计算机科学1999Vol. 26№. 8

从数据库中发掘定量型关联规则*)

Mining Quantitative Association Rules from Database

梁曼君 张 瑞 熊范纶

(合肥工业大学计算机系 合肥230009) (中科院合肥智能所 合肥230031)

Abstract In this paper, we introduce the technique of mining quantitative association rules in KDD. We apply it on an agriculture database to find some unknown useful knowledge.

Keywords Association rule, Quantitative association rule, Frequent itemset

一、引言

随着数据库技术和机器学习技术的发展,在数据库中发现新颖的、具有潜在效用的知识,简称KDD(Knowledge Discovery in Database)是近年来的一个新兴研究领域。KDD中的关联规则是描述数据库中数据项(属性,变量)之间所存在的(潜在)关系的规则,我们作如下形式化定义:

令 $I=\{i_1,i_2,\cdots,i_m\}$ 为项目集(itemset),D为事务数据库,其中每个事务T是一个项目子集($T\subseteq I$),并具有一个唯一的标识符ID。关联规则是形如 $X\Rightarrow Y$ 的逻辑蕴含式,其中 $X\subset T$、$Y\subset T$、且 $X\cap Y=\phi$。有两个因子与这条规则相关:如果事务数据库中有s%的事务包含 $X\cup Y$,那么我们说关联规则 $X\Rightarrow Y$ 的支持度(support)为s;如果事务数据库里包含X的事务中有c%的事务同时也包含Y,那么我们说关联规则 $X\Rightarrow Y$ 的置信度(confidence)为c。

从关联规则的定义我们可以知道,发掘关联规则问题可以被看作是在一个所有属性均为布尔类型的关系表中寻找"1(T)"值之间的关联。在关系表的一个记录中,某个属性的值为"1(T)"则表示在相应的事务中包含了相应的项目,否则属性值为"0(F)"。在这种情况下发现关联规则称为布尔型关联规则问题。但是,在绝大多数商业及科学领域中,属性的类型是多种多样的。当属性都是定量型(如:年龄、价格)或离散型(如:城市规模、商品种类),则在这些属性中发现关联规则问题就被称为定量型关联规则问题(这里我们把定量型属性和离散型属性统称为定量型属性)。下面我们首先介绍一下发现布尔型关联规则的算法。

二、布尔型关联规则的发现算法

由于发现关联规则的目的在于找出那些可信的并具有代表性的规则,所以要给定一个最小支持度阈值和一个最小置信度阈值。发现关联规则问题就是发现所有支持度和置信度均分别超过规定阈值的关联规则,这个发现过程可分为两步:第一步是识别所有的频繁项目集(简称频繁集),即所有支持度不低于用户规定的最小支持度阈值的项目集;第二步是从第一步得到的频繁集中构造置信度不低于用户最小置信度阈值的规则。其中第一步是关联规则发现算法的核心,也是计算量最大的部分,因为如果有m个项目,那么就有 2^m 个可能的频繁集,而事实上只有其中的一小部分确实是频繁的。

下面我们介绍进行关联规则发现的Aprior算法。我们引入若干记号:具有k个项目的项目集称为k-项目集(k-itemset),同时我们称该项目集的长度为k;由k-项目集构成的集合称为k-项目序列集。L_k 记由频繁k-项目集构成的集合,C_k 记由候选k-项目集构成的集合。频繁项目集的发现方法是一种渐进的方法,即按照项目集的长度,从发现频繁1-项目序列集开始,逐次增加项目集的长度。具体如下:

首先遍历目标数据库一次,记录每个项目或属性的出现次数,即计算每个项目的支持,收集所有支持度不低于用户最低支持度阈值的项目构成频繁1-项目序列集 L_1,然后链接 L_1 中所有的元素对形成候

*)本课题得到国家自然科学基金资助。

选2-项目序列集 C_2，再次遍历目标数据库，计算 C_2 中每个候选2-项目集的支持，收集所有支持度不低于用户最低支持度阈值的2-项目集构成频繁2-项目序列集 L_2，再通过链接 L_2 中的所有元素对形成候选3-项目序列集 C_3，再次遍历目标数据库，计算 C_3 中每个候选3-项目集的支持，收集所有支持度不低于用户最低支持度阈值的3-项目集构成频繁3-项目序列集 L_3。反复执行上述过程，直到没有新的候选产生为止，该过程可作如下算法描述：

