# 基于聚类的属性约简方法①

## A Method of Attribute Reduction Based on Clustering

陈 源 曾德胜 (罗定职业技术学院 电子信息系 广东 罗定 527200)

谢 冲 (深圳华为技术有限公司 广东 深圳 518000)

**摘 要:** 针对现有的属性约简方法在约简的过程中与用户交互过程太少的问题，提出了属性距离的定义及其基于聚类的约简方法。首先给出了属性依赖度和相对依赖度的定义，然后根据用户给定参数和由属性相对依赖度计算出的属性距离对属性进行聚类，将区分能力相似的属性聚集到同一个类中，最后从每个类中选取出属性组成约简属性集。实验结果表明：该方法比以往的属性约简方法有更好的交互性能，能通过用户的参数，约简出接近用户需求的属性集。

**关键词:** 数据挖掘 属性约简 聚类 属性距离

## 1 引言

属性约简是一项很有意义的工作，一方面通过对数据库的属性约简可以为后面的数据挖掘和知识发现工作减少工作量，另一方面属性约简可以为用户决策提供重点和参考依据。作为数据挖掘的前期工作，属性约简得到的结果也应该尽量满足用户的兴趣。不同用户有不同的要求：有些用户要求属性子集的区分能力一定要达到 100%，约简属性集的属性个数在一定范围即可；有些用户要求最后得到的约简属性子集中的属性个数尽可能的少，但是属性子集的区分能力不用达到 100%只要在用户认可的范围内即可。例如，某个数据库有 100 条属性，如果要得到区分能力为 100%的约简属性集，那么可能最后只能得到一个 20 个属性的属性集。可是如果用户对最后约简结果的区分能力放松一些，可能用户就能得到一个有含 10 个或者数量更少的约简集。那么对于那些对区分能力要求不太严格，但是希望能得到一个数量比较少的约简属性集的用户来说就会采用这种约简的方式；有些用户想通过属性约简找出一些平时单个区分能力不强，但是合在一起以有很强的区分能力的属性的集合。许多属性虽然单个的区分能力很强但是都是一些“大众化”的属性，也就是大家都知道认可的属性。用户要得到这样的属性明显作用不大，所以不少用户想得到一些“例外的”属性。例如：网上商店的 VIP 顾客群的数据中，有客户 ID，姓名，年龄，电话等等属性。如果依据单个属性的区分能力来进行属性约简，可能得到的会是“客户身份证号码”或是“客户 ID”等这些表示类型的属性，或者另外一些区分能力比较强的属性。这些属性单个区分能力都很强，但是对用户来说因为都是众所周知的属性所以用处不大。如果从属性之间的联系出发来进行属性约简就能找到一些“例外属性”：例如“年龄、籍贯、学历”这样一个属性集。虽然这个属性集里的每个属性单个区分能力都不强但是合在一起就能达到一个比较强的区分能力，而且这样的约简属性集是用户比较感兴趣的。

对于用户的这些要求，我们提出了基于聚类的属性约简方法，提供了用户交互的平台。该方法利用属性之间区分能力的相似性，将属性进行聚类最后选取约简属性子集，并且可以根据用户不同的要求而制定出的参数有选择的对数据库的属性进行聚类。这样最后选择出来的属性就比较符合用户的需求。该方法更加注重了属性之间的依赖，和与用户的交互性。实验结果表明，该方法效果较好。

① 基金项目:国家自然科学基金项目(60463003)
收稿时间:2008-10-15

## 2　基于聚类的属性约简方法

### 2.1 基本概念

我们根据属性间的距离来对属性进行聚类，使得区分能力相似的属性聚在一起。在数据库的属性中，有一些属性的区分能力比较相似，可以先通过聚类将这些属性聚集在一个类里，然后只要从每一类中选出一个这一类属性的代表即可，其他的属性都可以约简掉，下面先给出属性间依赖关系的定义。

定义 1．属性间的依赖关系

给定一个信息系统(数据库)(U,C,D,F)，其中 U 为对象集，即 U={x1,x2, ... ,xn}. U 中的每个 xi(i=1,2,...,n),称为一个对象。C 为条件属性集，即 C={c1,c2,...,ck}, C 中的每个 Cj(j=1,2,...,k),称为一个条件属性。D 为决策属性，F 为 U 和 C、D 的关系集，fCj (xi)为对象 xi 的 Cj 的属性值。

