人工智能
——关联规则挖掘

Yanghui Rao
Assistant Prof., Ph.D
School of Data and Computer Science,
Sun Yat-sen University
raoyangh@mail.sysu.edu.cn

关联规则挖掘

- 商家通常对于用户的购买行为感兴趣

关联规则挖掘

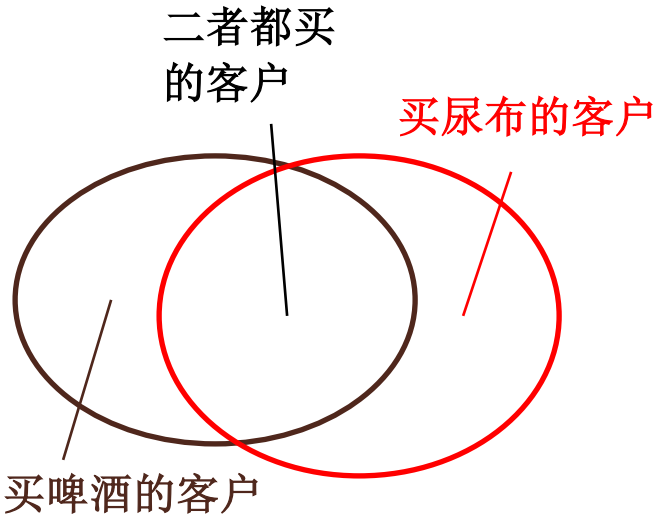
- **关联规则**
  - 前项(Antecedent) → 后项(Consequent) [支持度(support), 置信度(confidence)]
  - buys(x, “diapers”)→buys(x, “beers”) [0.5%, 60%]
  - major(x, “SE”)^takes(x, “AI”)→grade(x, “A”) [1%, 75%]

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

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

- **应用**
  - 绑定销售, 客户关系管理
  - 库存管理, 营销额提升
  - 分类 & 聚类…

关联规则挖掘



- *Rules: X & Y ⇒ 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%)

关联规则挖掘

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 $A \Rightarrow C$：

support = support({A ,C}) = 50%

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

频繁项集

- 频繁项集(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算法

- 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。
- 伪代码:

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

L_1 = {frequent items};
for (k = 1; L_k !=∅; 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算法

最小支持度计数：2

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算法

- 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算法

- 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算法

- 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算法

- *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

规则生成

- 假设Y={a,b,c}是一个频繁项集
- 则可以从Y产生六个候选关联规则
  - {a,b}→{c}
  - {a,c}→{b}
  - {b,c}→{a}
  - {a}→{b,c}
  - {b}→{a,c}
  - {c}→{a,b}
- 将他们的置信度与最小置信度(*minconf*)比较

