几何空间中度量和表示数据。还有一些概念型聚类算法单纯考虑数据是概念型的,它们采用信息理论中的一些方法来度量点和类之间的相似性。但随着聚类分析的应用从科学领域、工程领域发展到医学、商业和社会领域,大多数的数据由概念型和数值型属性来描述,也就是说大多数的数据是混合型数据。要为混合型数据定义一个相似性度量并不容易,因为概念型数据和数值型数据具有不同的特点[1]。

1987年,D. H. Fisher在文献[2]中提出COBWEB算法,Fisher使用来源于信息理论的Category Utility (CU)度量对概念型数据进行聚类。这个度量划分数据的目标是最大化所得簇中正确预测特征值的概率,这种方法只适合于概念型数据。为了处理混合型数据,1990年,Fisher在文献[3]中提出COBWEB/3算法,第二年,他在文献[4]中进一步改进COBWEB/3,提出ECOBWEB算法。Fisher假设数值型特征值都是正态分布的,从而改变CU度量使之能处理数值型属性。COBWEB/3和ECOBWEB算法在决定簇结构时都没有考虑对象间的距离。

1996年,P. Cheeseman和J. Stutz在文献[5]中提出AUTOCLASS算法,它是基于贝叶斯理论的一种非监督式分类算法。它假设混合数据集是由一个有限混合分布模型产生的,在使用EM算法得到模型参数的最大似然估计值后,使用贝叶斯分类方法来导出最可能的类分布。类内混合概率分布函数是单个属性的概率分布的乘积,或属性间的联合分布。AUTOCLASS算法假设概念型属性为柏努利分布(Bernoulli distributions),数值型属性为高斯分布。AUTOCLASS算法也有概率模型的最大似然估计的过拟合问题。

1997年,Zhexue Huang在文献[6]中提出k-prototypes

到稿日期:2009-12-17 返修日期:2010-03-01 本文受国家973项目(No. 2010CB327900),国家自然科学基金(No. 60303007),上海科技发展基金(No. 08511501703),上海市智能信息处理重点实验室开放课题(No. IIPL-09-009)资助。

罗会兰(1974—),女,博士后,副教授,主要研究方向为机器学习、模式识别,E-mail:luohuilan@sina.com;危 辉(1971—),男,博士,教授,主要研究方向为认知模型、计算机视觉。

算法，它改进 k-means 算法来处理混合型数据的聚类。k-prototypes 算法将概念型属性与数值型属性分开，以便使用适合概念型属性的相似性度量和适合数据值型属性的距离度量，然后将两者结合起来表示混合型数据间的相似性。它和 k-means 算法一样具有效率高的优点，但同样也受初始化的影响，容易陷于局部极小值。

2002 年，在文献[7]中 Cen Li 和 Gautam Biswas 提出了 SBAC 算法（Similarity-Based Agglomerative Clustering）。SBAC 算法是一种凝聚型层次聚类算法，在度量对象间和类间的相似性时，采用了 Goodall 相似性度量[8]。Goodall 相似性度量赋于不经常出现的属性值匹配更大的权值。对于数值型属性，相似性度量不仅考虑特征值的差别大小，而且考虑值对出现的独特性。值对出现的独特性通过位于此值对区间内的数据点数来度量。

2005 年，在文献[9]中 He Zengyou 等人提出数据值型属性聚类和概念型属性聚类，然后将这些聚类结果看成是概念型数据，在其上应用概念型数据聚类算法来进行聚类并得到最终聚类结果。

本文利用聚类集成技术对混合型数据聚类，不强加给数据集任何的分布模型，也不直接使用相似性度量。我们提出了基于集成技术和谱聚类技术的混合型数据聚类算法 CBEST（Clustering Based on Ensemble and Spectral Technique）。CBEST 先将数值型属性作为单独的数据集，在其上得到一个聚类集体，利用此聚类集体得到数据间的相似性，同样，将概念型数据作为一个聚类集体，在其上也得到它们间的相似性度量，然后两者根据一定的先验信息结合起来，最后利用谱聚类算法的优越性得到混合数据的聚类。实验证明它比其他的混合数据聚类算法更好。本文第 2 节是算法 CBEST 的论述；第 3 节是实验分析；第 4 节是 CBEST 算法的小结；最后是总结及进一步的工作。

