西安交通大学
XI'AN JIAOTONG UNIVERSITY

数据挖掘
第三章：关联规则挖掘

刘均

陕西省天地网技术重点实验室

西安交通大学计算机学院

1 关联规则挖掘的基本概念

2 由事务数据库挖掘单维布尔关联规则

3 多层关联规则挖掘

4 多维关联规则挖掘

5 关联规则、相关性、因果关系的区别

基本要求：掌握关联规则、多层关联规则、多维关联规则的基本概念，掌握各种关联规则挖掘的算法

本章内容

3.1 关联规则挖掘的基本概念

3.2 由事务数据库挖掘单维布尔关联规则

3.3 多层关联规则挖掘

3.4 多维关联规则挖掘

3.5 关联规则、相关性、因果关系的区别

3.1 基本概念

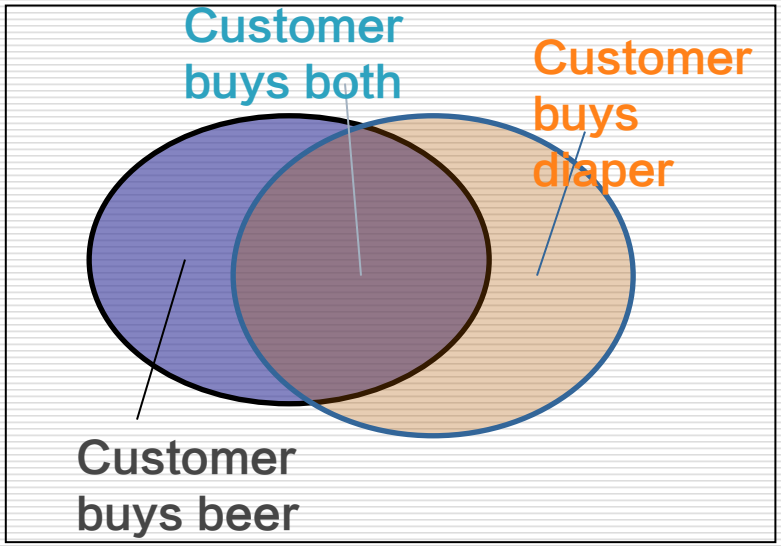
- **关联规则挖掘：** 从事务数据库中发现项集之间有趣的关联.
- **应用：** 购物篮分析（Basket data analysis）、交叉营销（cross-marketing）、产品目录设计（catalog design）等
- **例子：**
  - 规则形式: “Body → Head [support, confidence]”.
  - buys(x, “diapers”) → buys(x, “beers”) [0.5%, 60%]
  - major(x, “CS”) ^ takes(x, “DB”) → grade(x, “A”) [1%, 75%]

3.1 基本概念

- 布尔型与数值型关联规则 (基于要处理的数据类型)
  - buys(x, “SQLServer”) ^ buys(x, “DMBook”) → buys(x, “DBMiner”) [0.2%, 60%]
  - age(x, “30..39”) ^ income(x, “42..48K”) → buys(x, “PC”) [1%, 75%]
- 单维与多维关联规则
- 单层与多层关联规则
  - What brands of beers are associated with what brands of diapers?

3.1 基本概念

■ 规则度量：支持度和置信度



TID	Item
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

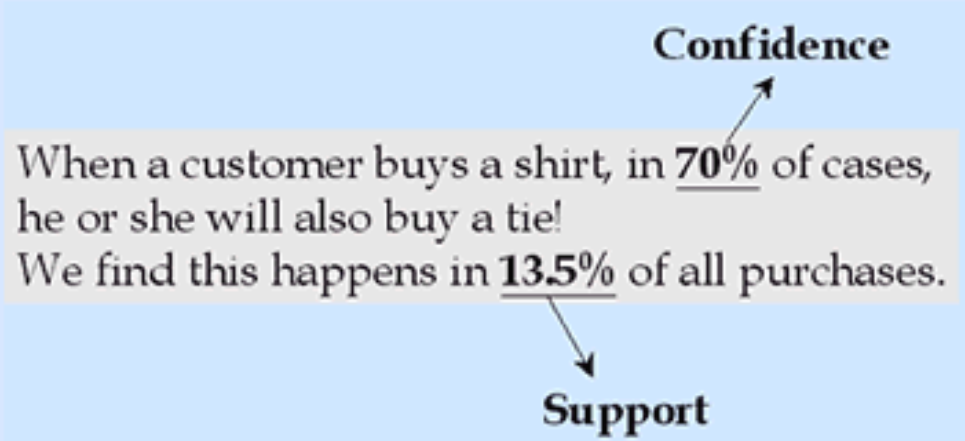
- 支持度s 是指事务集 D 中包含 $A \cup B$ 的百分比
  - $Support(A \rightarrow B) = P(A \cup B)$
- 置信度 c 是指D 中包含A 的事务同时也包含B的百分比
  - $Confidence(A \rightarrow B) = P(B \mid A) = P(A \cup B) / P(A)$

$A \rightarrow C$ ？ $C \rightarrow A$ ？

3.1 基本概念

■ 规则度量：支持度和置信度

Shirt → Tie (support = 13.5% and confidence = 70%).



- 设最小支持度阈值为50%，最小置信度阈值为50%，则有如下规则
  - A → C（50%, 66.6%）
  - C → A（50%, 100%）
- 同时满足最小支持度阈值和最小置信度阈值的规则称作**强规则**

3.1 基本概念

■ 关联规则的形式化表示

- Let $I = \{i_1, i_2, \cdots, i_m\}$ be a set of items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$.
- A transaction T contains X, a set of some items in I, if $X \subseteq T$.
- An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$.
- $X \Rightarrow Y$ holds in the transaction set D with *confidence* c if c% of transactions in D that contain X also contain Y.
- $X \Rightarrow Y$ has *support* s% in the transaction set D if s% of transactions in D contain $X \cup Y$.

3.1 基本概念

- 事务数据：商场购物数据
  - Market basket transactions:

    t1: {bread, cheese, milk}

    t2: {apple, eggs, salt, yogurt}

    ... ...

    tn: {biscuit, eggs, milk}

  - Concepts:

    An *item*: an item/article in a basket

    I: the set of all items sold in the store

    A *transaction*: items purchased in a basket; it may have TID (transaction ID)