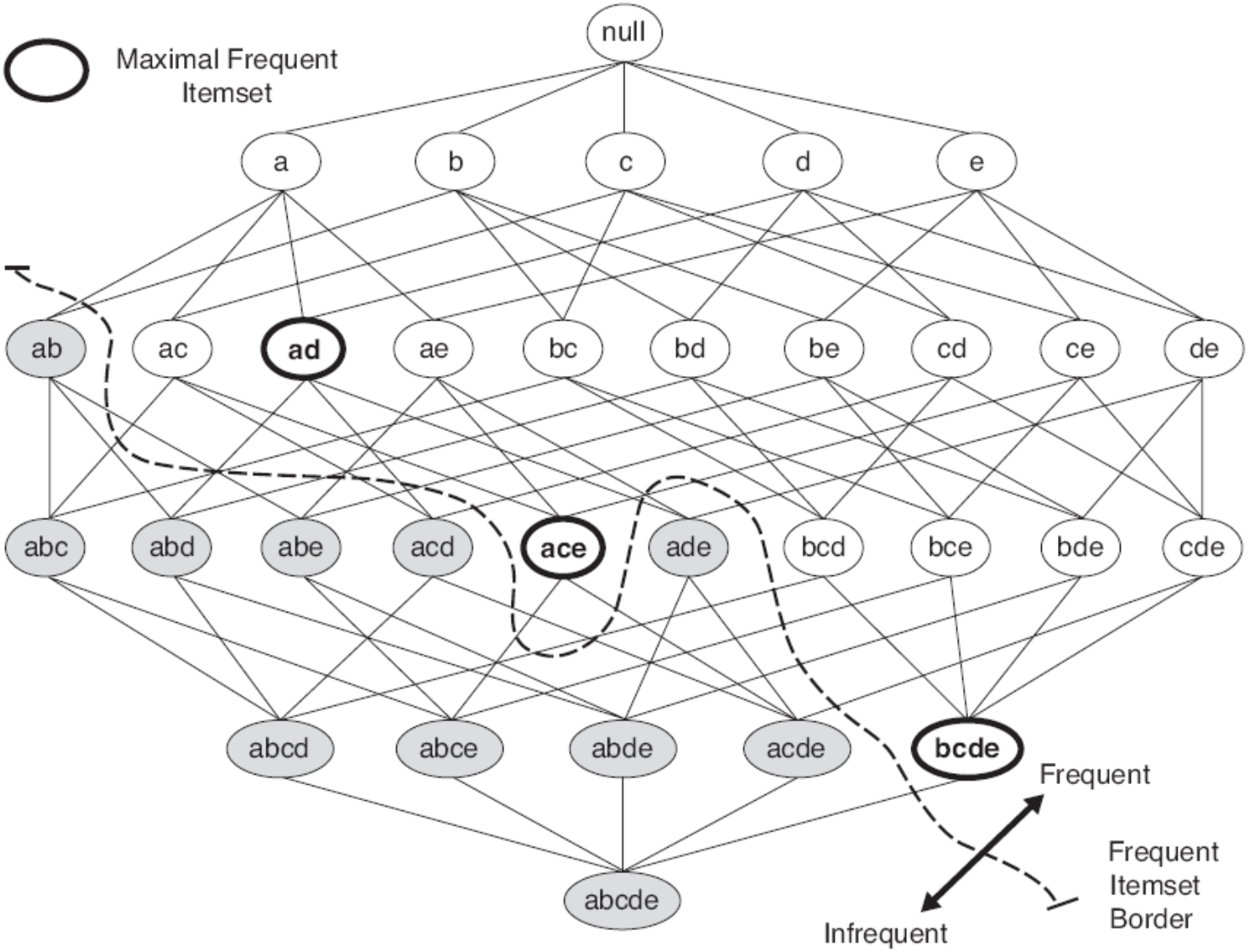
压缩表示

- 从一个交易数据集中产生的频繁项集的数量通常会十分巨大
- 希望找出一种表示方法，能占用较少的空间，存储较少的项集，并且同时能够派生（或者推演）出其他所有的频繁项集
- 两种压缩表示方法
  - 最大频繁项集(Maximal frequent itemsets)
  - 闭频繁项集(Closed frequent itemsets)

最大频繁项集

- 最大频繁项集：首先其本身是频繁项集，其次，它的直接超集都不是频繁项集
- 通常采用项集网格图考虑会更加直观
- 在下面的项集网格图中，有两种项集
  - 频繁项集
  - 非频繁项集

Maximal Frequent Itemsets



最大频繁项集

- {a,d}, {a,c,e} 和 {b,c,d,e} 都是最大频繁项集，因为他们的直接超集都不频繁（都不是频繁项集）并且他们自己都是频繁项集。
- {a,c} 不是最大频繁项集，因为它的一个直接超集 {a,c,e}是频繁的。