现有两个条件属性 Cm,Cn 和两个决策属性值不同的对象 x，y，函数 $tag(x,y)=\begin{cases}1 & if \quad x\neq y\\ 0 & if \quad x=y\end{cases}$，如果 $tag(f_{c_m}(x),f_{c_m}(y))=0$ 则称属性 $C_m$ 无法区分对象 x，y，反之称其为可区分

$\sum_{x,y\in U,f_D(x)\neq f_D(y)}(1-tag(f_{c_m}(x),f_{c_m}(y)))$，则称为属性 $C_m$ 在区分对象集 U 中两两对象(决策属性值不同的两个对象)时出现的不可区分的次数。如果两两对象是针对决策属性值不同的两个对象，而两个对象决策属性值相同，不管出现什么约简结果都不会对它们有影响所以就没有必要讨论了。如果 $tag(f_{c_m}(x),f_{c_m}(y))=0$ 而 $tag(f_{c_n}(x),f_{c_n}(y))=0$，属性 $C_m$ 不能区分对象 x，y 而属性 $C_m$ 可以，这样 $C_m$ 只要依赖属性 $C_n$ 就能将对象 x，y 区分开，此时称属性 $C_m$ 对属性 $C_n$ 产生一个依赖关系。

定义 2．属性依赖度

属性 $C_m$ 对属性 $C_n$ 的依赖度则定义为属性 $C_m$ 对属性 $C_m$ 在区分对象集 U 中两两对象时产生依赖关系次数的总和，计算公式如下：

$$R(c_m,c_n)=\sum_{x,y\in U,f_D(x)\neq f_D(y)}tag(f_{c_n}(x),f_{c_n}(y))\times tag(tag(f_{c_m}(x),f_{c_m}(y)),tag(f_{c_n}(x),f_{c_n}(y))) \quad (1)$$

定义 3．属性相对依赖度

属性 $C_m$ 对属性 $C_n$ 的相对依赖度定义为属性 $C_m$ 对属性 $C_n$ 在区分对象集中两两对象时产生依赖关系次数的总和与属性 $C_m$ 区分对象集两两对象出现不可区分次数的比，计算公式如下。

$$CR(c_m,c_n)=\frac{\sum_{x,y\in U,f_D(x)\neq f_D(y)}tag(f_{c_n}(x),f_{c_n}(y))\times\omega}{\sum_{x,y\in U,f_D(x)\neq f_D(y)}(1-tag(f_{c_m}(x),f_{c_m}(y)))} \quad (2)$$

$$\omega=tag(tag(f_{c_m}(x),f_{c_m}(y)),tag(f_{c_n}(x),f_{c_n}(y)))$$

定义 4．属性向量

给定属性 $C_m$，因为数据库中每个属性对 $C_m$ 都会有影响，而他们的影响就是 $C_m$ 对它们的依赖程度，所以我们定义 $C_m$ 的向量值就是 $C_m$ 对其他属性的相对依赖度，即属性 $C_m$ 的属性向量为：

$c_m(R(c_m,c_1),R(c_m,c_2),R(c_m,c_3),...,R(c_m,c_k))$，其中，k=card(C)。

定义 5．属性距离

我们运用经典欧式距离公式来计算属性间的距离

$$S(c_m,c_n)=\sqrt{(R(c_m,c_1)-R(c_n,c_1))^2+\theta} \quad (3)$$

$$\theta=(R(c_m,c_2)-R(c_n,c_2))^2+...+(R(c_m,c_k)-R(c_n,c_k))^2$$

其中，k=card(C)

从公式(3)可以看出如果两个属性对其他属性的相对依赖度越相似，那么它们之间的距离就越小。如果两个属性的属性距离比较大，这就说明这两个属性与其他属性的相对依赖度相差比较大，同时这两个属性间的相对依赖度也比较大。因为如果两个属性的相对依赖度越小，就说明它们的区分能力越相似，而且与其他属性的相对依赖度也越相似。

### 2.2 属性聚类

对属性进行聚类的目的是要将区分能力相似的属性聚在一个类里面，也就是一个类就是类中所有属性的代表，是这一类型属性的代表。而类与类之间的距离，实际上就是这两个类所代表的两种类型的属性之间的距离。根据类间的距离，运用聚类算法对属性进行聚类。将所有属性聚成若干类，使得同类属性间属性距离较小，而不同类的属性间属性距离较大。然后再从每个类中选取具有代表性的属性组成约简属性集。