    A *transactional dataset*: A set of transactions

3.1 基本概念

- 事务数据：文档数据集
  - ✓ 每个文档都可以看作一个词袋（ bag of words ）

    doc1: Student, Teach, School

    doc2: Student, School

    doc3: Teach, School, City, Game

    doc4: Baseball, Basketball

    doc5: Basketball, Player, Spectator

    doc6: Baseball, Coach, Game, Team

    doc7: Basketball, Team, City, Game

本章内容

3.1 关联规则挖掘的基本概念

3.2 由事务数据库挖掘单维布尔关联规则

3.3 多层关联规则挖掘

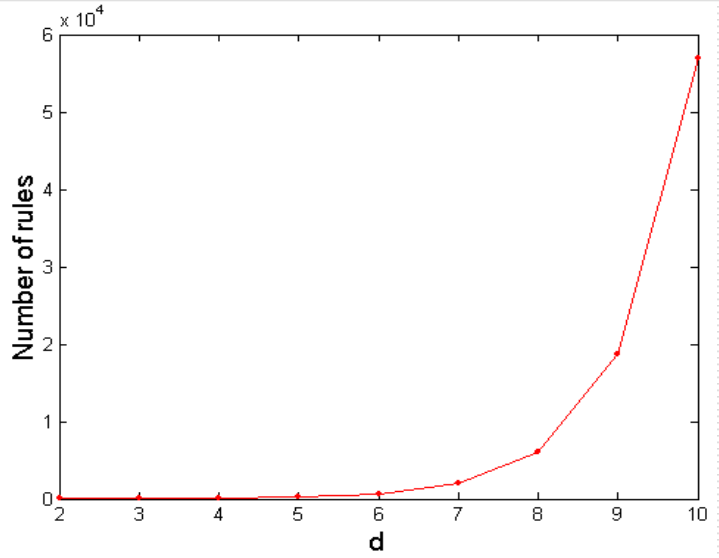
3.4 多维关联规则挖掘

3.5 关联规则、相关性、因果关系的区别

3.2 由事务数据库挖掘布尔关联规则

- **关联规则挖掘的原始算法：枚举每个可能的规则，计算其支持度与置信度。**
  - 包含d个项目的数据集中可能的规则数目为

$$R = 3^d - 2^{d+1} + 1$$



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right]$$

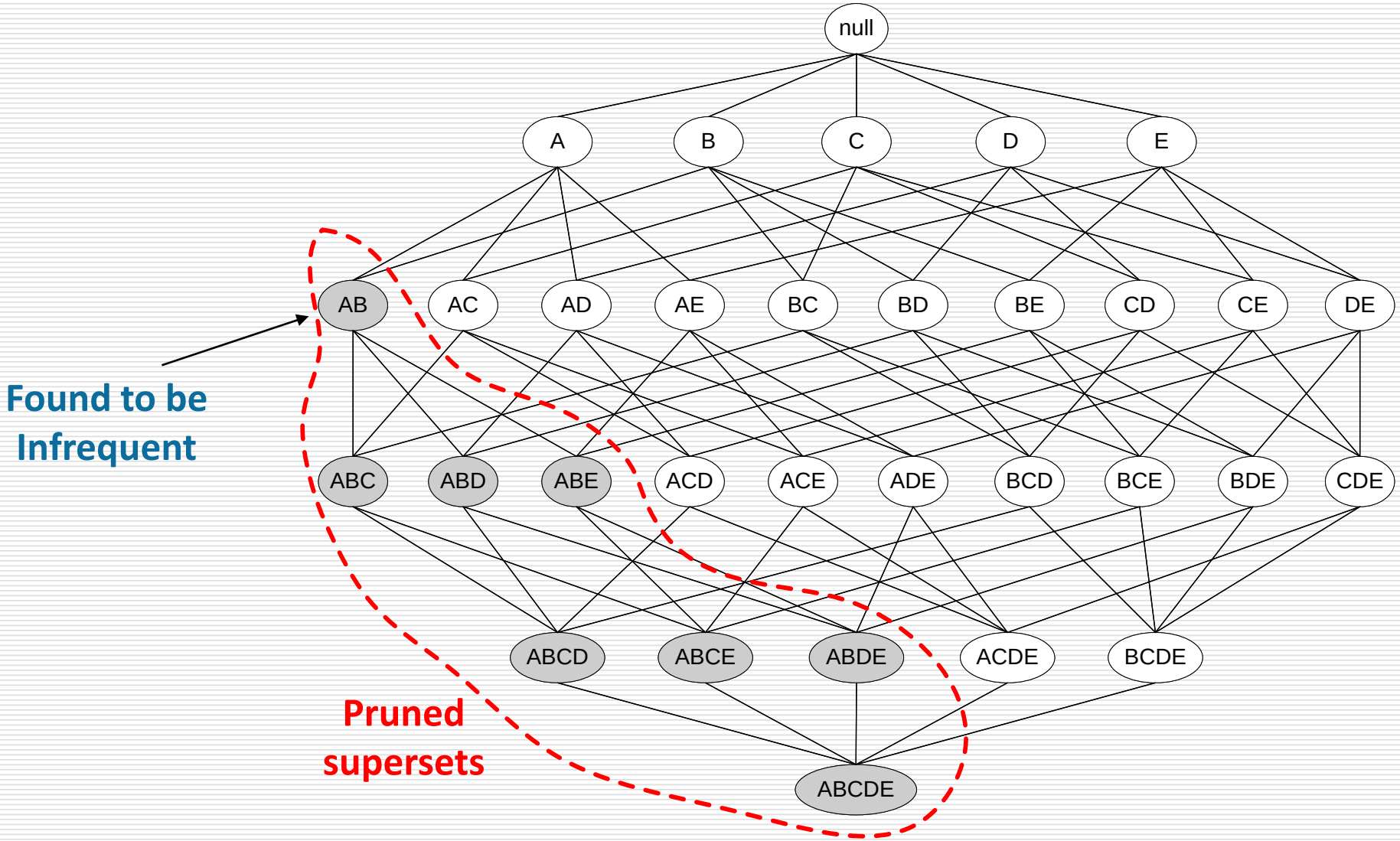
$$= 3^d - 2^{d+1} + 1$$

If d=6, R = 602 rules

3.2 由事务数据库挖掘布尔关联规则

- 频繁项集性质
  - 向下封闭性质（Downward closure）：频繁项集的任意子集都是频繁的
    - 如果 {beer, diaper, nuts} 是频繁的, 则 {beer, diaper}也是频繁的
    - 任何含有 {beer, diaper, nuts} 的事务数据也包含 {beer, diaper}
  - Apriori 修剪原理：非频繁项集的任意超集都是非频繁的，无需生成与测试

3.2 由事务数据库挖掘布尔关联规则

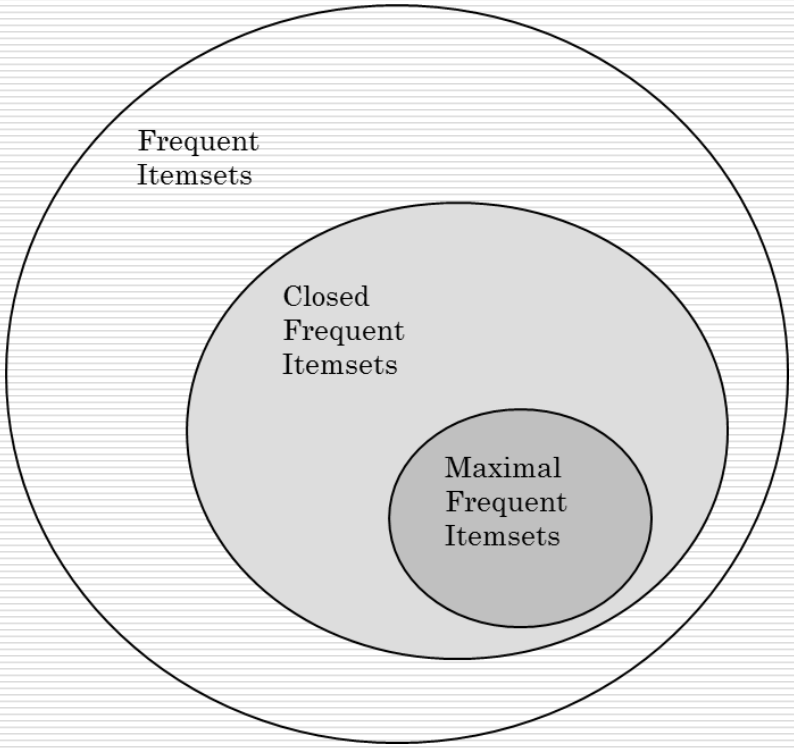


3.2 由事务数据库挖掘布尔关联规则

- A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, \cdots, a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + \cdots + \binom{1}{1}\binom{0}{0}\binom{0}{0} = 2^{100} - 1 = 1.27*10^{30}$ sub-patterns!
- Solution: *Mine closed patterns and max-patterns instead*
- An itemset X is **closed** if X is *frequent* and there exists *no super-pattern* $Y \supset X$, *with the same support* as X
- An itemset X is a **max-pattern** if X is frequent and there exists no frequent super-pattern $Y \supset X$
- Closed pattern is a lossless compression of freq. patterns

3.2 由事务数据库挖掘布尔关联规则

- **Closed Patterns and Max-Patterns**
  - a maximal itemset is a closed itemset but a closed itemset is not necessarily a maximal itemset.



3.2 由事务数据库挖掘布尔关联规则

- Closed Patterns and Max-Patterns
  - Exercise. DB = {<a_1, …, a_{100}>, < a_1, …, a_{50}>}
    - Min_sup = 1.
  - What is the set of closed itemset?
    - <a_1, …, a_{100}>: ?
    - < a_1, …, a_{50}>: ?
    - < a_1, …, a_{49}>:
  - What is the set of max-pattern?
    - <a_1, …, a_{100}>: ?
  - What is the set of all patterns?
    - !!

3.2 由事务数据库挖掘布尔关联规则

- Apriori算法
  - C_k: 候选 k−项集

    L_k: 频繁k−项集
  - Join 阶段: C_k 由L_{k-1}链接而成
  - Prune阶段: 任何包含非频繁的(k−1)−项集都不可能是频繁的k−项集

3.2 由事务数据库挖掘布尔关联规则

■ Apriori算法

1) L_1 = {large 1-itemsets};
2) for ($k = 2$; $L_{k-1} \neq \emptyset$; k++) do begin
3) C_k = apriori-gen(L_{k-1}); // New candidates
4) forall transactions $t \in D$ do begin
5) C_t = subset(C_k, t); //Candidates contained in t
6) forall candidates $c \in C_t$ do
7) c.count++;
8) end
9) $L_k = \{c \in C_k \mid c.\text{count} \geq \text{minsup}\}$
10) end
11) Answer = $\cup_k L_k$;

3.2 由事务数据库挖掘布尔关联规则

Apriori候选集生成

- apriori–gen 函数以所有的 $(k-1)$ 项集的集合 L_{k-1} 为输入，返回所有 k–项集的集合 C_k.
- 首先, 在 join 阶段:

  insert into C_k
  select $p.\text{item}_1, p.\text{item}_2, \cdots, p.\text{item}_{k-1}, q.\text{item}_{k-1}$
  from $L_{k-1}\ p, L_{k-1}\ q$
  where $p.\text{item}_1 = q.\text{item}_1, \cdots, p.\text{item}_{k-2} = q.\text{item}_{k-2}, p.\text{item}_{k-1} < q.\text{item}_{k-1}$
- 其次, 在 prune 阶段, 删除所有的满足 $(k-1)$ 子集不在 L_{k-1} 的项集 $c \in C_k$:

