# 关联规则挖掘基本过程

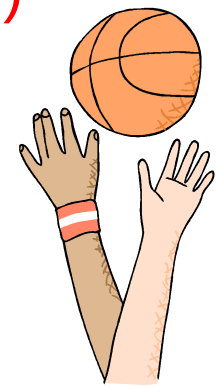
- 关联规则挖掘的目标：强关联规则，满足最小支持度和最小信任度的关联规则，Strong Association Rule。

- 关联规则挖掘问题可以划分成两个子问题：
  - 1、发现所有的频繁项集
    - Apriori算法
    - close算法
    - Fp-tree算法
    - CHARM算法
  - 2、从频繁项集中发现强关联规则

# 第三章 关联规则挖掘理论和算法

内容提要

- 基本概念与解决方法
- 经典的频繁项目集生成算法分析
- Apriori算法的性能瓶颈问题
- Apriori的改进算法
- 对项目集格空间理论的发展
- 基于项目序列集操作的关联规则挖掘算法（可选）
- 改善关联规则挖掘质量问题
- 约束数据挖掘问题
- 关联规则挖掘中的一些更深入的问题
- 数量关联规则挖掘方法



# 项目序列集概念

- “项目序列（Itemsequence）”来替代大多数文献中出现的“项目集（Itemset）”可以简化挖掘的过程。
- 所谓项目序列是指项目集中的元素按着某种标准进行有序排列。例如，我们可以按项目名称的字典顺序排列，也可以象FP-Tree算法那样，按它们在数据库中出现次数的多少降序排列。
- 为了重复利用对数据库的扫描信息，把来自数据库的信息组织成项目序列集（Set of itemsequences）形式，并且对项目序列集格及其操作代数化。在这样的代数系统下研究适应关联规则挖掘问题的操作算子．

# 项目序列集格

- 定义3-5 一个项目序列集格空间可以用三元组（I，S，p）来刻画，其中含义如下：
  - 项目定义域I：I={ $i_1$，$i_2$，…，$i_m$ }为所有项目集；
  - 项目序列集变量集S：S中的每个项目序列集变量形式为ISS={$IS_1$，$IS_2$，…，$IS_n$}，其中$IS_i$（i=1，2，…，n）是定义在I上项目序列；
  - 操作p ：关于S中的项目序列集变量的操作集。

- 定义3-6 项目序列集间上的属于（∈）、包含（⊆）、并（∪）、交（∩）、差（－）等操作和普通的集合操作相同。

# 项目序列集格

- **定义3-6** 项目序列集间上的属于（$\in$）、包含（$\subseteq$）、并（$\cup$）、交（$\cap$）、差（$-$）等操作和普通的集合操作相同。
  - **例：** 设$ISS_1=\{AB, CD\}$和$ISS_2=\{ABCD, AD\}$是定义在$I=\{A, B, C, D\}$上的项目序列集，则
  - $AB \in ISS_1$；是否成立？
  - $AB \in ISS_2$；是否成立？ $AB \notin ISS_2$；
  - $\{AB\} \subset ISS_1$；是否成立？
  - $\{AB\} \subset ISS_2$；是否成立？ $\{AB\} \not\subset ISS_2$；
  - $ISS_1 \cup ISS_2$=?
    - {AB, CD, ABCD, AD};
  - $ISS_1 \cap ISS_2$=?
    - Ø

# 项目序列集格上的亚操作

- 定义3-7 设$ISS_1$和$ISS_2$是定义在I上的两个项目序列集，IS是定义在I上的一个项目序列，定义如下操作：
  - 亚属于（$\in_{sub}$）：$IS\in_{sub} ISS_1$当且仅当$\exists\ IS_1\in ISS_1$使得$IS\subseteq IS_1$；
  - 亚包含（$\subseteq_{sub}$）：$ISS_1\subseteq_{sub} ISS_2$当且仅当 $\forall\ IS_1\in ISS_1\Rightarrow\ IS_1\in_{sub} ISS_2$；
  - 亚交（$\cap_{sub}$）：$ISS_1\cap_{sub} ISS_2=\{IS \mid IS\in_{sub} ISS_1$且$IS\in_{sub} ISS_2\}$；
  - 亚并（$\cup_{sub}$）：$ISS_1\cup_{sub} ISS_2=\{IS \mid IS\in_{sub} ISS_1$或$IS\in_{sub} ISS_2\}$。

# 项目序列集格

- 亚属于（$\in_{sub}$）：$IS \in_{sub} ISS_1$ 当且仅当 $\exists IS_1 \in ISS_1$ 使得 $IS \subseteq IS_1$；

  - **例：** 设 $ISS_1=\{AB, CD\}$ 和 $ISS_2=\{ABCD, AD\}$ 是定义在 $I=\{A, B, C, D\}$ 上的项目序列集，则
  - $AB \in ISS_1$；
  - $AB \notin ISS_2$；
  - $AB \in_{sub} ISS_2$；是否成立？