2 基于集成技术和谱聚类技术的混合型数据聚类算法 CBEST

算法的总体思路是先利用聚类集成技术得到混合型数据的相似性，然后基于此相似性，在其上应用谱聚类方法得到最终聚类结果。算法的核心是先在数值型数据上运行多次 k-means 得到一个聚类集体，或利用其他方法得到一个聚类集体，然后基于此得到一个互联合矩阵。另外对于概念型属性，把每一个属性当成一个聚类，则 m 个概念型属性就得到 m 个聚类，然后根据这个聚类集体也得到一个互联合矩阵。最后将这两个互联合矩阵以一个权值结合起来得到混合数据的互联合矩阵。互联合矩阵中的值是两个数据点被聚为同一个簇的次数占整个集体的比例[10]。我们把混合数据的互联合矩阵作为数据间的相似性矩阵，并在此相似矩阵上应用谱聚类算法 NJW[11] 来得到最终聚类结果。

2.1 基于集成技术的相似性度量方法

将混合型数据分成两个数据子集，一个子集由所有数值型属性构成，另一个子集由所有概念型属性构成，然后在两个数据子集上面各自产生聚类集体，这样对象间的相似性就可以根据两个对象被聚为同簇的次数来估计。这种相似性定义可以表示成对象互联合的强度矩阵，其中每对值为[12]：

$$S(i,j)=S(x_i,x_j)=\frac{1}{H}\sum_{k=1}^{H}\delta(\pi_k(x_i),\pi_k(x_j))$$
$$\delta(a,b)\equiv\begin{cases}1, & \text{if } a=b\\ 0, & \text{if } a\neq b\end{cases} \tag{1}$$

式中，H 表示聚类个数，$\pi_k(x_i)$，$\pi_k(x_j)$ 分别是在成员聚类 π_k 中 x_i，x_j 的簇标签。

2.1.1 为数值型属性计算相似性

我们把混合型数据集中的所有数值型属性单独提取出来构成一个数据集，并在此数据集上多次运行 k-means 算法，k 值设置得相对比较大，这样纯数值型数据就被分解成很多紧凑的球状簇。然后根据它们同时出现在相同簇中的次数来决定它们互联合的强度：

$$S_d(i,j)=\frac{n_{ij}}{N}=\frac{\sum_{l=1}^{N}C^l(i,j)}{N} \tag{2}$$

式中，n_{ij} 是数据对 x_i 和 x_j 在 N 个聚类中被分到相同簇的次数。如果它们在第 l 个成员聚类中被分到相同簇，则 $C^l(i,j)=1$，否则 $C^l(i,j)=0$。所以 N 个聚类成员的信息根据式(2)就集成起来了，并从中导出了数据对间新的相似性度量方法。

这种方法有两个参数需要设定，一个是 k，运行 k-means 算法时指定的簇个数，一个是 N，欲产生聚类集体的大小，或者说聚类成员个数。k-means 算法在这里被看成对数据执行一个分解，将数据分解成球状簇的集体。太小的 k 值可能不足于分离，但太大的值又会过于分解数据[13]。

2.1.2 为概念型属性计算相似性值

同样地，对于概念型属性，如果把每个属性值看成是簇标签的话，则每个属性可以看成是原来数据的一个聚类结果[14]。比如，表 1 所列的概念型数据有 8 条记录，2 个概念型属性。如果只考虑属性 “Gender”，我们可以将这 8 条记录聚成两簇(1,4,5,7)和 (2,3,6,8)。相似地，如果只考虑属性 “Chest Pain Type”，则可得到一个有 4 个簇的聚类{(1,7),(2,8),(3,5),(4,6)}。

表 1 概念型数据例子

Record Number	Gender	Chest Pain Type
1	F	angina
2	M	abnang
3	M	notang
4	F	asympt
5	F	notang
6	M	asympt
7	F	angina
8	M	abnang

所以如果待聚类混合型数据集有 m 个概念型属性，则可以得到 m 个聚类。这样就可以像数值型属性一样根据它们出现在同一簇中的次数来得到一个 $n\times n$ 互联合矩阵，n 表示待聚类数据集的大小，其中矩阵单元 (i,j) 的值为数据对 x_i 和 x_j 的互联合强度，或称为相似性，它的值定义如下：

$$S_c(i,j)=\frac{n_{ij}}{m}=\frac{\sum_{l=1}^{m}C^l(i,j)}{m} \tag{3}$$

式中，n_{ij} 是数据对 x_i 和 x_j 在 m 个聚类中被分到相同簇的次数。

2.1.3 结合数值型属性相似度和概念型属性相似度

我们将数值型属性的相似值与概念型属性相似值结合起来产生最终的相似性矩阵 S。

$$S=S_d+\alpha S_c \quad (4)$$

式中，α 是一个由用户确定的参数，如果 $\alpha>1$ 则概念型属性在聚类中占有重要作用，如果 $\alpha<1$，则给予数值型属性更重要的作用。如果我们对于数据具有这样的先验信息，则可以用来选择此参数。但当我们无这样的先验信息时，则只是简单地设置 $\alpha=1$。