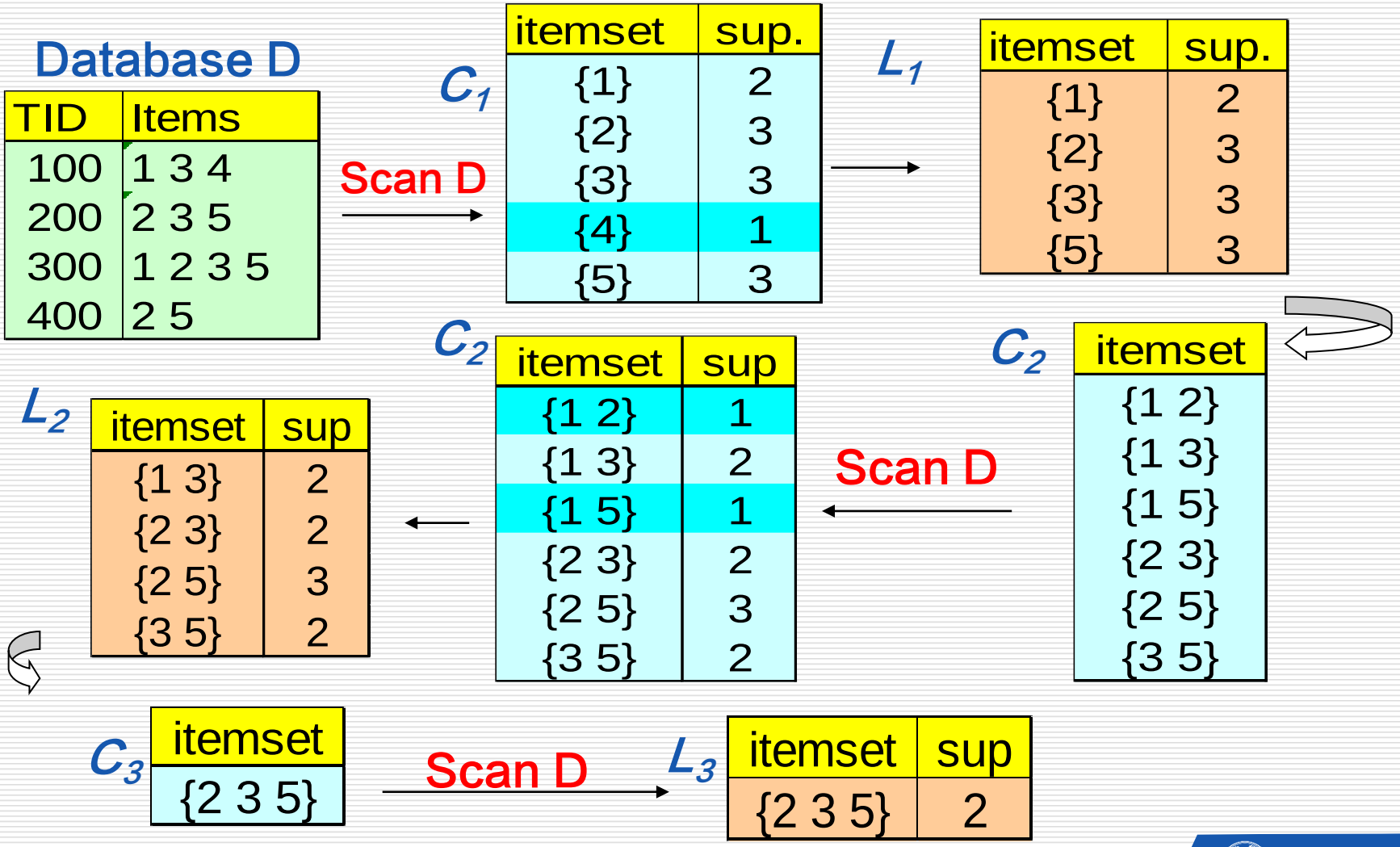
  forall itemsets $c \in C_k$ do
  forall $(k-1)$–subsets s of c do
  if $(s \notin L_{k-1})$ then
  delete c from C_k;

3.2 由事务数据库挖掘布尔关联规则

- L_3={*abc, abd, acd, ace, bcd*}
- **Self-joining:** L_3*L_3
  - *abcd* from *abc* and *abd*
  - *acde* from *acd* and *ace*
- **Pruning:**
  - *acde* is removed because *ade* is not in L_3
- C_4={*abcd*}

3.2 由事务数据库挖掘布尔关联规则

- Apriori算法举例



3.2 由事务数据库挖掘布尔关联规则

■ 由频繁项集产生关联规则

✓ 同时满足最小支持度和最小置信度的才是强关联规则，从频繁项集产生的规则都满足支持度要求，而其置信度则可由以下公式计算：

$$confidence(A \Rightarrow B) = P(A \mid B) = \frac{\sup port\_count(A \cup B)}{\sup port\_count(A)}$$

✓ 每个关联规则可由如下过程产生：

- 对于每个频繁项集l，产生l的所有非空子集；
- 对于每个非空子集s，如果 $\frac{\sup port\_count(l)}{\sup port\_count(s)} \geq \min\_conf$

  则输出规则“ $s \Rightarrow (l - s)$ ”

有多少个候选规则?

3.2 由事务数据库挖掘布尔关联规则

- 如果{A,B,C,D} 频繁项集，候选规则为:

ABC →D,	ABD →C,	ACD →B,	BCD →A,
A →BCD,	B →ACD,	C →ABD,	D →ABC
AB →CD,	AC → BD,	AD → BC,	BC →AD,
BD →AC,	CD →AB		

- 如果|L| = k, 那么有2^k – 2候选规则 (忽略L → ∅ and ∅ → L)
- 候选规则天生满足最小支持度，关键是需要判断是否满足最小置信度

3.2 由事务数据库挖掘布尔关联规则

■ 如何有效地从频繁项集生成关联规则

- 一般来说，置信度不具有反单调性

  c(ABC →D) can be larger or smaller than c(AB →D)

- 有相同项集生成的规则的置信度具有反单调性

如果规则 $X \rightarrow Y-X$ 不满足置信度阈值，则形如 $X' \rightarrow Y-X'$ 的规则一定也不满足置信度阈值，其中 X' 是 X 的子集。

e.g., L = {A,B,C,D}:

为什么?

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

3.2 由事务数据库挖掘布尔关联规则

- **对于频繁k-项集 (k > 2)**
  - 计算规则头中只有一个项的规则的置信度
  - 利用这些规则，迭代方法增加规则头中项集大小，计算置信度
  - 如果置信度小于阈值，则裁剪

{Bread, Milk, Diaper}

1-item rules

{Bread, Milk} → {Diaper}

{Milk, Diaper} → {Bread}

{Diaper, Bread} → {Milk}

2-item rules

{Bread} → {Milk, Diaper}

{Diaper} → {Milk, Bread}

{Milk} → {Diaper, Bread}

3.2 由事务数据库挖掘布尔关联规则

例子(Association Rule.ipynb)

3.2 由事务数据库挖掘布尔关联规则

Min_sup 40% (2/5)

TID	List of Items
1	Beer,Diaper,Baby Powder,Bread,Umbrella
2	Diaper,Baby Powder
3	Beer,Diaper,Milk
4	Beer,Diaper,Detergent
5	Beer,Milk,Coca-Cola

3.2 由事务数据库挖掘布尔关联规则

- 例子

C1 → L1

Item	Support
Beer	"4/5"
Diaper	"4/5"
Baby Powder	"2/5"
Bread	"1/5"
Umbrella	"1/5"
Milk	"2/5"
Detergent	"1/5"
Coca-Cola	"1/5"

Item	Support
Beer	"4/5"
Diaper	"4/5"
Baby Powder	"2/5"
Milk	"2/5"

3.2 由事务数据库挖掘布尔关联规则

- 例子

C2 → L2

Item	Support
Beer, Diaper	
Beer, Baby Powder	
Beer, Milk	
Diaper,Baby Powder	
Diaper,Milk	
Baby Powder,Milk	

Item	Support
Beer, Diaper	"3/5"
Beer, Milk	"2/5"
Diaper,Baby Powder	"2/5"

3.2 由事务数据库挖掘布尔关联规则

- 例子

C3 → empty

Item	Support
Beer, Diaper,Baby Powder	
Beer, Diaper,Milk	

3.2 由事务数据库挖掘布尔关联规则

- 例子

min_sup=40%　min_conf=70%

Item	Support(A,B)	Suport A	Confidence
Beer, Diaper			
Beer, Milk			
Diaper,Baby Powder			
Diaper,Beer			
Milk,Beer			
Baby Powder, Diaper			

3.2 由事务数据库挖掘布尔关联规则

- 例子

$$Beer \Rightarrow Diaper$$

support 60%, confidence 75%

$$Diaper \Rightarrow Beer$$

support 60%, confidence 75%

$$Milk \Rightarrow Beer$$

support 40%, confidence 100%

$$Baby\_Powder \Rightarrow Diaper$$

support 40%, confidence 100%

3.2 由事务数据库挖掘布尔关联规则

- Apriori算法通常是可行的（三个因素：单调性、稀疏性、支持度）
- 提高Apriori算法的有效性
- Apriori算法主要的挑战
  - 要对数据进行多次扫描；
  - 会产生大量的候选项集；
  - 对候选项集的支持度计算非常繁琐；

Assume the total number of items be 100.

Cardinality of Itemsets	Number of Itemsets
1	100
2	4,950
3	161,700
4	3,921,225
5	75,287,529
6	1,192,052,400
7	16,007,560,800
8	186,087,894,300

If a typical supermarket has at least 1,0000 different items, there are almost 50,000,000 possible 2-itemsets (i.e., itemset with the cardinality of 2) and over 1,000,000,000 possible 3-itemsets.

3.2 由事务数据库挖掘布尔关联规则

■ 提高Apriori算法的有效性

➢ 解决思路

- 减少对数据的扫描次数；
- 缩小产生的候选项集；
- 改进对候选项集的支持度计算方法

3.2 由事务数据库挖掘布尔关联规则

■ AprioriTid算法

1. 同Apriori 算法一样利用Apriori-gen 过程生成候选项集；
2. 与Apriori算法的重要区别是支持度的确定：仅用扫描一次数据库
3. 用集合C’k 代替数据库
4. 每个C’k的元素形如<TID, $\{X_k\}$ >， 每个X_k 是事务 TID包含的k项集
5. C’1 对应于数据库 D.