最大频繁项集

- 最大频繁项集不包含他们子集的支持度信息。
- 所以，为了得到非最大频繁项集的支持数目，还需要在数据库中遍历一次。

闭频繁项集

- 闭频繁项集：一个频繁项集X，如果它的直接超集中，没有一个项集和X的支持数目相同，那X就是一个闭频繁项集

- 也就是说，如果项集X的直接超集中，存在至少一个项集和项集X的支持数目相同，那项集X不是一个闭项集

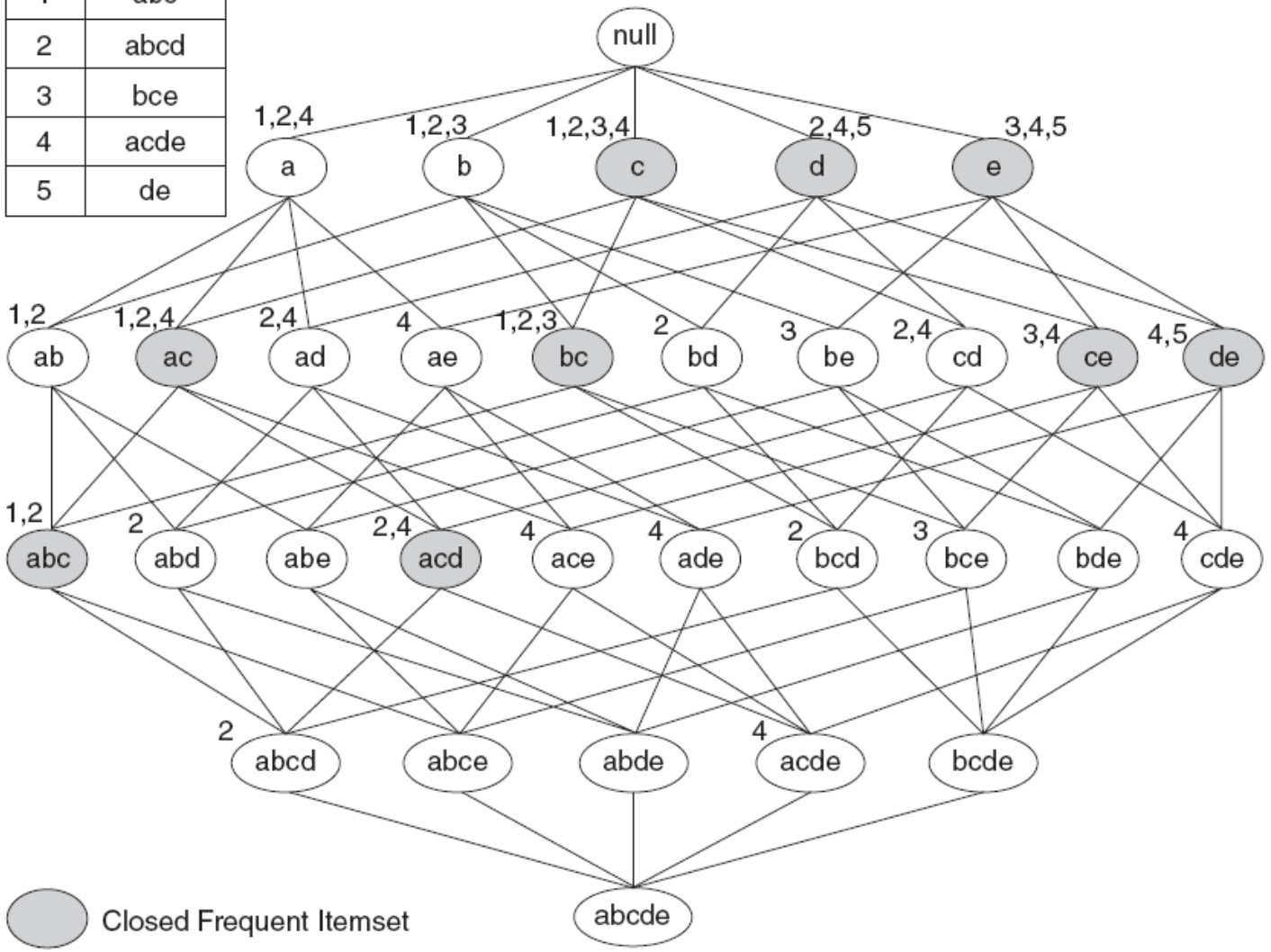
闭频繁项集

- 下图中展示了几个闭频繁项集的例子
- 下图的网格中，每一个节点（项集），都有一个TID的列表与它对应。

闭频繁项集

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

minsup = 40%



闭频繁项集

- 注意包含b的每个交易，同时都包含了c
- 因此， {b} 的支持度和{b,c}相同.

- 所以{b} 不是一个闭项集

闭频繁项集

- 同理, 项集{a,d} 不是闭项集, 因为包含了a,d的所有交易中，同时都包含了c.
- {b,c} 是闭项集
- 因为{b,c}的直接超集中，没有任何一个与它有相同的支持数目。

闭频繁项集

- 满足以下条件的项集是闭频繁项集
  - 它是一个闭项集
  - 支持度大于最小支持度阈值（频繁）.
- 在前一个例子中，假设最小支持度阈值是40%.
- {b,c} 是闭频繁项集，因为它的支持度是60%>40%，而且它是一个闭项集.
- 闭频繁项集在前面的网格图中，都用阴影表示

闭频繁项集

- 我们可以通过闭频繁项集，来得到非闭频繁项集的支持数目
- 例如，考虑图中的{a,d}.
- 因为{a,d}不是闭项集, 它的支持数目一定与某一个它的直接超集的支持数目相同.
- 所以，关键是要找出与{a,d}支持数目相同的直接超集是它的所有直接超集({a,b,d},{a,c,d},{a,d,e})中的哪一个

闭频繁项集

- 包含了{a,d}的超集的项，一定会包含{a,d}
- 但是，包含{a,d}的项，不一定会包含{a,d}的超集
- 因此，{a,d}的支持数目=={a,d}的超集的最大支持数目