为了保证类内属性间的属性距离较小，我们采用

的聚类方法是最小距离聚类法。首先将每个属性都看成一个类，而每个属性间的属性距离就是这些类的类间距离。聚类时，每次都寻找类间距离最小的两个类，如果当前最小类间距离小于聚类系数，则将这两个类合并成一个新类，并且重新计算新类与其它类的类间距离。然后再重复上面的工作：寻找最小类间距离、判断、合并、重算类间距离。直到寻找到的最小类间距离大于给定的聚类系数，工作停止，聚类完成。

### 2.3 属性的选取

通过对属性的聚类我们将属性聚成了类。同类的属性都有区分能力相似，相互依赖度较低的特点，从每个类中只挑选出一个属性作为这个类中属性的代表来组成最后的约简属性集。为了保证约简出来的属性集的区分能力尽可能的强，在选取属性时要保证选取出来的属性之间的距离较大。

文中采用的是组合法选取属性。从每个类都选取出一个属性组合成属性集，然后计算这个属性集中属性之间距离的总和。最后选取属性间距离总和最大的一组属性集，作为聚类后选取出来的约简属性集。

### 2.4 算法设计

算法 ARBCA

Input： D：数据文件；veracity：用户希望约简属性集达到的精确度；step：用户希望的聚类步数

Output：FS：最后约简出的属性集

S1:用公式(1)计算各个属性相互之间的依赖度；用公式(2)计算各个属性之间的相对依赖度；用公式(3)计算属性之间的距离。

S2:初始化聚类条件：聚类上限 up=1，下限 low=0；run=0。

S3:对属性集进行聚类系数为(up +1ow)/2 的聚类：cluster((up +1ow)/2，distance)

S4:使用组合选取法，对聚类结果选取属性组成约简属性集 FS。

S5:计算约简属性集的分类准确率 V。

S6: 判 断 ： 如 果 V >veracity ， low = (up+1ow)/2；

if (run>0) run =run +1； else run =0；

如果 V<veracity，up = (up+1ow)/2；

if (run<0) run=run-1； else run=0；

S7:判断如果|run|=step 则输出 FS ,否则转到第 S3 步。

## 3 实验

在实验过程中，算法采用的编程语言为 VC++，系统开发环境为 VC++6.0，CPU 为 intel Pentium 4，2.4GHZ，内存为 256M。实验所用的真实数据来自：http://www.ics.uci.edu/mlearn/MLSummary.html/。实验中使用的数据和下载的数据存储方式一样，为文本文件方式存储。而合成数据集是利用网址 http://www.cse.cuhk.edu.hk/~kdd/data/IBM_VC++.zip 上提供的数据生成器生成的。

### 3.1 实验 1

为了证明基于聚类的属性约简方法的有效性和与用户的交互性，我们在真实数据集上进行了实验。实验选择的数据及数据处理设置如下：

数据集来源： http://www.ics.uci.edu/mlearn/MLSummary.html/

选择的数据集：Zoo Database（动物数据库），该数据库中的数据是分类数据。

数据集描述：该数据库中记录了 101 个动物实例以及它们的 18 个属性(animal name，15 boolean attributes，2 numeric attributes)。我们去掉了 animal name 属性。因为每个动物的名字都是唯一的，所以这个属性保留没有什么意义。类别 1-7 代表的动物分别显示在表 1 中。

表 1 数据集中动物的类别

| 类别号 | 动物个数 | 所属类别 |
|---|---|---|
| Type 1 | 41 | Mammal(哺乳动物) |
| Type 2 | 20 | Aves(鸟类) |
| Type 3 | 5 | Creeping animal(爬行动物) |
| Type 4 | 13 | Fish(鱼类) |
| Type 5 | 4 | Amphibian(两栖动物) |
| Type 6 | 8 | Insect(昆虫类) |
| Type 7 | 10 | Arthropod(节肢动物) |

16 个属性的名称分别为：hair、feathers、eggs、milk、airborne、aquatic、predator、toothed、backbone、breathes、venomous、fins、legs、tail、domestic、catsize。我们就用属性的顺序号代替属性名称。