Database

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

$C^\wedge_1$

TID	Set-of-itemsets
100	{ {1},{3},{4} }
200	{ {2},{3},{5} }
300	{ {1},{2},{3},{5} }
400	{ {2},{5} }

L_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

$C^\wedge_2$

TID	Set-of-itemsets
100	{ {1 3} }
200	{ {2 3},{2 5} {3 5} }
300	{ {1 2},{1 3},{1 5}, {2 3}, {2 5}, {3 5} }
400	{ {2 5} }

L_2

Itemset	Support
{1 3}	2
{2 3}	3
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

$C^\wedge_3$

TID	Set-of-itemsets
200	{ {2 3 5} }
300	{ {2 3 5} }

L_3

Itemset	Support
{2 3 5}	2

3.2 由事务数据库挖掘布尔关联规则

- **Apriori算法**
  - 计算项集的支持度需要花费大量时间，需要检查数据库中的每个事务，涉及大量I/O资源
- **AprioriTid算法**
  - 初始阶段候选项集较大.时间开销和 Apriori 相当，并且要占用大量的内存开销

3.2 由事务数据库挖掘布尔关联规则

- Apriori Hybrid算法
  - ✓ 初始阶段: Apriori 性能较好
  - ✓ 后续阶段: AprioriTid性能较好
  - ✓ Apriori Hybrid:
    - 在初始阶段使用Apriori算法，后续使用AprioriTid算法
    - 缺点: Apriori 向AprioriTid过度会带来额外开销

3.2 由事务数据库挖掘布尔关联规则

■ 关联规则的兴趣度度量

- **客观度量：** 支持度、置信度
- **主观度量：** 最终，只有用户才能确定一个规则是否有趣的，而且这种判断是主观的，因不同的用户而异；通常认为一个规则（模式）是有趣的，如果：
  - 它是出人意料的
  - 可行动的（用户可以使用该规则做某些事情）

3.2 由事务数据库挖掘布尔关联规则

■ 关联规则可视化

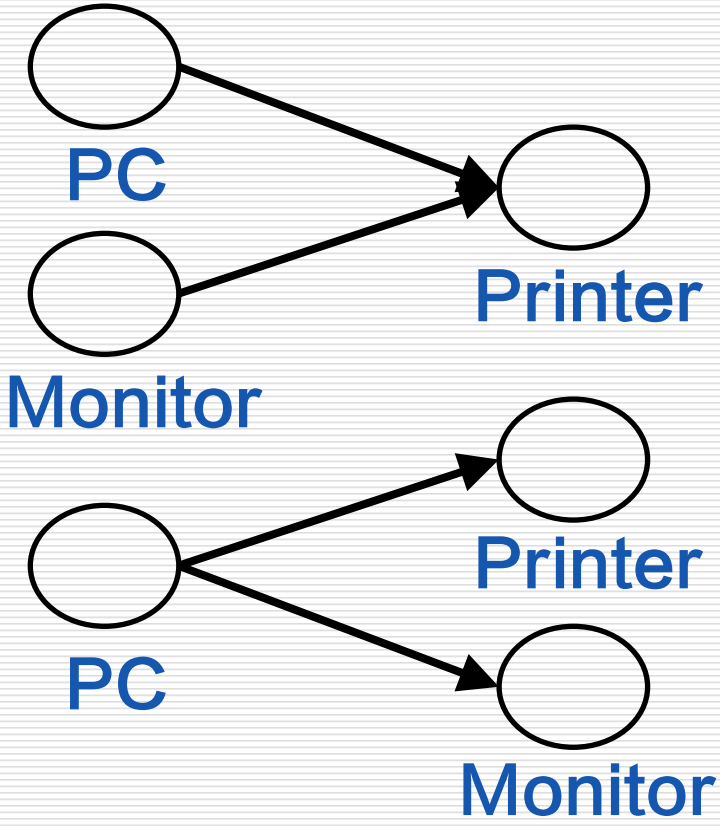
➢ 表格

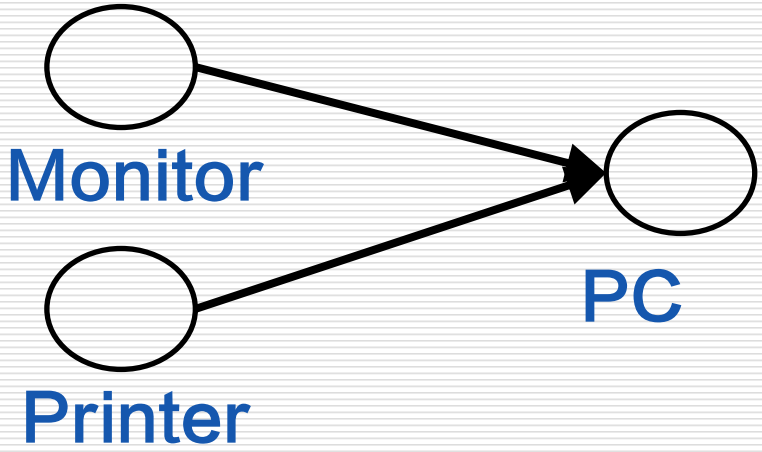
Antecedent		Consequent	Support	Confidence
PC, Monitor	⇨	Printer	90%	85%
PC	⇨	Printer, Monitor	90%	75%
Printer, Monitor	⇨	PC	80%	70%

