Artificial Intelligence & Machine Learning and Pattern Recognition ——Association Rule Mining

Yanghui Rao
Assistant Prof., Ph.D
School of Mobile Information Engineering,
Sun Yat-sen University
raoyangh@mail.sysu.edu.cn

Association Rule Mining

- Retailers (商家) are interested in the purchasing behavior of their customers.

Association Rule Mining

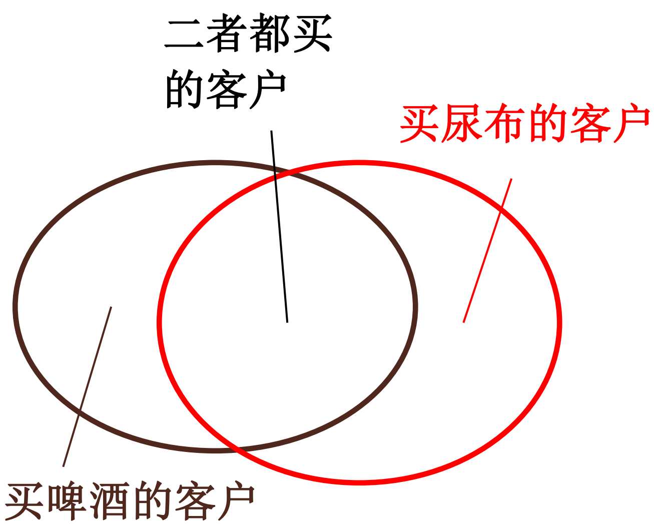
- **Association rules**
  - Antecedent → Consequent [support, confidence]
  - 前项 → 后项 [支持度, 置信度]
  - buys(x, “diapers”)→buys(x, “beers”) [0.5%, 60%]
  - major(x, “SE”)^takes(x, “AI”)→grade(x, “A”) [1%, 75%]

$$Support, s(A \to C) = s(C \to A) = p(A, C)$$

$$Confidence, c(A \to C) = p(A, C) / p(A)$$

- **Applications**
  - Cross-selling, Customer relationship management
  - Inventory management, Marketing promotions
  - Classification & Clustering…

Association Rule Mining



- *Rules:* $X \,\&\, Y \Rightarrow Z$
  满足最小支持度和置信度
  - 支持度, *s*, 一次交易中包含{X 、 Y、 Z}的可能性
  - 置信度, *c*, 包含{X 、 Y}的交易中也包含Z的条件概率

交易ID	购买的商品
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

设最小支持度为50%, 最小置信度为 50%, 则可得到

- $A \Rightarrow C$ (50%, 66.6%)
- $C \Rightarrow A$ (50%, 100%)

Association Rule Mining

交易ID	购买商品
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

最小支持度 (*minsup*) 50%
最小置信度 (*minconf*) 50%

频繁项集	支持度
{A}	75%
{B}	50%
{C}	50%
{A,C}	50%

对于 $A \Rightarrow C$：

$$\text{support} = \text{support}(\{A, C\}) = 50\%$$

$$\text{confidence} = \text{support}(\{A, C\}) / \text{support}(\{A\}) = 66.6\%$$

Key Step: Get Frequent Itemset

- *Frequent Itemset*: 满足最小支持度 (*minsup*) 的项目集合
  - 频繁项集的子集一定是频繁的
    - 例如，如果{A,B}是频繁项集，则{A}、{B}也一定是频繁项集
  - 从1到k (k-频繁项集)递归查找所有频繁项集
- 用得到的频繁项集生成所有关联规则
  - 应满足最小置信度 (*minconf*)

Apriori Algorithm

- 自连接: 用 L_{k-1} 自连接得到 C_k
- 修剪: 一个k-项集，如果他的一个k-1项集（他的子集）不是频繁的，那他本身也不可能是频繁的。
- pseudo code:

C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k !=$\varnothing$; k++) **do begin**
    C_{k+1} = candidates generated from L_k;
    for each transaction t in database do
        increment the count of all candidates in C_{k+1} that are contained in t
    L_{k+1} = candidates in C_{k+1} with *minsup*
    end
return $\cup_k L_k$;

Apriori Algorithm

Database D

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

Scan D

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

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

Scan D

C_2

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

L_2

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

C_3

itemset
{2 3 5}

Scan D

L_3

itemset	sup
{2 3 5}	2

Apriori Algorithm

- 假定 L_{k-1} 中的项按顺序排列
- 第一步: 自连接 L_{k-1}

insert into C_k

select $p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$

from $L_{k-1}\ p, L_{k-1}\ q$

where $p.item_1=q.item_1, \ldots, p.item_{k-2}=q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$