最后，原待聚类混合型数据集的聚类通过应用谱聚类方法到此相似矩阵 S 上得到。

2.1.4 谱聚类技术

近年来，谱聚类技术(Spectral Clustering)有了很大的发展，并且显示出很强的聚类能力[11,15,16]。它们能发现具有不同复杂形状的类，还能发现具有不同密度的类。它们容易实现，不容易陷于局部最优点，对于数据的分布也没有做任何的假设。基于谱聚类的这种良好性能，我们选择在相似性矩阵 S 上应用谱聚类来完成混合型数据的聚类。在我们的实验中使用了文献[11]中的谱聚类算法 NJW，此算法概述如下：

begin

Step 1 将有 n 个数据的待聚类数据集 X 的相似性矩阵 $S\in R^{n\times n}$ 的对角线单元的值全部设置成 0，$S_{ii}=0$；

Step 2 定义对角线矩阵 D，它的值 $D_{ii}=\sum_{j=1}^{n}S_{ij}$，并构造规范化相似性矩阵 $L=D^{-1/2}SD^{-1/2}$；

Step 3 设置数据待聚类成的类个数 C；

Step 4 将矩阵 L 对应着的前 C 个最大特征值的特征向量 $u_1,u_2,\cdots,u_C$，组成一个矩阵 $U=[u_1,\cdots,u_C]\in R^{n\times C}$；

Step 5 将矩阵 U 的行向量规范化后得到一个新的矩阵 $Y\in R^{n\times C}$：$Y_{ij}=U_{ij}/(\sum_j u_{ij}^2)^{1/2}$；

Step 6 把 Y 看成是一个新的数据集，在其上应用 k-means 算法将数据聚类成 C 类；

Step 7 原待聚类数据集 X 的聚类结果直接从 Y 上得到，也就是说，X 与 Y 中相应行对应着同一个数据点。

end

2.1.5 算法 CBEST

假设待聚类混合数据集 $X=\{x_1,x_2,\cdots,x_n\}$ 有 m 个概念型属性和 d 个数值型属性 $m+d=p$。为了描述方便，将混合数据表示成 $x_i=\{x_i^{A_1},\cdots,x_i^{A_d},x_i^{B_1},\cdots,x_i^{B_m}\}$，前面 d 个属性是数字型属性，后面 m 个属性是概念型属性。整体的 CBEST (Clustering Based on Ensemble and Spectral Technique) 算法概述如下：

begin

Step 1 将待聚类数据集 X 的全部 d 个数值型属性组成一个新的数据集 A，如果 X 表示成一个矩阵，其中每行对应着一个数据点，则 A 由 X 的前 d 列组成，$A=[X_1,X_2,\cdots,X_d]$；

Step 2 在 A 上利用 k-means 算法的固有随机性，也就是每次都随机选择 k 个初始中心点，产生一个聚类集体(或利用其他方法产生一个聚类集体)，然后根据式(2)计算数值型属性的相似性；

Step 3 将每个概念型属性当成一个聚类，则 m 个概念型属性就成为一个聚类集体，然后根据式(3)计算概念型属性的相似性；

Step 4 根据式(4)结合数值型属性的相似性和概念型属性的相似性，得到原待聚类数据集上的一个相似性矩阵 S；

Step 5 在相似性矩阵 S 上运行谱聚类算法 NJW 得到原待聚类混合数据集 X 的聚类。

End

3 实验研究与分析

实验的目的有两个，一个是获得基于集成技术的相似性度量的更好理解。二是比较 CBEST 与其他几种经典混合型数据聚类算法。为了达到这个目标，我们使用了人工合成数据和真实数据。我们评估算法性能的方法是聚类错误率。因为我们使用的数据集的正确分类是已知的，所以只需将聚类结果与正确分类进行类标签的统一，从而计算出聚类错误率。对类标签进行统一的算法使用的是 Hungarian 方法[17]，它对 k 个类标签进行统一的计算复杂性是 $O(k^3)$。