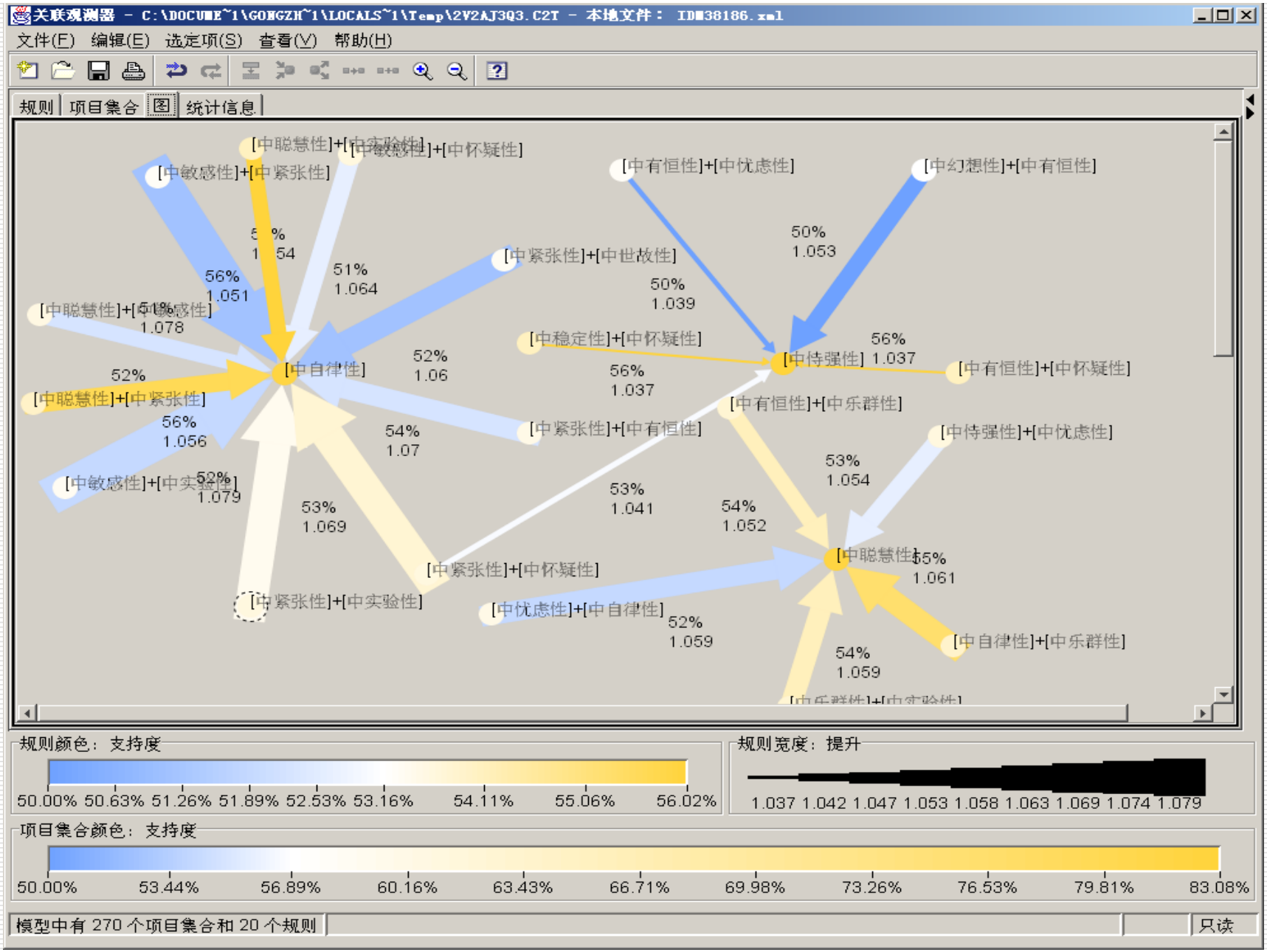
3.2 由事务数据库挖掘布尔关联规则

■ 关联规则可视化

➢ 有向图

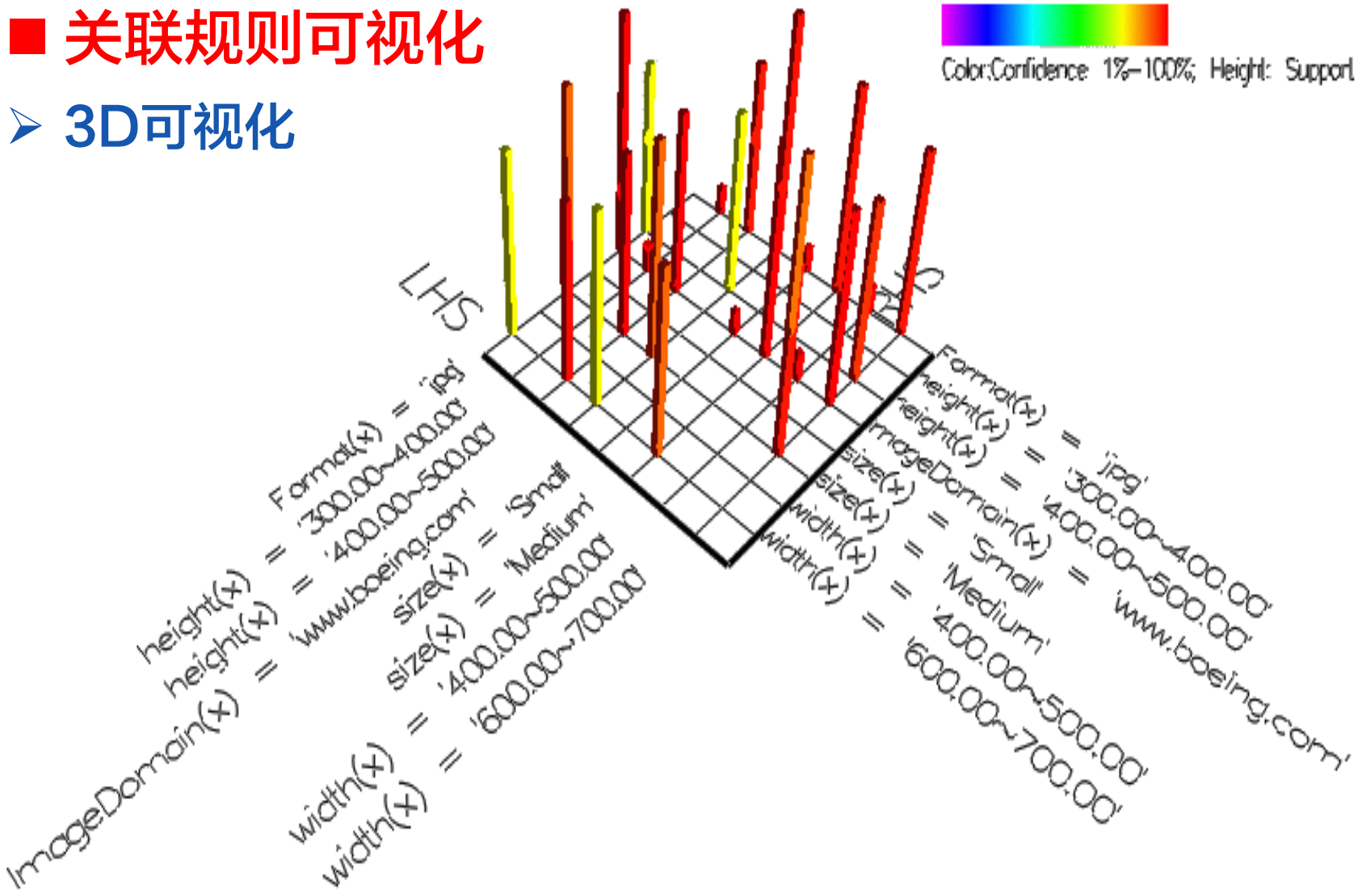





关联观测器 - C:\DOCUME~1\GONGZH~1\LOCALS~1\Temp\2V2AJ3Q3.C2T - 本地文件： ID■38186.xml
文件(F) 编辑(E) 选定项(S) 查看(V) 帮助(H)
规则 项目集合 图 统计信息
[中聪慧性]+[中实验性]
[中敏感性]+[中紧张性]
[中敏感性]+[中怀疑性]
[中有恒性]+[中忧虑性]
[中幻想性]+[中有恒性]
[中紧张性]+[中世故性]
[中聪慧性]+[中敏感性]
[中稳定性]+[中怀疑性]
[中自律性]
[中侍强性]
[中有恒性]+[中怀疑性]
[中聪慧性]+[中紧张性]
[中有恒性]+[中乐群性]
[中紧张性]+[中有恒性]
[中侍强性]+[中忧虑性]
[中敏感性]+[中实验性]
[中聪慧性]
[中紧张性]+[中怀疑性]
[中紧张性]+[中实验性]
[中忧虑性]+[中自律性]
[中自律性]+[中乐群性]
[中乐群性]+[中实验性]
56% 1.051
51% 1.064
50% 1.053
50% 1.039
56% 1.078
56% 1.037
52% 1.06
52%
56% 1.056
54% 1.07
52% 1.079
53% 1.069
53% 1.041
54% 1.052
53% 1.054
55% 1.061
52% 1.059
54% 1.059
规则颜色： 支持度
50.00% 50.63% 51.26% 51.89% 52.53% 53.16% 54.11% 55.06% 56.02%
规则宽度： 提升
1.037 1.042 1.047 1.053 1.058 1.063 1.069 1.074 1.079
项目集合颜色： 支持度
50.00% 53.44% 56.89% 60.16% 63.43% 66.71% 69.98% 73.26% 76.53% 79.81% 83.08%
模型中有 270 个项目集合和 20 个规则
只读

3.2 由事务数据库挖掘布尔关联规则

- **关联规则可视化**
  - 3D可视化



本章内容

3.1 关联规则挖掘的基本概念

3.2 由事务数据库挖掘单维布尔关联规则

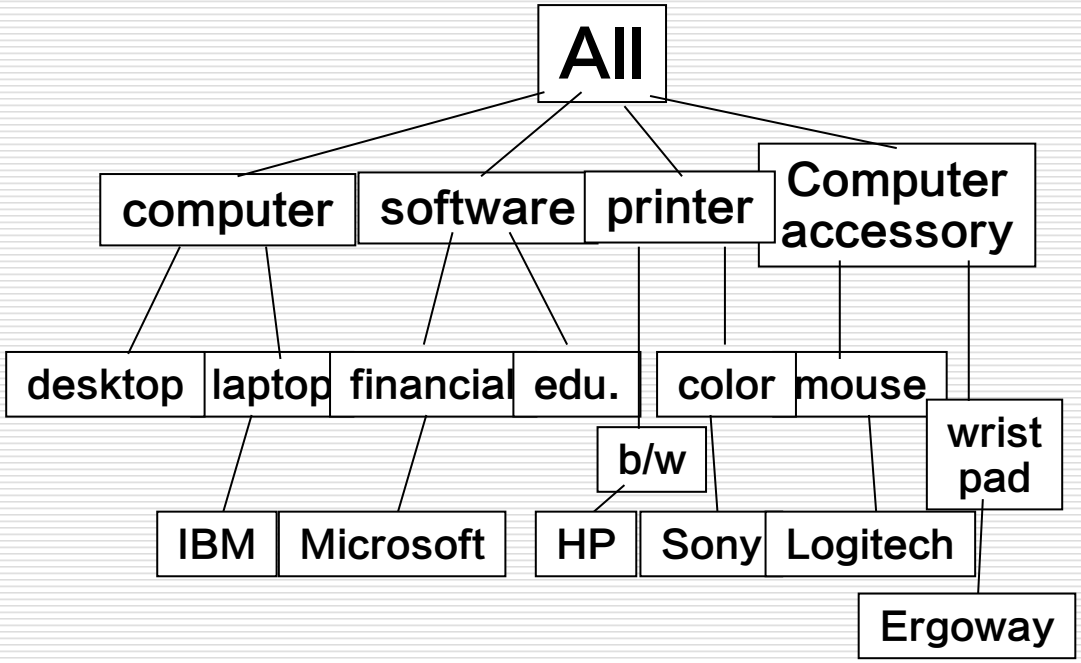
3.3 多层关联规则挖掘

3.4 多维关联规则挖掘

3.5 关联规则、相关性、因果关系的区别

3.3 多层关联规则挖掘

- 数据项中经常会形成概念分层
- 底层数据项，其支持度往往也较低
  - 挖掘底层数据项之间的关联规则必须定义不同的支持度



TID	Items
T1	{IBM D/C, Sony b/w}
T2	{Ms. edu. Sw., Ms. fin. Sw.}
T3	{Logi. mouse, Ergoway wrist pad}
T4	{IBM D/C, Ms. Fin. Sw.}
T5	{IBM D/C}

3.3 多层关联规则挖掘

- 在适当的等级挖掘出来的数据项间的关联规则可能是非常有用的
- 事务数据库中的数据也是根据维和概念分层来进行储存的
  - 这为从事务数据库中挖掘不同层次的关联规则提供了可能。
- 在多个抽象层挖掘关联规则，并在不同的抽象层进行转化，是数据挖掘系统应该提供的能力