- 第二步: 修剪

For all ***itemsets c in*** C_k do

For all ***(k-1)-subsets s of c*** do

if *(s is not in* L_{k-1}*)* **then delete** *c* **from** C_k

Apriori Algorithm

- L_3={*abc*, *abd*, *acd*, *ace*, *bcd*}
- 自连接 : L_3*L_3
  - *abc* 和 *abd* 得到 *abcd*
  - *acd* 和 *ace* 得到 *acde*

Apriori Algorithm

- L_3={*abc, abd, acd, ace, bcd*}
- 自连接 : L_3*L_3
  - *abc* 和 *abd* 得到 *abcd*
  - *acd* 和 *ace* 得到 *acde*
- 修剪:
  - *ade* 不在 L_3中，删除 *acde*
- C_4={*abcd*}

Apriori Algorithm

- *Apriori*的核心
  - 用频繁的$(k-1)$-项集生成候选的频繁 k-项集
  - 用数据库扫描和模式匹配计算候选项集的支持度
- *Apriori* 的瓶颈
  - 巨大的候选项集
    - 10^4 个频繁1-项集要生成 10^7 个候选 2-项集
    - 要找尺寸为100的频繁模式，如 $\{a_1, a_2, \ldots, a_{100}\}$, 你必须先产生$2^{100} \approx 10^{30}$ 个候选集
  - 多次扫描数据库

Rule Generation

- Let Y={a,b,c} be a frequent itemset.
- There are six candidate association rules that can be generated from Y
  - $\{a,b\} \rightarrow \{c\}$
  - $\{a,c\} \rightarrow \{b\}$
  - $\{b,c\} \rightarrow \{a\}$
  - $\{a\} \rightarrow \{b,c\}$
  - $\{b\} \rightarrow \{a,c\}$
  - $\{c\} \rightarrow \{a,b\}$
- Compare their confidence with *minconf*

Compact Representation

- The number of frequent itemsets produced from a transaction data set can be very large.
- It is useful to identify a small representative set of frequent itemsets from which all other frequent itemsets can be derived.
- Two compact representations are
  - Maximal frequent itemsets
  - Closed frequent itemsets