3.1 基于集成技术的相似性度量实验分析

为了获得基于集成技术的相似性度量的更好理解，我们使用了文献[7]中的实验分析方法。我们在人工生成的数据集上进行此实验分析，在产生人工数据的过程中，分别对概念型属性和数值型属性进行人为破坏，以模拟数据收集过程中的噪声。然后计算在不同噪声比例下 CBEST 的错误率，以此来分析两种类型的属性噪声对基于集成技术的相似性度量的影响。

图 1 中的实验结果是在一个有 3 个相同大小的类，共 180 个数据的人工数据集上产生的。此人工数据集有 4 个属性，其中两个概念型属性和两个数值型属性。每个概念型属性的值是预先定义好的，平均分配到 3 个类中。第一个概念型属性对于每个类有一个唯一的值，而第二个概念型属性给每个类赋于了等概率的两个值。我们使用正态分布产生数值型属性，对于第一个类产生两个数值型属性的均值和标准差分别是 $N(\mu=4,\sigma=1)$ 和 $N(\mu=20,\sigma=2)$；对于第二个类，产生两个数值型属性的均值和标准差分别是 $N(\mu=10,\sigma=1)$ 和 $N(\mu=32,\sigma=2)$；对于第三个类产生两个数值型属性的均值和标准差分别是 $N(\mu=16,\sigma=1)$ 和 $N(\mu=44,\sigma=2)$。按照这种方法生成人工数据集后，分别单独对概念型数据和数值型数据进行人工破坏，以模仿噪声，但没有同时对两者进行人工破坏。目的是分析基于集成技术的相似性度量对于数值型属性噪声或概念型属性噪声的忍受程度，并且分析此相似性度量方法对于概念型属性和数值型属性的偏向。我们通过将每个类中的一定比例的数据的特征值改变成其他类的特征值来模拟人工噪声，而不是再加入噪声数据。在图 1 所示的实验结果中，我们破坏数据的比例从 0 增加到 0.5。

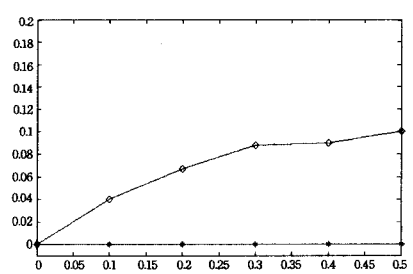

图 1 不同比例噪声数据对 CBEST 算法错误率(星形符号表示只破坏概念型属性，而钻石形符号表示只破坏数值型属性)

在有不同比例破坏数据的人工数据集上运行 CBEST 算法，并计算聚类错误率，这样就得到图 1 所示的一个折线图。

在此实验中，CBEST 算法中的参数 $\alpha=1$，在计算数值型属性相似性时产生的聚类集体大小为 100，运行 k-means 算法产生成员聚类时，参数 $k=3$(设置成正确类数)。图 1 中星号表示逐渐增加概念型属性的破坏程度对 CBEST 算法聚类错误率的影响，而钻石符号表示逐渐增加数值型属性破坏程度对 CBEST 算法聚类错误率的影响。从图 1 可以看出我们的 CBEST 算法随着噪声比例的增加，性能下降的幅度很小。对于数值型属性破坏程度的增加，CBEST 算法错误率从 0 增加到 0.1，而概念型属性的破坏对 CBEST 算法错误率几乎没有什么影响。从此实验可以得知我们的算法对于噪声数据具有一定的稳健性。

3.2 算法比较实验

在本小节的实验中，将在 3 个人工数据集和 3 个真实数据集上比较混合型数据聚类算法 AUTOCLASS[5]，k-prototypes[6]，SBAC[7] 和我们的 CBEST 算法。在所有的实验结果中，由于 SBAC[7] 不具有随机性，而且它的时间复杂性最高，因此所有有关 SBAC[7] 算法的结果都是一次实验的结果。而对于算法 k-prototypes[6] 和我们的算法 CBEST，由于它们具有一定的随机性，并且它们的时间复杂性低，因此所有有关它们的实验结果都是 20 次实验结果的平均值。AUTOCLASS[5] 的参数设置中，我们固定每次搜索的类数为真实类数，并且最大重新搜索次数设为 6(max-n-tries=6)。在运行我们的算法 CBEST 时，为计算数值型属性的相似性，我们运行 k-means 时设定 $k=20$，集体大小 $H=100$。下面将分别阐述在 3 个人工数据集和 3 个真实数据集上的比较实验。

算法 k-prototypes[6] 和我们的算法 CBEST 在结合概念型属性和数值型属性的相似性时都有一个参数 α 要设定。所以在本小节的实验中，对于这两个算法，我们分别在不同的 α 值上进行了实验，以分析不同 α 值对算法性能的影响。