3.3 多层关联规则挖掘

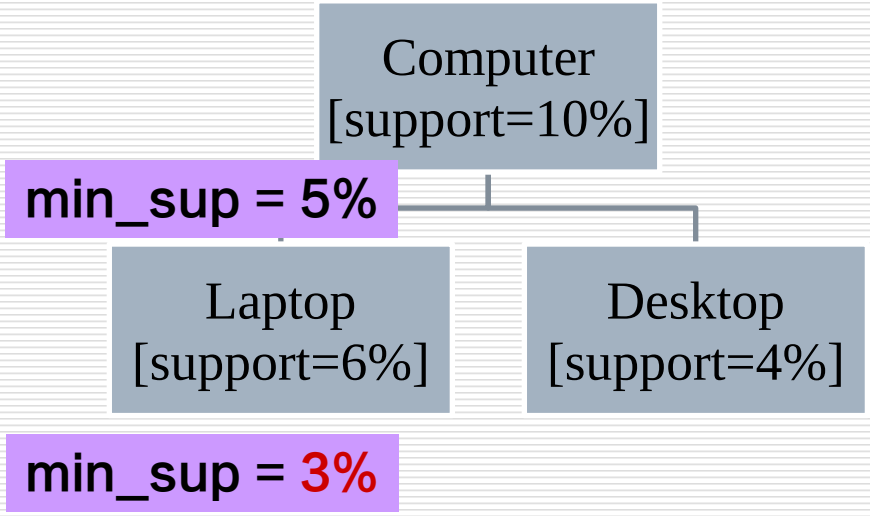
- 多层关联规则的挖掘还是使用置信度－支持度框架，可以采用自顶向下策略
  - 概念分层中，一个节点的支持度肯定不小于该节点的任何子节点的支持度（为什么？）
  - 由概念层1开始向下，到较低的更特定的概念层，对每个概念层的频繁项计算累加计数
  - 每一层的关联规则挖掘可以使用Apriori等多种方法
  - 例如：
    - 先找高层的关联规则：computer –> printer [20%, 60%]
    - 再找较低层的关联规则：laptop –> color printer [10%, 50%]

3.3 多层关联规则挖掘

- **一致支持度：对所有层都使用一致的最小支持度**
  - **优点：**搜索时容易采用优化策略，即一个项如果不满足最小支持度，它的所有子项都可以不用搜索
  - **缺点：**最小支持度值设置困难
    - 太高：将丢掉出现在较低抽象层中有意义的关联规则
    - 太低：会在较高层产生太多的无兴趣的规则

3.3 多层关联规则挖掘

- 使用递减支持度，可以解决使用一致支持度时在最小支持度值上设定的困难
- 递减支持度：在较低层使用递减的最小支持度
  - 每一层都有自己的一个独立的最小支持度
  - 抽象层越低，对应的最小支持度越小



3.3 多层关联规则挖掘

■ 多层关联——基于分组的支持度

- 用户和专家清楚哪些分组比其他分组更加重要，在挖掘多层关联规则时，使用用户指定的基于项或者基于分组的最小支持度阈值
  - E.g. 对laptop_computer或者flash_drives设置特别低的支持度阈值，以便特别关注这类商品的管理模式

3.3 多层关联规则挖掘

Hierarchy-information encoded transaction table: T[1]

TID	Items
T_1	{111, 121, 211, 221}
T_2	{111, 211, 222, 323}
T_3	{112, 122, 221, 411}
T_4	{111, 121}
T_5	{111, 122, 211, 221, 413}
T_6	{211, 323, 524}
T_7	{323, 411, 524, 713}

3.3 多层关联规则挖掘

minsup[1] = 4

① **Level−1 large 1−itemsets: L[1, 1]**

Itemset	Support
{1**}	5
{2**}	5

② **Filtered transaction table: T[2]**

TID	Items
T_1	{111, 121, 211, 221}
T_2	{111, 211, 222}
T_3	{112, 122, 221}
T_4	{111, 121}
T_5	{111, 122, 211, 221}
T_6	{211}

③ **Level−1 large 2−itemsets: L[1, 2]**

– Use L[1,1] and T[2]

L[1,1] is used to filter:
(1) any item which is not large in a transaction
(2) the transactions in T[1] which contain only small items

3.3 多层关联规则挖掘

minsup[2] = 3

Level−2 large 1−itemsets: L[2, 1] (Note 1)

Itemset	Support
{11*}	5
{12*}	4
{21*}	4
{22*}	4

Level−2 large 2−itemsets: L[2, 2] (Note 2)

Itemset	Support
{11*, 12*}	4
{11*, 21*}	3
{11*, 22*}	4
{12*, 22*}	3
{21*, 22*}	3

Level−2 large 3−itemsets: L[2, 3] (Note 3)

Itemset	Support
{11*, 12*, 22*}	3
{11*, 21*, 22*}	3

TID	Items
T_1	{111, 121, 211, 221}
T_2	{111, 211, 222}
T_3	{112, 122, 221}
T_4	{111, 121}
T_5	{111, 122, 211, 221}
T_6	{211}

3.3 多层关联规则挖掘

minsup[3] = 3

Level−3 large 1−itemsets: L[3, 1] (Note 1)

Itemset	Support
{111}	4
{211}	4
{221}	3

Level−3 large 2−itemsets: L[3, 2]

Itemset	Support
{111, 211}	3

TID	Items
T_1	{111, 121, 211, 221}
T_2	{111, 211, 222}
T_3	{112, 122, 221}
T_4	{111, 121}
T_5	{111, 122, 211, 221}
T_6	{211}

本章内容

3.1 关联规则挖掘的基本概念

3.2 由事务数据库挖掘单维布尔关联规则

3.3 多层关联规则挖掘

3.4 多维关联规则挖掘

3.5 关联规则、相关性、因果关系的区别

3.4 多维关联规则挖掘

- 单维关联规则：
  - buys(X, “milk”) = buys(X, “bread”)
- 多维关联规则：涉及两个或多个维或谓词的关联规则
  - 维间关联规则：不包含重复的谓词
    - age(X,” 19-25”) ∧ occupation(X, “student”) => buys(X, “coke”)
  - 混合维关联规则：包含某些谓词的多次出现
    - age(X,” 19-25”) ∧ buys(X, “popcorn”) => buys(X, “coke”)
- 在多维关联规则挖掘中，搜索的不是频繁项集，而是频繁谓词集。k-谓词集是包含k个合取谓词的集合。
  - 例如：{age, occupation, buys}是一个3-谓词集

3.4 多维关联规则挖掘

- 数据属性可以分为类别属性和数值属性
  - 类别属性：具有有限个不同值，值之间无序
  - 数值属性：数值类型，并且值之间有一个隐含的序
- 对量化属性的处理：
  - 1. 静态离散化：使用预定义的概念分层对数值属性进行静态地离散化
  - 2. 量化关联规则：根据数据的分布，将数值属性离散化到“箱”
  - 3. 基于距离的关联规则：考虑数据点之间的距离，动态地离散化量化

3.4 多维关联规则挖掘

RecordID	Age	Married	NumCars
100	23	No	1
200	25	Yes	1
300	29	No	0
400	34	Yes	2
500	38	Yes	2

- <Age: 30..39> and <Married: Yes> => <NumCars: 2>
- Support = 40%, Conf = 100%

3.4 多维关联规则挖掘

- **基本思路：** 将数值型关联规则挖掘转化为布尔型关联规则挖掘
  - 引入<attribute: value> 作为新属性，这个属性是布尔型

ID	Age: 20..29	Age: 30..39	Married: Yes	Married: No	NumCars: 0	NumCars: 1
100	1	0	0	1	0	1
200	1	0	1	0	0	1
300	1	0	0	1	1	0
400	0	1	1	0	0	0
500	0	1	1	0	0	0

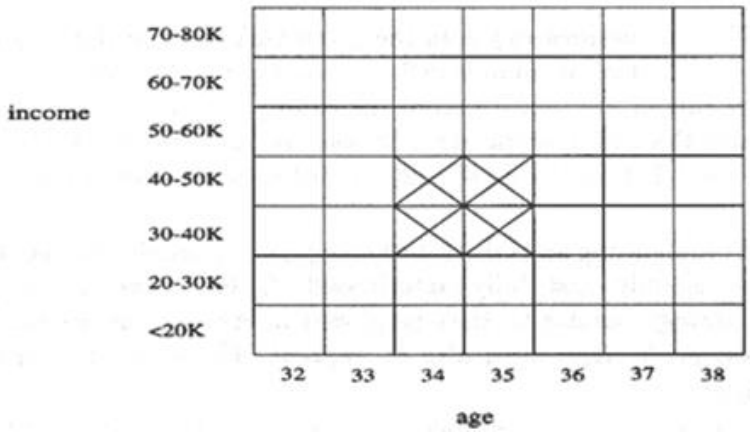
3.4 多维关联规则挖掘

- 聚焦于以下形式的2-维量化关联规则：
  - 两个量化属性和一个分类属性间的关联：

$$A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$$

  - 如： $age(X,"30-39") \wedge income(X,"42K-48K") \Rightarrow buys(X,"high\ resolution\ TV")$