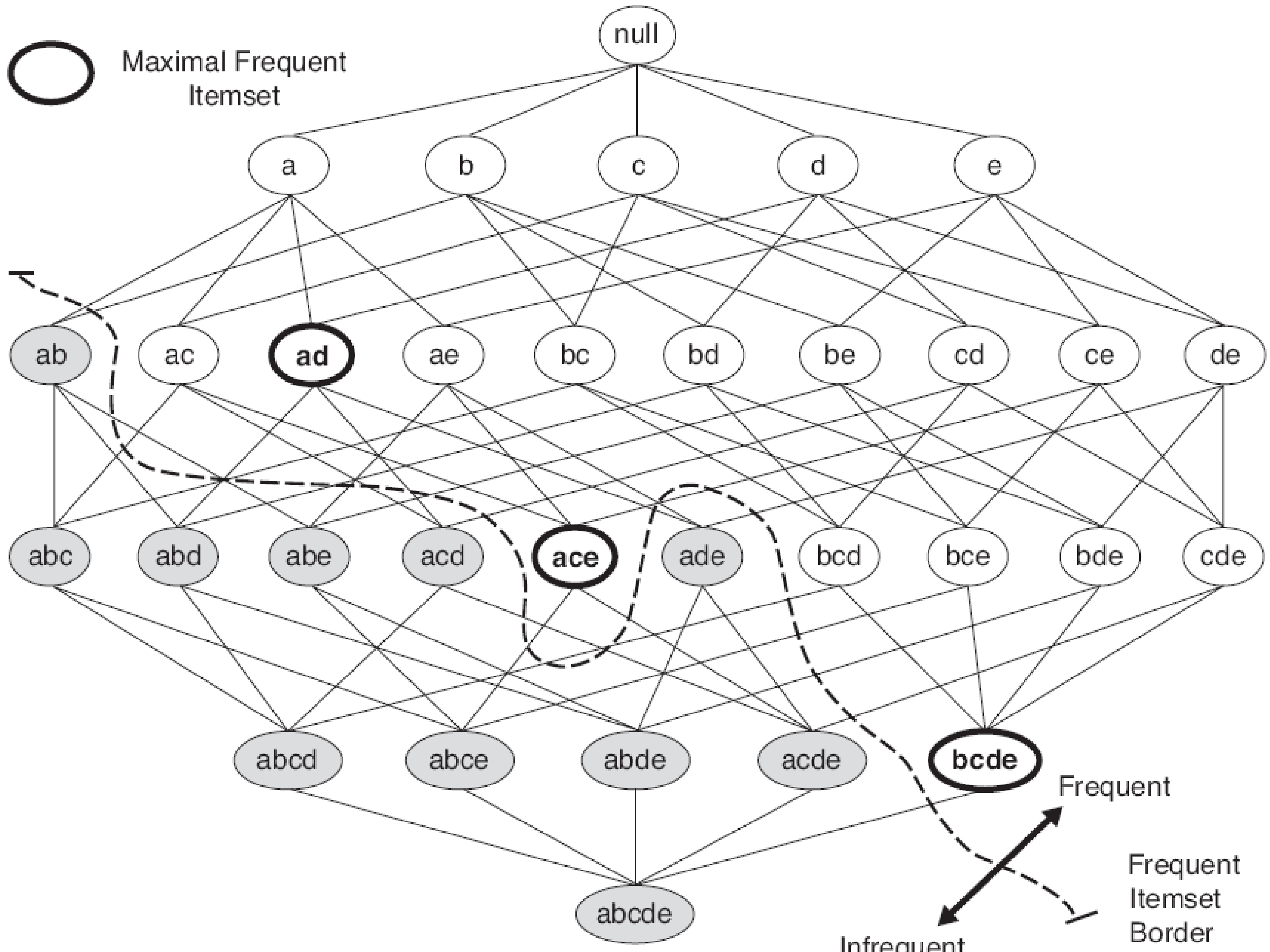
Maximal Frequent Itemsets

- A maximal frequent itemset is defined as a frequent itemset for which none of its immediate supersets are frequent.

- We consider the itemset lattice shown in the following figure.

- The itemsets in the lattice are divided into two groups
  - Those that are frequent
  - Those that are infrequent

Maximal Frequent Itemsets



Maximal Frequent Itemsets

- {a,d}, {a,c,e} and {b,c,d,e} are considered to be maximal frequent itemsets.
  - This is because their immediate supersets are infrequent.
- {a,c} is non-maximal because one of its immediate supersets, {a,c,e}, is frequent.

Maximal Frequent Itemsets

- Maximal frequent itemsets do not contain the support information of their subsets.
- An additional pass over the database is required to determine the support counts of the non-maximal frequent itemsets.

Closed Frequent Itemsets

- An itemset X is closed if none of its immediate supersets has exactly the same support count as X.

- In other words, X is not closed if at least one of its immediate supersets has the same support count as X.