1. 人工数据集 Dataset1 上的比较实验

我们人工生成了有 3 个类，每个类 60 个对象，共 180 个四维对象的数据集 Dataset1，其中前二维是数值型属性，后两维是概念型属性。用二维高斯分布生成二个数值型属性，产生两个数值型属性的多维高斯分布的均值和协方差分别是：第一个类 mu = [6 4]，sigma = [1 0;0 1]；第二个类 mu = [4 5.5]，sigma = [1 0;0 1]；第三个类 mu = [8 7]，sigma = [1 0;0 1]。两个数值型属性形成的形状如图 2 所示。然后再生成了两个概念型属性，其中一个概念型属性每个类中被分别赋于不同的两个值，另一个概念型属性每个类赋于不同的一个值，然后在此基础上对其进行一定程度的扰乱，即将一个类的值按一定程度随机替换成其他类的值。在这个数据集中，我们按 0.2 的概率来扰乱第一个概念型属性的值，以模拟真实数据中存在的噪声或异常点。

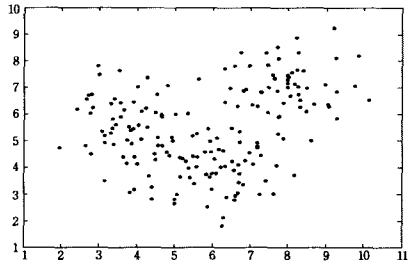

图 2 人工数据集 Dataset1 的两个数值型属性分布图

在生成人工数据集 Dataset1 后，我们在其上运行不同的混合数据聚类算法，并比较它们的聚类性能。由于数据集的正确分类已知，而这里比较的 4 个算法都要求输入聚类数，即算法得到的聚类结果中簇的个数与正确类数相同，所以使用错误率来比较它们的性能，实验结果如表 2 所列。

表 2 人工数据集 Dataset1 上的实验结果

算法	错误率
SBAC	0.11111
AUTOCLASS	0
k-prototypes	0.2103(α=0.2)
	0.2161 (α=1)
	0.2722(α=1.2)
CBEST	0.0325 (α=0.2)
	0.1803 (α=1)
	0.1833(α=1.2)

从表 2 的实验结果可以看出，由于数据分布与 AUTOCLASS[5] 算法的假设相符，因此 AUTOCLASS[5] 算法在此数据集上得到了最低的错误率 0。我们的算法 CBEST 在 $\alpha=0.2$时，也取得了很低的错误率 0.0325。在此数据集上 SBAC[7] 算法的性能比 k-prototypes[6] 更好一点。并且从表 2 我们也发现 α 参数对 k-prototypes[6] 算法似乎没有什么影响，而我们的算法 CBEST 却对参数 α 的设定很敏感，在此数据集中，由于数值型属性并没有被污染，而只是概念型属性有噪声，因此在 α 值比较小时，算法的性能更好，也就是给予数值型属性相似性更大的权重时算法取得了更低的错误率。

2. 人工数据集 Dataset2 上的比较实验

此人工数据的生成方法与第一个人工数据一样，只是在这个数据集中，我们按 0.8 的概率来扰乱第一个概念型属性的值，以模拟真实数据中存在的不正确值或异常点。我们还是报告了各个算法的错误率来比较它们的性能，实验结果如表 3 所列。

表 3 人工数据集 Dataset2 上的实验结果

算法	错误率
SBAC	0.36667
AUTOCLASS	0.0056
k-prototypes	0.2319 (α=0.2)
	0.2364 (α=1)
	0.1944 (α=1.2)
CBEST	0.0286 (α=0.2)
	0.5389 (α=1)
	0.5889 (α=1.2)

从表 3 的实验结果可以看出，与人工数据集 Dataset1 上的实验结果相似，由于数据分布与 AUTOCLASS[5] 算法的假设相符，因此 AUTOCLASS[5] 算法在此数据集上得到了最低的错误率 0.0056。我们的算法 CBEST 在 $\alpha=0.2$ 时，也取得了很低的错误率 0.0286。在此数据集上 SBAC[7] 算法的性能比 k-prototypes[6] 更差。同样从表 3 我们也发现 α 参数对算法 k-prototypes[6] 似乎没有什么影响，而我们的算法 CBEST 却对参数 α 的设定很敏感。在此数据集中，由于数值型属性并没有被污染，而只是概念型属性有噪声，因此在 α 值比较小时，算法的性能更好，而又因为此数据集比 Dataset1 对概念型数据污染程度更高，所以当 α 参数调高后，也就是加重概念型属性的权重后，算法的性能急剧下降。