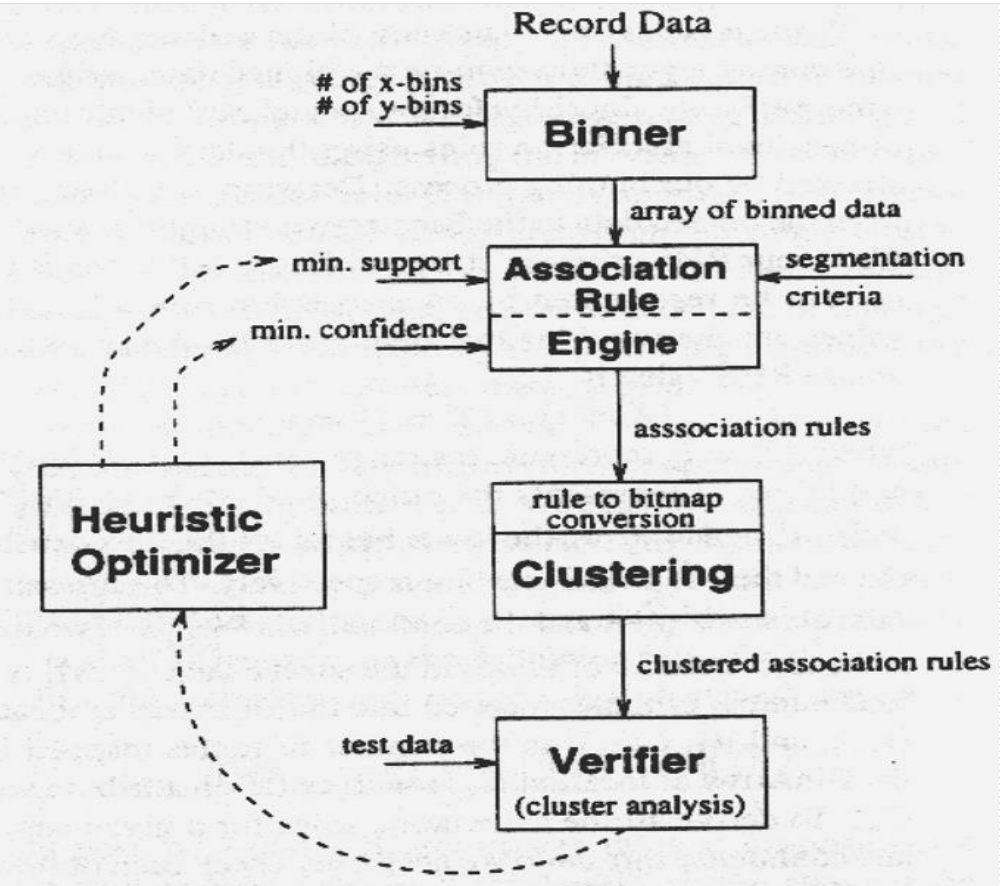
- 关联规则聚类系统 (Association Rule Clustering System, ARCS)
  - ✓ 源于图像处理技术，该技术将量化属性对映射到满足给定分类属性条件的2-D栅格上，然后通过搜索栅格点的聚类而产生关联规则



3.4 多维关联规则挖掘

- ARCS过程:

1. Binning
2. Find frequent predicateset
3. Clustering
4. Optimize



3.4 多维关联规则挖掘

■ ARCS过程:

1. 分箱（根据不同分箱方法创建一个2-D数组），目的在于减少量化属性相对应的巨大的值个数，使得2-D栅格的大小可控
   - 等宽分箱
   - 等深分箱
   - 基于同质的分箱（每个箱中元组一致分布）
2. 找出频繁谓词集
   - 扫描分箱后形成的2-D数组，找出满足最小支持度和置信度的频繁谓词集

3.4 多维关联规则挖掘

■ ARCS过程:

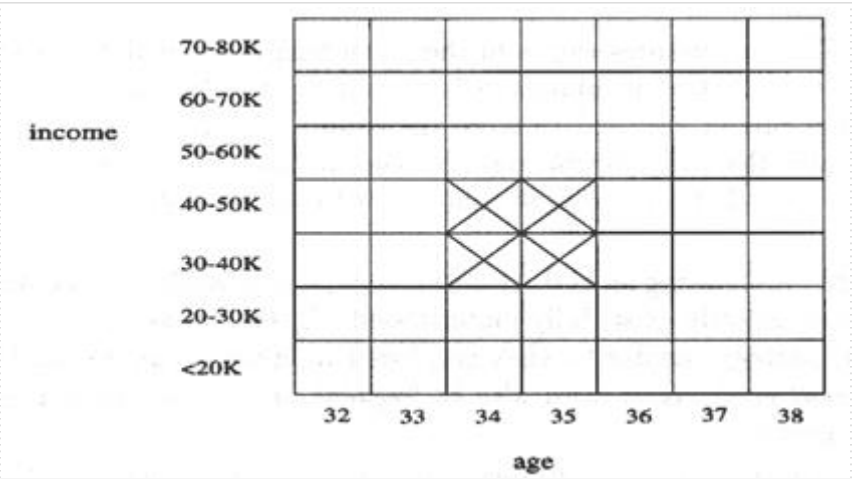
3. 关联规则聚类

- 将上一步得到的强关联规则映射到2-D栅格上，使用聚类算法，扫描栅格，搜索规则的矩形聚类

$$age(X,35) \wedge income(X,"31K...40K") \Rightarrow buys(X,"high\_resolution\_TV")$$

$$age(X,34) \wedge income(X,"41K...50K") \Rightarrow buys(X,"high\_resolution\_TV")$$

$$age(X,35) \wedge income(X,"41K...50K") \Rightarrow buys(X,"high\_resolution\_TV")$$



$$age(X,34...35) \wedge income(X,"31K...50K") \Rightarrow buys(X,"high\_resolution\_TV")$$

3.4 多维关联规则挖掘

- ARCS的局限性:
  - ✓ 规则的左手边只能有两个量化属性（2-D栅格的限制）

3.4 数值型关联规则挖掘

- *MinSup* 问题：如果数值型属性离散化后产生的区间的数量过多，单个区间的支持度就会较小，那么由于最小支持度限制，涉及这个属性的规则就可能不存在。区间数量增多也是布尔型属性增多，潜在规则会爆炸性增长

- *MinConf* 问题：当把数值型属性离散化为区间时，会产生信息丢失，当区间越大时，信息丢失的也就越多。

- *Catch-22* 两难境地：
  - 如果区间过大，一些规则由于最小置信度问题可能丢失
  - 如果区间过小，一些规则由于最小支持度问题可能丢失

本章内容

3.1 关联规则挖掘的基本概念

3.2 由事务数据库挖掘单维布尔关联规则

3.3 多层关联规则挖掘

3.4 多维关联规则挖掘

3.5 关联规则、相关性、因果关系的区别

3.5 关联规则、相关性、因果关系的区别

- 问题1：强关联规则是否是正相关的?
- 问题2：强关联规则是否存在因果关系?
- 问题3：相关性是否意味着因果关系?
- 问题4：如何定义因果关系?

3.5 关联规则、相关性、因果关系的区别

- 问题1：强关联规则是否是正相关的?
- 相关性：两个变量存在联系，一个变量会随着另一个变量变化，分为正相关和负相关

3.5 关联规则、相关性、因果关系的区别

	打篮球	不打篮球	合计
喝麦片	2000	1750	3750
不喝麦片	1000	250	1250
合计	3000	2000	5000

■ 例：(Aggarwal & Yu, PODS98)

- 在5000个学生中
  - 3000个打篮球
  - 3750个喝麦片粥
  - 2000个学生既打篮球又喝麦片粥
- 然而，打篮球 => 喝麦片粥 [? %, ? %]是无趣的，因为全部学生中喝麦片粥的比率是75%，比打篮球学生的66.7%要高
- 打篮球 => 不喝麦片粥 [20%, 33.3%]这个规则远比上面那个要精确，尽管支持度和置信度都要低的多