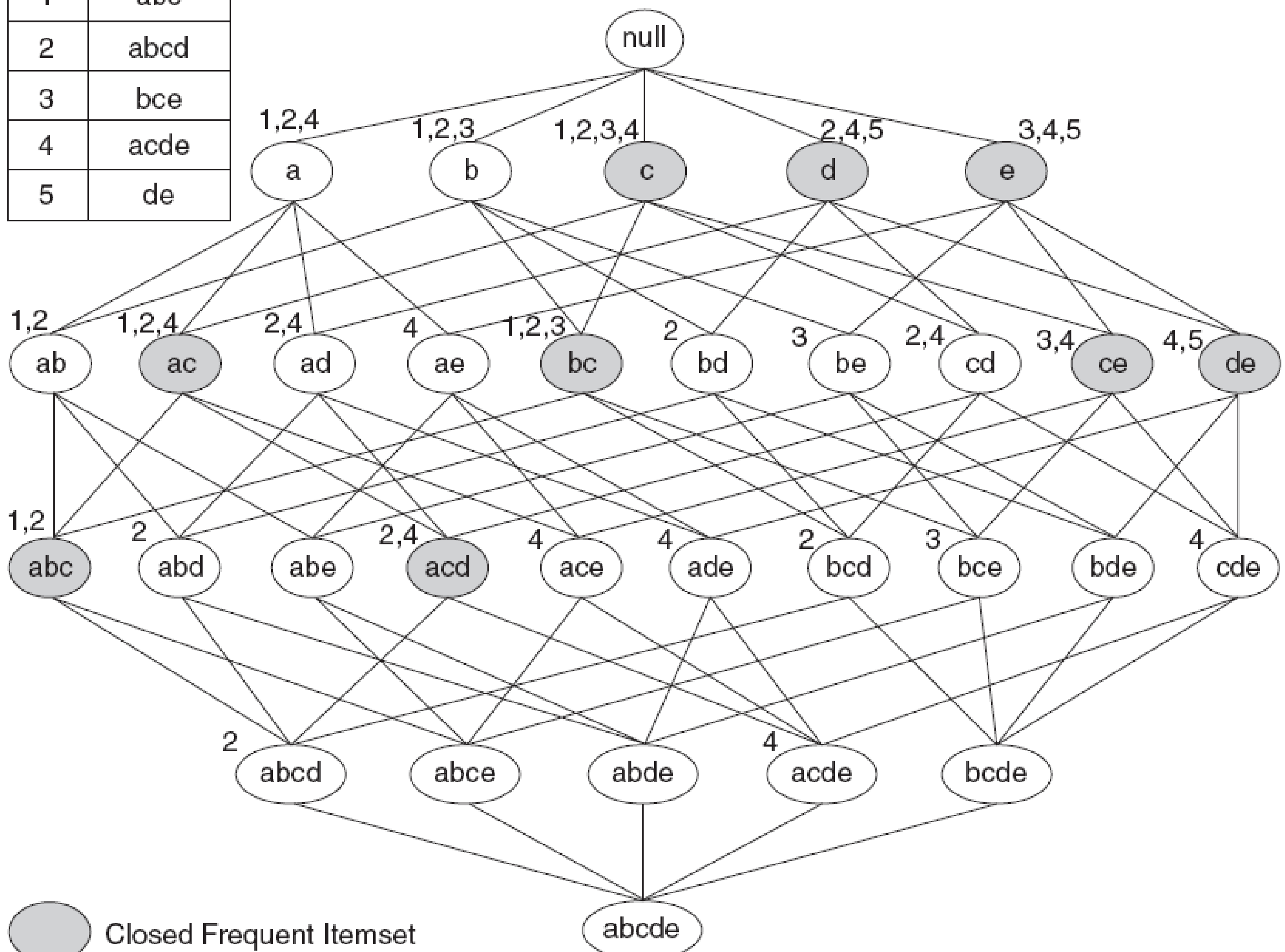
Closed Frequent Itemsets

- Examples of closed itemsets are shown in the following figure.
- Each node (itemset) in the lattice is associated with a list of its corresponding TIDs.

Closed Frequent Itemsets

TID	Items
1	abc
2	abcd
3	bce
4	acde
5	de

minsup = 40%



Closed Frequent Itemsets

- We notice that every transaction that contains b also contains c.
- Consequently, the support for {b} is identical to {b,c}.
- {b} should not be considered a closed itemset.

Closed Frequent Itemsets

- Similarly, the itemset {a,d} is not closed, since c occurs in every transaction that contains both a and d.
- On the other hand, {b,c} is a closed itemset.
  - This is because it does not have the same support count as any of its supersets.

Closed Frequent Itemsets

- An itemset is a closed frequent itemset if
  - It is closed and
  - Its support is greater than or equal to *minsup*.
- In the previous example, assuming that the support threshold is 40%.
- {b,c} is a closed frequent itemset because its support is 60%.
- The rest of the closed frequent itemsets are indicated by the shaded nodes.

Closed Frequent Itemsets

- We can use the closed frequent itemsets to determine the support counts for the non-closed frequent itemsets.
- For example, we consider the frequent itemset {a,d} shown in the figure.
- Because the itemset is not closed, its support count must be identical to one of its immediate supersets.
- The key is to determine which superset (among {a,b,d}, {a,c,d} or {a,d,e}) has exactly the same support count as {a,d}.

Closed Frequent Itemsets

- Any transaction that contains the superset of {a,d} must also contain {a,d}.

- However, any transaction that contains {a,d} does not have to contain the supersets of {a,d}.

- For this reason, the support for {a,d} must be equal to the largest support among its supersets.