闭频繁项集

- {a,c,d} 的支持数目>{a,b,d}的支持数目
- {a,c,d} 的支持数目>{a,d,e}的支持数目

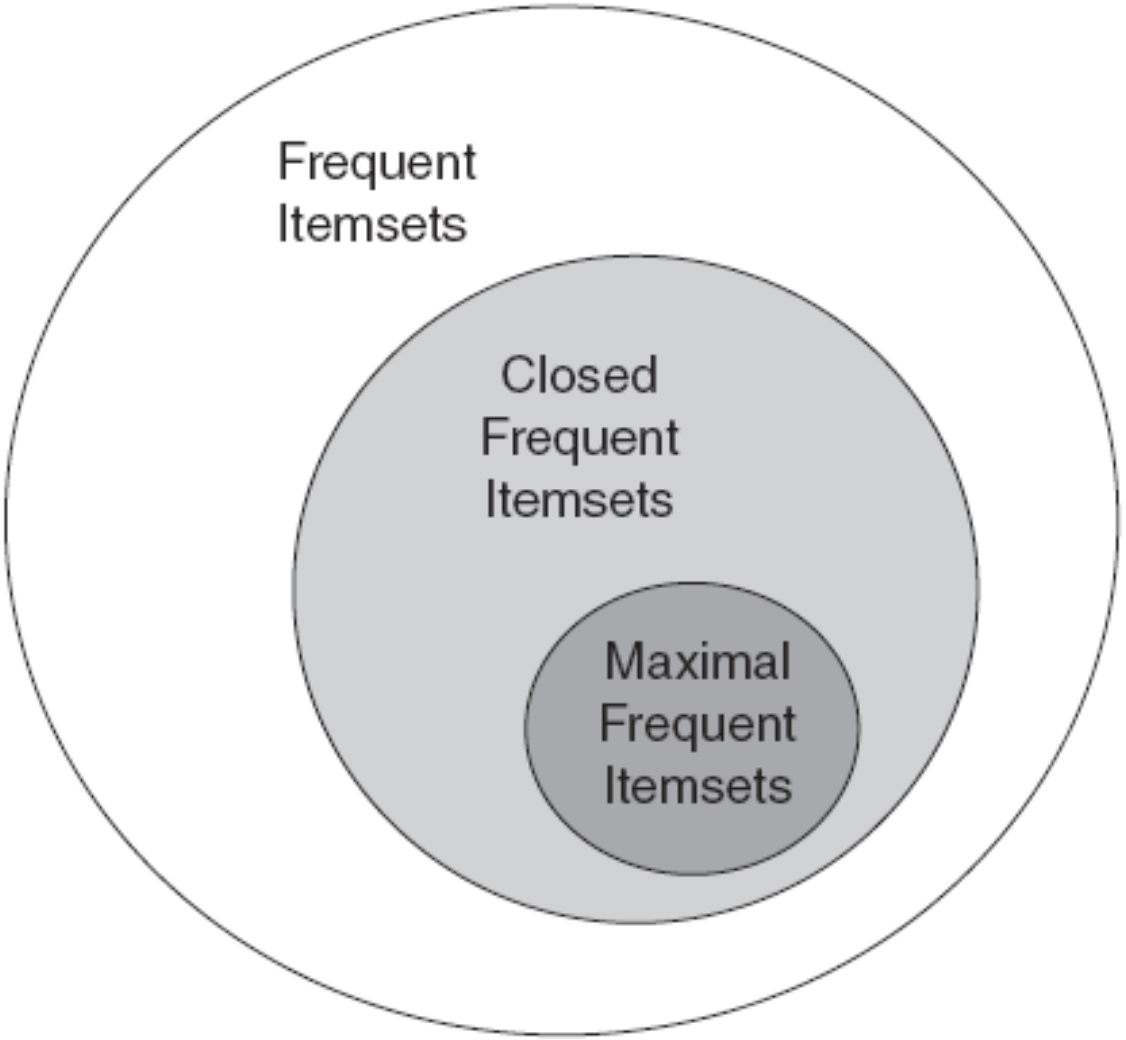
- 因此，{a,d}的支持数目=={a,c,d}的支持数目

- 为了求某一非闭项集的支持数目，需要先知道它的所有超集的支持数目

闭频繁项集

- 所有的最大频繁项集都是闭频繁项集
- 因为最大频繁项集的直接超集不可能和它的直接超集有相同的支持数目。
- 频繁项集，最大频繁项集，闭频繁项集的关系如下图所示。

总结


Frequent
Itemsets
Closed
Frequent
Itemsets
Maximal
Frequent
Itemsets