3.5 关联规则、相关性、因果关系的区别

- X and Y：正相关
- X and Z：负相关

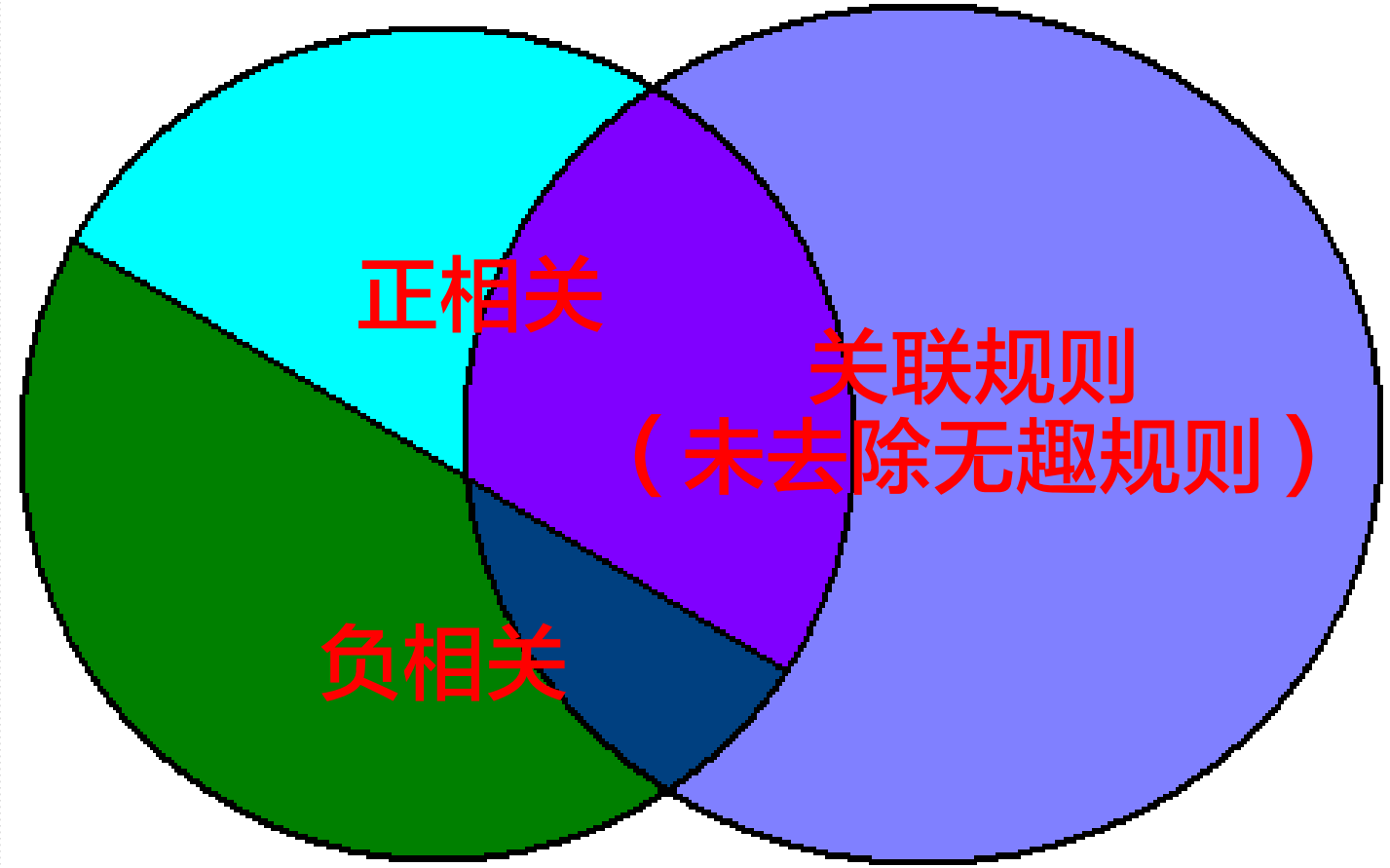
X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

Rule	Support	Confidence
X=>Y	25%	50%
X=>Z		

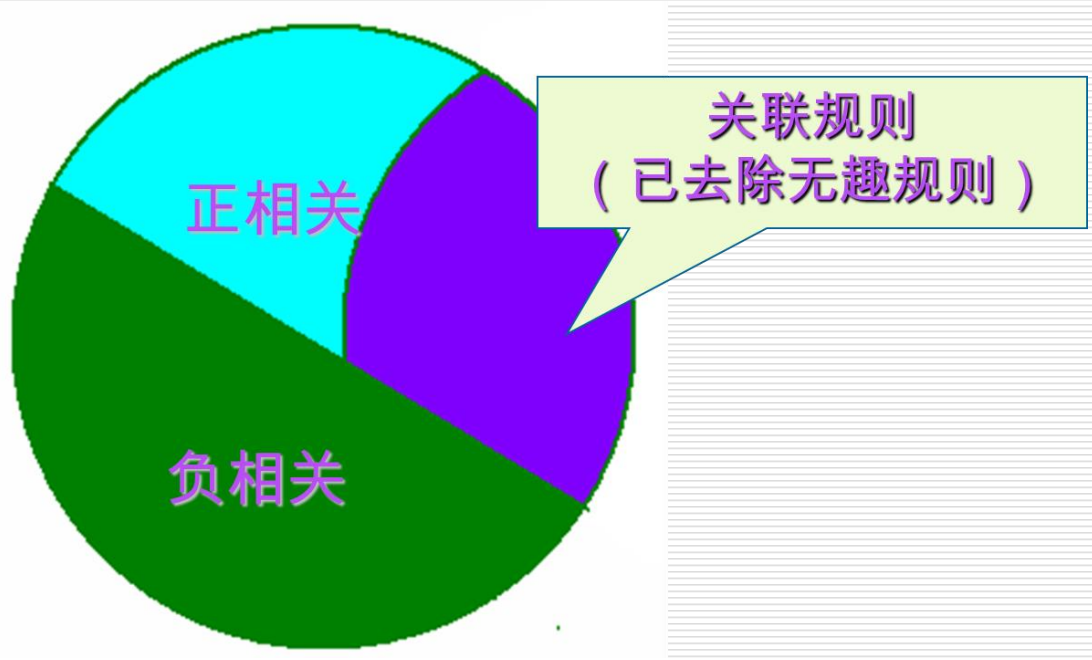
强关联规则 ≠ 正相关！

3.5 关联规则、相关性、因果关系的区别

关联规则与相关性



3.5 关联规则、相关性、因果关系的区别



- 可使用相关度来扩充关联规则的支持度---置信度框架

$$A \Rightarrow B[\text{sup } port, confidence, correlation]$$

- 需要一种度量事件间的相关性或者是依赖性的指标

3.5 关联规则、相关性、因果关系的区别

- **提升度指标（lift）**

$$A和B间的提升度：\ lift(A,B)=\frac{P(A\cup B)}{\mathrm{P(A)P(B)}}=P(B\mid A)/P(B)$$

- **当项集A的出现独立于项集B时，P(A∪B)=P(A)P(B)，即lift(A,B)=1，表明A与B无关，lift(A,B)>1表明A与B正相关，lift(A,B)<1表明A与B负相关**

3.5 关联规则、相关性、因果关系的区别

X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

Itemset	Support	left
X,Y	25%	2
X,Z	37.50%	0.9
Y,Z	12.50%	0.57

3.5 关联规则、相关性、因果关系的区别

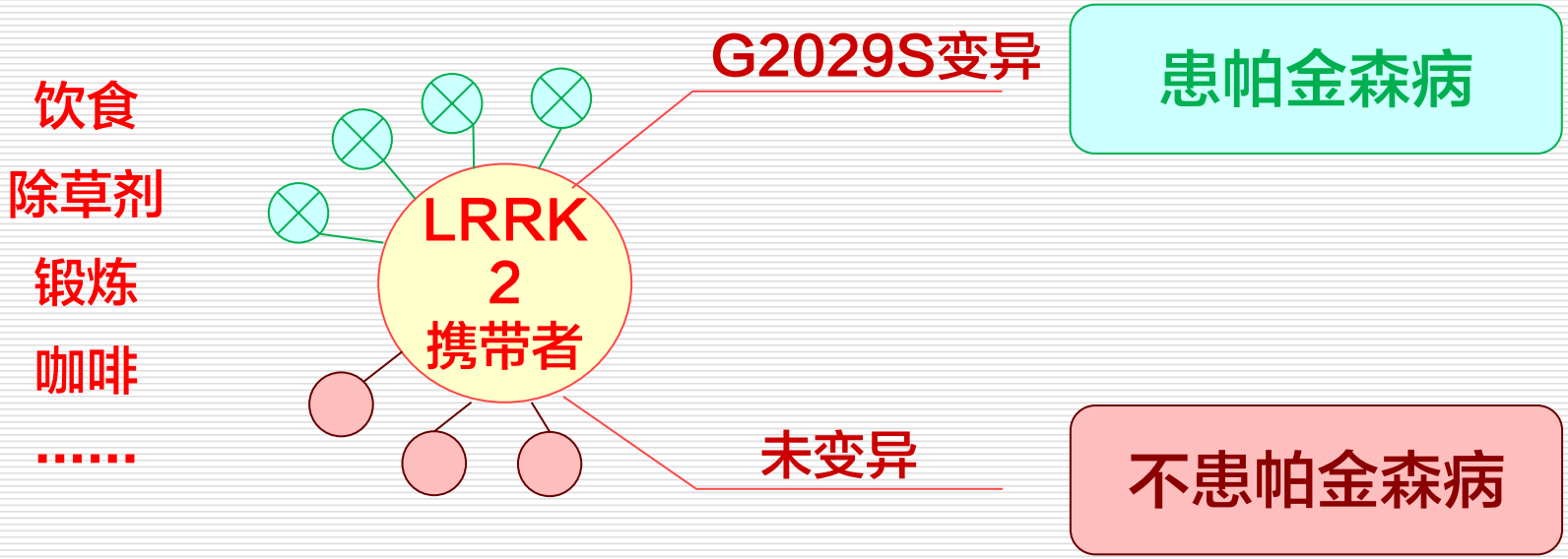
- 问题2：强关联规则是否存在因果关系?
- 因果关系：变量或事件之间的直接作用关系

太阳东方升起 => 周一上DM课 [1/7, 100%]

强关联规则 ≠ 因果关系!

3.5 关联规则、相关性、因果关系的区别

- 相关性有助于为发现因果关系提供线索



LRRK2携带者患帕金森的几率达到 30%-75%，普通患症的几率只有1%

3.5 关联规则、相关性、因果关系的区别

- 问题4：如何定义因果关系？
- Granger因果关系：若在包含了变量X、Y的过去信息的条件下，对变量Y的预测效果要优于只单独由Y的过去信息对Y的预测效果，即变量X 、Y有因果关系，变量X是变量Y的Granger原因

Clive W.J.Granger
诺贝尔经济学奖获奖者

总结

- Association rule mining consists of first finding frequent itemsets, from which strong association are generated. Associations can be further analyzed to uncover correlation rules, which convey statistical correlations between itemsets A and B.
- The Apriori algorithm is a seminal algorithm for mining frequent item sets for Boolean association rules.
- Mining frequent itemsets and associations has been extended in various ways to include mining multilevel association rules and multidimensional association rules.
- Not all strong association rules are interesting.