3. 人工数据集 Dataset3 上的比较实验

我们人工生成了有 3 个类，每个类 100 个对象，共 300 个

四维对象的人工数据集 Dataset3，其中前二维是数值型属性，后两维是概念型属性。我们用二维高斯分布生成两个数值型属性，产生两个数值型属性的多维高斯分布的均值和协方差分别是：第一个类 mu = [6 3]，sigma = [12 3;3 1]；第二个类 mu = [4 9]，sigma = [11 3;3 1]；第三个类 mu = [4 6]，sigma = [11 3;3 1]。两个数值型属性形成的形状如图 3 所示。然后再生成了两个概念型属性，其中一个概念型属性每个类中被分别赋于不同的两个值，另一个概念型属性每个类赋于不同的一个值。然后我们按 0.3 的概率来扰乱第一个概念型属性的值，以模拟真实数据中存在的不正确值或异常点。

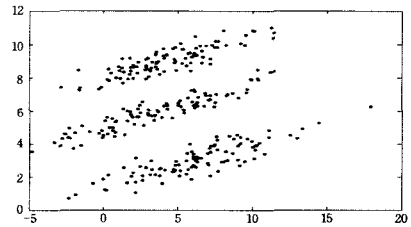

图 3　人工数据集 Dataset3 的两个数值型属性分布图

我们还是报告了各个算法的错误率来比较它们的性能，实验结果如表 4 所列。

表 4　人工数据集 Dataset3 上的实验结果

算法	错误率
SBAC	0.41
AUTOCLASS	0.0033
k-prototypes	0.6102 (α=0.2)
	0.5933 (α=1)
	0.6275 (α=1.2)
CBEST	0.0033 (α=0.2)
	0.2733 (α=1)
	0.2903 (α=1.2)

从表 4 的实验结果可以看出，在此数据集上我们的算法 CBEST 在 α=0.2 时得到了与 AUTOCLASS[5] 算法一样低的错误率 0.0033。SBAC[7] 算法和 k-prototypes[6] 在此数据集上性能很不好，特别是 k-prototypes[6]。这其中的原因可能是由于数值型属性呈线状分布，并没有呈球状，而 k-prototypes[6] 是基于 k-means 方法的。同样从表 4 我们发现 α 参数对算法 k-prototypes[6] 似乎没有什么影响，但综合前面的实验可以看出，它对数值型属性分布很敏感。在此数据集中，由于数值型属性没有被污染，概念型属性破坏程度也不高，因此在 α 值比较小时，我们的算法 CBEST 性能很好，α 值比较大时，性能也比 BAC[7] 算法和 k-prototypes[6] 好得多。

4. 真实数据集 the German credit dataset 上的比较实验

此数据集来自于 UCI Machine Learning Repository[18]，它是 Statlog Project Databases 中的一个数据集，有 7 个数值型属性和 13 个概念型属性，两个类。它本来有 1000 个数据点，我们从中随机选择了 300 个点用于我们的实验中。

我们还是报告了各个算法的错误率来比较它们的性能，此数据集上的实验结果如表 5 所列。

表 5　真实数据集 the German credit dataset 上的实验结果

算法	错误率
SBAC	0.3
AUTOCLASS	0.26667
k-prototypes	0.2833 (α=0.2)
	0.2833 (α=1)
	0.2833 (α=1.2)
CBEST	0.4197 (α=0.2)
	0.3553 (α=1)
	0.3643 (α=1.2)

从表 5 的实验结果可以看出，AUTOCLASS[5] 算法得到了最低的错误率 0.26667，其次是 k-prototypes[6]。但是从表 5 也可以看出所有的算法在此数据集上的效果都不是很理想，我们的算法在此数据集上表现最差。

5. 真实数据集 Australian Credit Approval 上的比较实验

此数据集来自于 UCI Machine Learning Repository[18]，它也是 Statlog Project Databases 中的一个数据集。Australian Credit Approval 是信用卡应用数据，为了数据的保密，所有的属性名和值都转换成了无意义的符号。此数据集包含了 6 个数值型属性和 8 个概念型属性，它是一个很有代表性的混合型数据集。它有两个类，一共 690 个对象点。