我们在用户输入不同的准确度和不同的聚类步数下做了实验。一般用户要求的准确度不会太低，我们分别用了 0.98，0.99 和 1 三个不同的准确度要求来

进行实验。而且分别对聚类步数 2，3，4 进行了实验。实验的聚类效果如下：

表 2　实验 1 的实验结果

| 用户定义 | | | | |
|---|---|---|---|---|
| 准确率 | 聚类步数 | 约简结果 | 个数 | 准确率 |
| 0.98 | 2 | 4、6、8、13、14、16 | 6 | 0.998218 |
| | 3 | 4、6、8、13、16 | 5 | 0.998013 |
| | 4 | 4、6、13 | 3 | 0.992673 |
| 0.99 | 2 | 4、6、8、13、14、16 | 6 | 0.998218 |
| | 3 | 4、6、8、13、16 | 5 | 0.998013 |
| | 4 | 4、6、13 | 3 | 0.992673 |
| 1.00 | 2 | 2、4、6、7、8、10、11、13、14、16 | 10 | 1.00 |
| | 3 | 2、4、6、7、8、11、13、14、16 | 9 | 1.00 |
| | 4 | 4、6、8、10、11、13、14 | 7 | 1.00 |

从表 2 中，我们可以看出在保证达到用户要求的准确率的前提下聚类步数越大，最后约简出来的属性集里的属性个数越少，属性集越精简。如果用差距矩阵方法对数据库 ZOO 进行约简得到的结果是 3、4、6、8、13、14 这与聚类准确率为 1.00，聚类步数为 4 的聚类约简结果差不多。对比而言，使用聚类属性约简方法更加灵活，可以就用户的需求给出合适用户要求的约简结果。

### 3.2 实验 2

在这个实验中，我们选取了 Flag 数据库,该数据库有 200 条记录，记录了各个国家国旗的特征，其中包括：zone、area、population、language、religion 等 29 条属性，其中 landmass 是类别属性。同样也用了不同的准确度和不同的聚类步数做了实验。

从表 3 中，我们同样可以看出在保证达到用户要求的准确率的前提下聚类步数越大，最后约简出来的属性集里的属性个数越少，属性集越精简。

表 3　实验 2 的实验结果

| 用户定义 | | | | |
|---|---|---|---|---|
| 准确率 | 聚类步数 | 约简结果 | 个数 | 准确率 |
| 0.99 | 2 | 2、6、8、13、14、16 | 6 | 0.998218 |
| | 3 | 2、8、13、16 | 4 | 0.996071 |
| | 4 | 2、8 | 2 | 0.995460 |
| 1.00 | 2 | 1、2、3、4、5、8、16、27、28 | 9 | 1.00 |
| | 3 | 2、3、4、5、8、16、27、28 | 8 | 1.00 |
| | 4 | 2、3、4、8、16、27、28 | 7 | 1.00 |

实验结果表明，通过对用户参数的调整，约简出来的结果会不断向用户需求的方向进行变化。基于聚类的属性约简方法它的思想完全不同于传统的属性约简方法，它是一种面向用户的约简方法，用户可以根据自己的需要去制定约简的约束条件，得到的结果更加符合用户的要求。现有的属性约简方法很多都是以在知识不减少的情况下约简属性集的个数多少为评价标准。可随着研究的发展，我们对属性约简的结果要以用户的要求和满意来进行评估。一个约简属性集就算它的数据再好，如果不是用户想要的那种属性集合，那它就是一个不合格的属性集。这里的用户不完全是指在市场上的用户或是政府部门，而是多种多样的。

## 4　小结

在信息时代，数据挖掘起到了越来越大的作用，应用数据挖掘技术的用户也会来自各行各业不同的层次。因此，用户的需求和参与也是进行数据挖掘必需考虑的因素。本文提出了基于聚类的属性约简方法。

## 参考文献


1 朱明.数据挖掘.北京:中国科学技术大学出版社, 2002.

2 Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. Proceedings of the 2000 ACM-SIGMOD International Conference Management of Data. Washington, 2000:1－12.

3 Miao DQ, Wang J. Analysis on Attribute Reduction Strategies of Rough Set. Chinese Journal of Computer Science and Technology, 1998,13(2):189－192.

4 夏文克,刘明霄,张志伟.基于属性相似度的属性约简算法.河北工业大学学报,2005,36(4):50－52.

5 刘靖,陈福生.结合粗糙集和模糊聚类方法的属性约简算法.计算机应用与软件, 2004,11(21):24－25.