Aprior 算法

```
Begin
    (1)L1={frequent 1-itemsets};
    (2)for (k=2;Lk-1≠φ;K++){
    (3)   Ck=aprior_gen(Lk-1);
    (4)   for all transaction t∈D do {
    (5)   Ct=subset(Ck,t);
    (6)   for all candidate c∈Ct do c.count++;
    (7)      }
    (8)   Lk-1={c∈C.|c.count≥min_sup};
    (9)   }
    (10)Answer=∪kLk-1;
End.
```

其中，Aprior_gen 是以频繁(k－1)-项目序列集 L_{k-1} 为自变量的候选生成函数。该函数返回所有频繁 k-项目集的超集，分链接和修剪两步执行：

```
(1)链接(join)
insert into Ck
select p.item1, p.item2,…,p.itemk-1, q.itemk-1 Ck
from Lk-1 p, Lk-1 q
where p.item1=q.item1,p.item2=q.item2,…,p
      itemk-2=q.itemk-2,p.itemk-1<q.itemk-1;
(2)修剪(prune)
```

对 C_k 中的任一候选 c，如果 c 中存在一个不属于 C_{k-1} 的长度为 k－1的子序列，那么就从 C_k 中删除该候选 c。

```
for all itemsets c∈Ck do
    for all (k-1)-subset s of c do
        if (s∉Lk-1)then delete c from Ck;
```

三、定量型关联规则的发现算法

关于定量型关联规则问题，人们很容易想到的解决方法就是将其对应到布尔型关联规则问题上，即每个属性值都对应于一个新的布尔型属性，再在其上使用发掘布尔型关联规则的算法。但是如果所有属性都是离散型的或者定量型属性只取少数几种值，这种对应还比较直接明了；如果定量型属性取值范围很大，这种一一对应就显得不实用，就有必要对属性进行划分，再对应到布尔型属性上去。这样，发现定量型关联规则问题可以分三步进行：

(1)确定每一个定量型属性值域的区间划分；

(2)发现所有的频繁项目集；

(3)从所得的频繁项目集中构造关联规则。

针对农业数据库，我们对定量型属性的划分采用等距离划分的方法。例如，我们把"降水"(单位：毫米)这一定量型属性划分为七个区间，从而构成表1中所示的七个布尔型属性：

表1

降水	降水	降水	降水	降水	降水	降水
00～10	10～20	20～30	30～40	40～50	50～60	80～90
…	…	…	…	…	…	…

四、实验结果

我们根据我国某地区二十年来的小麦苗情资料和气象信息建立了一个农业数据库，在此基础上进行定量型关联规则的发现。数据库中的属性(字段)如表2所示，其中："节气"是指小麦生长期经历的"小雪"到"清明"十个不同节气名称；"苗数"是指每亩小麦的植株密度；"苗高"是指小麦的植株高度；"降水"是指该节气期间降水的毫米数；"日照"是指该节气期间的日照小时数。如果把这个数据库中的记录按照时间顺序来排列，该数据库内容就如表2所示。

表2

编号	节气	苗数	苗高	降水	日照
1	小雪	19.6	11.5	8.3	21.8
2	小雪	22.4	12.0	8.3	21.8
…	…	…	…	…	…
1	大雪	23.4	13.5	13.1	20.3
2	大雪	25.2	15.0	13.1	20.3
…	…	…	…	…	…

这里有两点需要说明：表2中的"编号"是测得数据的试验田块的编号；由于要发现具有共性的知识，我们在表2中省略了"年份"字段。

表2中除了"节气"是离散型属性外，其余属性均为定量型属性。我们把表2中的所有属性都转化为布尔型属性，结果如表3所示。

从表3中我们可以看到，所有的定量型属性(包括离散型属性)都被分解转化成布尔型属性，原来的一个属性现在对应于多个属性。由于现在所有的属性都是布尔型的，所以发现定量型关联规则问题就被转化成了发现布尔型关联规则问题。于是我们对表3所示的数据库运行发现布尔型关联规则的算法，就可以完成对所有关联规则的发现。