我们还是报告了各个算法的错误率来比较它们的性能，此数据集上的实验结果如表 6 所列。

表 6　真实数据集 Australian Credit Approval 上的实验结果

算法	错误率
SBAC	0.4841
AUTOCLASS	0.3623
k-prototypes	0.4006 (α=1)
	0.4007 (α=1.2)
	0.4009 (α=2.7)
CBEST	0.2194 (α=1)
	0.1812 (α=1.2)
	0.1799 (α=2.7)

从表 6 的实验结果可以看出，在此数据集上我们的算法 CBEST 在 α=2.7 时得到了最低的错误率 0.1799，远远好于其他 3 种算法。在此数据集上，AUTOCLASS 算法和 k-prototypes，SBAC 算法效果都不理想，特别是 SBAC[7] 算法。同样从表 6 我们也发现 α 参数对算法 k-prototypes[6] 似乎没有什么影响，而我们的算法 CBEST 却对参数 α 的设定很敏感，但 α 设定的不是很小时，我们的算法 CBEST 始终保持着比较好的性能，好于其他 3 种算法。

6. 真实数据集 the heart-disease/cleve 上的比较实验

此数据集来自于 UCI Machine Learning Repository[18]，是 the heart-disease 中的一个数据集。它也是一个比较典型的混合型数据集，有 7 个概念型属性和 6 个数值型属性，及一个类标签属性。cleve 数据集包含两个类，一类是没有心脏病，一类是有不同程度的心脏病。数据集中包含一些缺失数据，我们去掉缺失数据后得到一个有 296 个对象点的数据集。

我们还是报告了各个算法的错误率来比较它们的性能，此数据集上的实验结果如表 7 所列。

表 7　真实数据集 the heart-disease/cleve 上的实验结果

算法	错误率
SBAC	0.4966
AUTOCLASS	0.1182
k-prototypes	0.3806 (α=0.2)
	0.3770 (α=1)
	0.3818 (α=1.2)
	0.3794 (α=2)
	0.3758(α=3)
	0.2742 (α=0.2)

(下转第 274 页)

[10] Reniers D, van Wijk J J, Telea A. Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure [J]. IEEE Transactions on Visualization and Computer Graphics, 2008, 14(2): 355-368
[11] Ruberto D C. Recognition of shapes by attributed skeletal graphs[J]. Pattern Recognition, 2004, 37(1): 21-31
[12] Zhang Dengsheng, Lu Guojun. Review of Shape Representation and Description Techniques [J]. Pattern Recognition, 2004, 37: 1-19
[13] Wang J Z, Li Jia, Wiedrhold G. SIMPLICITY: Semantics-Sensitive Integrated Macthing for Picture Libraries [J]. IEEE Trans. Pattern Analysis and Machine Intelligence, 2001, 23(9): 947-963
[14] Iqbal Q, Aggarwal J K. Applying Pecepture Grouping to Content-based Image Retrieval: Building images [C]// IEEE Workshop on Content-based Aeeess of Image and Video Libraries. Fort Collins, CO, 1999: 42-48

(上接第 238 页)

CBEST	0.2017 (α=1)
	0.2086 (α=1.2)
	0.1926 (α=2)
	0.1892 (α=3)

从表 7 的实验结果可以看出，在此数据集上 AUTOCLASS[5] 算法得到了最低的错误率 0.1182，我们的算法 CBEST 在 α=3 时也得到了仅比 AUTOCLASS[5] 差一点的性能，错误率为 0.1892。SBAC[7] 算法和 k-prototypes[6] 在此数据集上性能很不好，特别是 SBAC[7] 算法。总体来说，在此数据集上，我们的算法 CBEST 性能较好，在各种 α 值情况下，性能都比 SBAC[7] 算法和 k-prototypes[6] 好得多。

4 CBEST 算法小结

我们的算法 CBEST 既不像 AUTOCLASS[5] 算法一样对数据的分布做出假设；也没有像 SBAC[7] 算法那样对相似性的计算过于复杂，并强加一种信息在其中，也就是认为频率低的属性值匹配应赋予更重的权重，当这与实际数据不匹配时，效果很不好，从我们的实验中可以看到这一点。对于 k-prototypes[6] 算法来说，它的效率高，但效果一般。针对 k-prototypes[6] 算法用于调节概念型属性和数值型属性权重比例的参数似乎对算法的性能没有影响，也就是说这个参数起不到调节作用，而且当数值型属性的分布不呈现球状分布时，它的效果很不好。我们的算法 CBEST 则具有以下特点：

