关联规则挖掘——Apriori 算法

2153726 罗宇翔

(经济与管理学院, 信息管理与信息系统)

摘　要

关联规则 (*Association　Rules*) 反映一个事物与其他事物之间的相互依存性和关联性，是数据挖掘的一个重要技术，用于从大量数据中挖掘出有价值的数据项之间的相关关系。

Apriori 算法是种挖掘关联规则的频繁项集算法，一种最有影响的挖掘布尔关联规则频繁项集的算法。

关键词：关联规则、*Apriori* 算法

目录

1 关联规则

关联规则挖掘是数据挖掘中最活跃的研究方法之一，最早由 $Agrawal$[1] 等人于 1993 年提出。当时是针对购物篮分析问题提出的，其目的是为了发现交易数据库中不同商品之间的联系规则。

关联规则由以下几个概念组成：

1. 项目 ($Items$): 交易数据库中的一个字段，对超市的交易来说一般是指一次交易中的一个物品。如：牛奶
2. 事务 ($Transactions$): 某个客户在一次交易中，发生的所有项目的集合。如 [牛奶，面包，啤酒]
3. 项集 ($Item \quad Sets$): 包含（一次事务中的）若干个项目的集合，一般会大于 0 个
4. 频繁项集 ($Frequent \quad Item \quad Sets$): 某个项集的支持度 ($Support$) 大于设定阈值（人为设定或者根据数据分布或者经验来设定），即称这个项集为频繁项集
5. 规则 ($Rules$): 若 X、Y 为项集，则称 $X \Rightarrow Y$ 为一条规则
6. 支持度 ($Support$): 见下文
7. 置信度 ($Confidence$): 见下文
8. 提升度 ($Lift$): 见下文

设事务集为 D，$count(X)$ 意为项集 X 在事务集 D 中出现的次数，$|D|$ 意为事务集 D 中共有多少条事务。则项集 X 的支持度 ($Support$) 定义为：

$$
\begin{aligned}
Support(X) &= P(X) \\
P(X) &= \frac{count(X)}{|D|} \\
count(X) &= \sum_{d_i \in D} appear(X, d_i) \\
appear(X, d_i) &= \begin{cases} 1, & \text{if } X \subseteq d_i \\ 0, & \text{else} \end{cases}
\end{aligned}
$$

进而可以定义规则 $X \Rightarrow Y$ 的置信度 ($Confidence$) 和提升度 ($Lift$) 的定义分别如下：

$$\begin{aligned} Confidence(X \Rightarrow Y) &= P(Y|X) \\ Lift(X \Rightarrow Y) &= Confidence(X \Rightarrow Y)/Support(Y) \\ &= P(Y|X)/P(Y) \end{aligned}$$

2 Apriori 算法

Agrawal 等人于 1993 年提出关联规则挖掘后，次年提出了挖掘关联规则的 *Apriori* 算法 [2]。

2.1 算法流程

Apriori 原理：如果某个项集是频繁的，那么它的任何子集也是频繁的。

证明如下：

$$\begin{aligned} &\text{设支持度阈值为}n\text{、事务集为}D \\ &\text{对一频繁项集}X\text{，有}Support(X) = P(X) \geq n \\ &\text{根据}P(X)\text{的定义，有}P(X) = count(X)/|D| \geq n \\ &\because \forall X_{sub} \subseteq X\text{且}X \subseteq d_i\text{，有}X_{sub} \subseteq X \subseteq d_i\text{，即}appear(X_{sub}, d_i) = 1 \\ &\text{又}\because \exists X_{sub} \subseteq X\text{且}X \nsubseteq d_i\text{，有}X_{sub} \subseteq d_i\text{，即}appear(X_{sub}, d_i) = 1 \\ &\therefore \forall X_{sub} \subseteq X, \quad appear(X_{sub}, d_i) \geq appear(X, d_i) \\ &\therefore \forall X_{sub} \subseteq X, \quad Support(X_{sub}) \\ &= P(X_{sub}) = count(X_{sub})/|D| = \sum_{d_i \in D} appear(X, d_i)/|D| \\ &\geq \sum_{d_i \in D} appear(X, d_i)/|D| \\ &= count(X_{sub})/|D| = P(X) = Support(X) \geq n \quad \square \end{aligned}$$