表3

编号	小雪	大雪	…	苗数 10～20	苗数 20～30	…	苗高 10～20	…	降水 00～10	降水 10～20	…	日照 20～30	…
1	T	F	…	T	F	…	T	…	T	F	…	T	…
2	T	F	…	F	T	…	T	…	T	F	…	T	…
…	…	…	…	…	…	…	…	…	…	…	…	…	…
1	F	T	…	F	T	…	T	…	F	T	…	T	…
2	F	T	…	F	T	…	T	…	F	T	…	T	…
…	…	…	…	…	…	…	…	…	…	…	…	…	…

在这个农业数据库上发现部分关联规则如下：

〈小雪〉⇒〈降水00～10〉
（6%support，100%confidence）
〈清明〉⇒〈日照60～70〉
（7%support，67%confidence）
〈苗高60～70〉⇒〈清明〉
（3%support，81%confidence）
〈苗高70～80〉⇒〈清明〉
（2%support，100%confidence）
〈苗数10～20〉⇒〈苗高00～10〉
（8%support，66%confidence）
〈苗高70～80〉⇒〈苗数30～40〉
（1%support，67%confidence）
〈降水80～90〉⇒〈春分〉
（3%support，100%confidence）
〈降水50～60〉⇒〈日照10～20〉
（3%support，100%confidence）
〈降水80～90〉⇒〈日照30～40〉
（3%support，100%confidence）
〈苗数10～20，日照20～30〉⇒〈小雪〉
（1%support，67%confidence）
〈苗高00～10，日照20～30〉⇒〈小雪〉
（2%support，78%confidence）
〈小雪，苗数10～20〉⇒〈苗高00～10，降水00～10〉
（2%support，67%confidence）

对上面的结果有一点需要说明：由于小麦的生长情况在不同节气中有不同的特点，而我们考察的数据库中包含了所有十个节气的数据信息，所以我们在进行关联规则的发现时，可以把最小支持度阈值适当地设小一点。

我们可以用同样的方法，对不同内容的农业数据库进行关联规则的发现，希望发现有用知识，并把它应用到农业生产中。例如我们发现关联规则：

〈小雪，苗数10～20〉⇒〈苗高00～10，降水00～10〉（2%support，67%confidence）

即发现这样的知识：在小雪这个节气，对每亩小麦植株数在10～20万的田块，其平均苗高在10厘米以下，并且这个节气的平均降水量在10毫米以下，对照当前情况，对于小雪时平均苗高超过10厘米的田块，就要控制氮肥的施用，以防止植株过高，不易过冬。上面发现的其它有关苗高、苗数与节气、日照、降水等因素的关联规则，也将有助于我们对农作物田间管理、施肥、施药等各方面进行有针对性的科学指导，并预测作物的生长与收成。

参考文献

1 Agrawal R，Imielinski T，Swami A. Mining Association Rules Between Sets of Items in Large Databases. In：Proc. of the ACM SIGMOD Conf. on Management of Data. Washington，1993

2 Agrawal R，Srikant R. Fast Algorithms for Mining Association Rules. In：Proc. of the 20th Intl. Conf. on Very Large Database. Santiago，Chile，1994

3 Savasere A，Omiecinski E，Navathe S. An Efficient Algorithm for Mining Association Rules in Databases. In：Proc. of the VLDB Conf. Zurich，Switzerland，1995

（上接第57页）

通过感知器和效应器直接与问题域相连接；问题域是复杂和动态的，要在有限的时间里动作；典型地，它们是自主的，其运作不需要人类的指导，不需要从用户那里得到显式的目标陈述；强调导致的系统行为、而不是知识；结果可能是突现的，不需要归于特定的内部结构；非常强调反应和自适应性；自下而上设计。

概括而言，基于知识的人工智能和基于行为的人工智能的区别在于：①焦点是显式符号知识还是导致的行为；②单一的高层活动还是多种低层活动；③用户驱动计算还是自主运作；④学习还是自适应。

智能影响行为，行为是智能的某种表现，基于行为的人工智能可能是对基于知识的人工智能所面临的困境的一种弥补。移动 Agent 是智能 和行为的统一。随着计算系统变得越发分布、互联和开放，智能 Agent 将成为一个关键技术，其意义将不断显露。由于多 Agent，尤其是移动 Agent 系统是不确定和难以预测的，系统的整体行为和性质只有在运行时才能体现，在设计阶段不能确定。所以，除了 Agent 的内涵需不断深入、明晰外，面向 Agent 的程序设计方法有待深入研究。（参考文献共25篇，略）