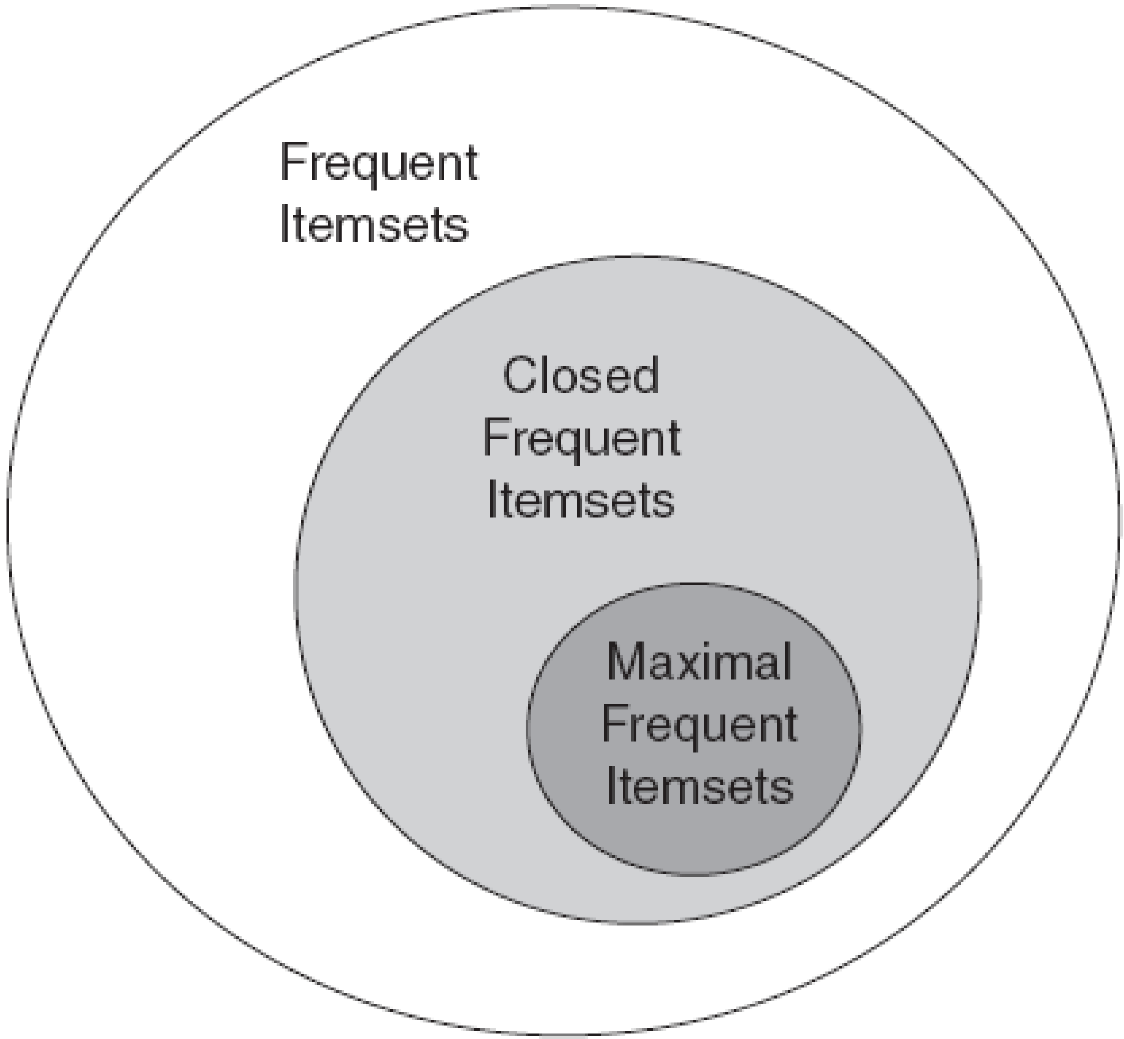
Closed Frequent Itemsets

- {a,c,d} has a larger support than both {a,b,d} and {a,d,e}.
- As a result, the support for {a,d} must be identical to the support for {a,c,d}.
- To find the support for a non-closed frequent itemset, the support for all of its supersets must be known.

Closed Frequent Itemsets

- All maximal frequent itemsets are closed.
- This is because none of the maximal frequent itemsets can have the same support count as their immediate supersets.
- The relationship among frequent, maximal frequent, and closed frequent itemsets are shown in the following figure.

Summary


Frequent
Itemsets
Closed
Frequent
Itemsets
Maximal
Frequent
Itemsets