1. 它没有强加给数据任何的分布模型，没有假设数值型数据是高斯分布，也没有假设概念型属性是多项式分布；

2. 它没有在相似性度量中强加任何的信息和偏向(bias)；

3. 它能有效融合先验知识，也就是说，对于概念型属性和数值型属性间的偏移，可以通过调节参数 α 来实现。当数值型属性足以区分不同的簇，而概念型属性的区分能力有限时，α 参数设置小一点，反之设置大一点。我们如果有这方面的先验信息时，可以将此先验信息融合到此参数中，当没有这方面的先验信息时，也可以通过多设置一些参数来得到不同的聚类结果，再从中择优或进行集成，从而得到一个理想而又稳定的效果。

结束语 早期的聚类算法和准则函数不适于处理混合型数据，所以我们专门针对混合型数据聚类进行了研究。在综述了混合型数据聚类算法的国内外研究现状后，我们提出了一种基于集成技术的混合数据聚类算法 CBEST。它利用聚类集成技术产生混合数据间的相似性矩阵，这就避免了直接计算混合型数据间相似性的难题，最后基于此相似性矩阵，应用谱聚类算法得到混合数据聚类结果。我们用了大量的实验来验证了它的有效性，及对噪声的鲁棒性，并且其能有效融合先验知识，通过参数的调节可以敏感地在不同属性间偏移。

由于 CBEST 是基于互联合矩阵或者说相似性矩阵上的聚类算法，它的时间复杂性是二次的。所以我们进一步的工作包括改进此算法的时间复杂性，并且使其能自动确定类个数。

参 考 文 献

[1] Huang Z. Extensions to the k-means algorithm for clustering large data sets with categorical values [J]. Data Mining and Knowledge Discovery, 1998, 2(3): 283-304
[2] Fisher D H. Knowledge acquisition via incremental conceptual clustering [J]. Machine Learning, 1987, 2(2): 139-172
[3] Mckusick K B, Thompson K. COBWEB/3: A portable implementation [R]. FIA-90-6-182. Moffett Field, CA: NASA Ames Research Center, 1990
[4] Reich Y, Fenves S. The formation and use of abstract concepts in design Concept Formation; Knowledge and Experience in Unsupervised Learning [M]. Los Altos, CA: Morgan Kaufmann, 1991: 323-353
[5] Cheeseman P, Stutz J. Bayesian classification (AutoClass): Theory and results Advances in Knowledge Discovery and Data Mining [M]. AAAI Press/The MIT Press, 1996: 153-180
[6] Huang Z. Clustering Large Data Sets with Mixed Numeric and Categorical Values [C] // Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining, (PAKDD). Singapore, 1997: 21-34
[7] Li C, Biswas G. Unsupervised Learning with Mixed Numeric and Nominal Data [J]. IEEE Trans. Knowl. Data Eng., 2002, 14(4): 673-690
[8] Goodall D W. A New Similarity Index Based On Probability [J]. Biometrics, 1966, 22: 882-907
[9] He Z, Xu X, Deng S. Clustering Mixed Numeric and Categorical Data: A Cluster Ensemble Approach [Z]. eprint arXiv: cs/0509011 2005
[10] Fred A L N. Finding Consistent Clusters in Data Partitions[C]// Multiple Classifier Systems, Second International Workshop, MCS 2001. Cambridge, UK, 2001: 309-318
[11] Ng A Y, Jordan M I, Weiss Y. On Spectral Clustering: Analysis and an algorithm [C]// NIPS. Vancouver, British Columbia, Canada: MIT Press, 2001: 849-856
[12] Topchy A, Jain A K, Punch W. A Mixture Model for Clustering Ensembles [C]// Proc. SIAM Conf. on Data Mining. 2004: 379-390
[13] Fred A L N, Jain A K. Data Clustering using Evidence Accumulation [C]// Proc. of the 16th Intl. Conference on Pattern Recognition ICPR 2002. Quebec, Canada: IEEE Computer Society, 2002: 276-280
[14] He Z, Xu X, Deng S. A cluster ensemble method for clustering categorical data [J]. Information Fusion, 2005, 6: 143-151
[15] Verma D, Meila M. A comparison of spectral clustering algorithms [Z]. university of washington, uw-cse-03-05-01, 2003
[16] Zelnik-Manor L, Perona P. Self-Tuning Spectral Clustering [C]// Eighteenth Annual Conference on Neural Information Processing Systems, (NIPS). 2004
[17] Kuhn H W. The hungarian method for the assignment problem [J]. Naval Research Logistics, 1955, 2(2): 83-97
[18] http://www.ics.uci.edu/~mlearn/databases/