该定理的逆否命题为：如果一个项集是非频繁的，那么它的任何超集也是非频繁的。*Apriori* 算法根据这一定理对项集搜索过程进行剪枝，其流程如下：

1. 从空集开始，此时待搜索空间为总项集的幂集

2. 扫描事务集，搜索频繁 k 项集并将其从搜索空间中删去

3. 将非频繁的 k 项集以及它的超集从搜索空间中删去

4. k ← k+1

5. 重复步骤 2、3、4，直至搜索空间为空集

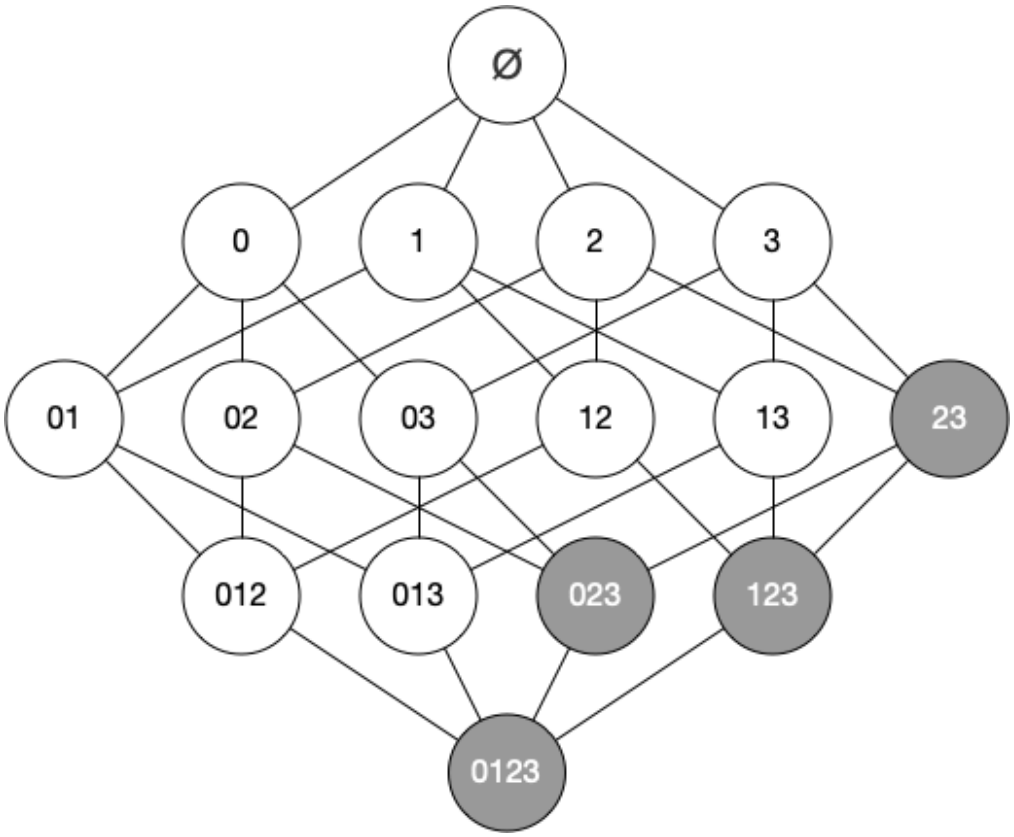

Figure 1: 频繁项集搜索过程

见上图。初始状态为空集，此时搜索空间为上图中从第二行直到最后一行的结点。

第一轮搜索为频繁 1 项集。

第二轮搜索为频繁 2 项集。此时非频繁的 2 项集 $\{2,3\}$ 以及它的超集 $\{0,2,3\}$、$\{1,2,3\}$ 和 $\{0,1,2,3\}$ 将从待搜索空间中删去。

第三轮搜索为频繁 3 项集。此时待搜索空间已为空集，算法结束。

2.2 算法复杂度

设事务集中共有 n 个项目、k 条事务。

时间复杂度:

最坏情况下，步骤 2、3、4 将重复 n 次。步骤 2 扫描一次事务集的时间复杂度为 $O(k)$，故算法时间复杂度为 $O(kn)$。

空间复杂度:

最坏情况下，算法将储存总项集幂集的所有元素，即空间复杂度为 $O(2^n)$。

3 总结

尽管 *Apriori* 算法的时空复杂度都相当高（分别为平方级和指数级），其作为关联规则挖掘的开山之作的功劳不容忽视。*Apriori* 算法的提出为后续关联规则挖掘的研究奠定了基础。

参考文献

[1] Agrawal R, Imieliński T, Swami A. Mining association rules between sets of items in large databases[C]//Proceedings of the 1993 ACM SIGMOD international conference on Management of data. 1993: 207-216.

[2] Agrawal R, Srikant R. Fast algorithms for mining association rules[C]//Proc. 20th int. conf. very large data bases, VLDB. 1994, 1215: 487-499.