# 项目序列集格上的亚操作

- 定义3-7 设$ISS_1$和$ISS_2$是定义在I上的两个项目序列集，IS是定义在I上的一个项目序列，定义如下操作：
  - 亚属于（$\in_{sub}$）：$IS\in_{sub} ISS_1$当且仅当$\exists\ IS_1\in ISS_1$使得$IS\subseteq IS_1$；
  - 亚包含（$\subseteq_{sub}$）：$ISS_1\subseteq_{sub} ISS_2$当且仅当 $\forall\ IS_1\in ISS_1\Rightarrow\ IS_1\in_{sub} ISS_2$；
  - 亚交（$\cap_{sub}$）：$ISS_1\cap_{sub} ISS_2=\{IS \mid IS\in_{sub} ISS_1$且$IS\in_{sub} ISS_2\}$；
  - 亚并（$\cup_{sub}$）：$ISS_1\cup_{sub} ISS_2=\{IS \mid IS\in_{sub} ISS_1$或$IS\in_{sub} ISS_2\}$。

# 项目序列集格

- 亚包含（$\subseteq_{sub}$）： $ISS_1 \subseteq_{sub} ISS_2$当且仅当 $\forall IS_1 \in ISS_1 \Rightarrow IS_1 \in_{sub} ISS_2$;

- **例：** 设$ISS_1=\{AB, CD\}$和$ISS_2=\{ABCD, AD\}$是定义在$I=\{A, B, C, D\}$上的项目序列集,则
- $AB \in ISS_1$;
- $AB \notin ISS_2$;
- $AB \in_{sub} ISS_2$; 是否成立？
- $\{AB\} \subset ISS_1$;
- $\{AB\} \not\subset ISS_2$;
- $ISS_1 \subseteq_{sub} ISS_2$; 是否成立？

# 项目序列集格上的亚操作

- 定义3-7 设$ISS_1$和$ISS_2$是定义在I上的两个项目序列集，IS是定义在I上的一个项目序列，定义如下操作：
  - 亚属于（$\in_{sub}$）：$IS\in_{sub} ISS_1$当且仅当$\exists\ IS_1\in ISS_1$使得$IS\subseteq IS_1$；
  - 亚包含（$\subseteq_{sub}$）：$ISS_1\subseteq_{sub} ISS_2$当且仅当 $\forall\ IS_1\in ISS_1\Rightarrow\ IS_1\in_{sub} ISS_2$；
  - 亚交（$\cap_{sub}$）：$ISS_1\cap_{sub} ISS_2=\{IS \mid IS\in_{sub} ISS_1$且$IS\in_{sub} ISS_2\}$；
  - 亚并（$\cup_{sub}$）：$ISS_1\cup_{sub} ISS_2=\{IS \mid IS\in_{sub} ISS_1$或$IS\in_{sub} ISS_2\}$。

# 项目序列集格

- 亚交（$\cap_{sub}$）：$ISS_1 \cap_{sub} ISS_2 = \{IS \mid IS \in_{sub} ISS_1$且$IS \in_{sub} ISS_2\}$；
- 亚并（$\cup_{sub}$）：$ISS_1 \cup_{sub} ISS_2 = \{IS \mid IS \in_{sub} ISS_1$或$IS \in_{sub} ISS_2\}$。

- **例：**设$ISS_1 = \{AB, CD\}$和$ISS_2 = \{ABCD, AD\}$是定义在$I = \{A, B, C, D\}$上的项目序列集，则

- $ISS_1 \cap_{sub} ISS_2$；$ISS_1 \cup_{sub} ISS_2$ ……

# 基于项目序列集操作的关联规则挖掘算法

**算法3-17** ISS-DM Algorithm

输入：数据库*D*， minsup_count

输出：最大频繁项目序列集ISS*

(1) Input（minsup_count）；

(2) ISS ← Ø; ISS* ← Ø;　　//两个线性数据结构

(3) FOR all IS∈*D* DO BEGIN //取D的一个项目序列IS

(4)　join（IS，ISS）；　　//加入项目集格

(5)　make_fre（IS，ISS，ISS*）；//频繁项目集格生成

(6) END

(7) Answer←ISS*

- join（IS，ISS）完成数据库中一个项目序列（元组）加入项目序列集后，它及它的子项目序列的频度维护。算法3-14
- make_fre（IS，ISS，ISS*）从ISS挑选频繁的并加入到ISS*。算法3-15

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| | | | | |
| | | | | |
| | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（BCE，1），（ABCE，1）} | | |
| 4 | | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（BCE，1），（ABCE，1）} | {BC, ABC，BCE} | |
| | | | | |
| | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（BCE，1），（ABCE，1）} | {BC, ABC，BCE} | |
| | | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | | | |
| 5 | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | | |
| | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| | | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| 5 | ABCD | | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| 5 | ABCD | {（ABCD，2），（ABCE，1），（BDE，1）} | | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| 5 | ABCD | {（ABCD，2），（ABCE，1），（BDE，1）} | {ABC，BCE，BD,ABCD} | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| 5 | ABCD | {（ABCD，2），（ABCE，1），（BDE，1）} | {ABC，BCE，BD,ABCD} | |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（BCE，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| 5 | ABCD | {（ABCE，1），（BDE，1）} | {ABCD，BCE} | 裁*ABC;BD;ABCD |

# ISS-DM例子

| 操作 | IS | ISS | 频繁ISS* | 说明 |
|---|---|---|---|---|
| 初始 | | Ø | Ø | |
| 1 | ABCD | {（ABCD，1）} | Ø | |
| 2 | BCE | {（ABCD，1），（BCE，1）} | {BC} | |
| 3 | ABCE | {（ABCD，1），（BCE，1），（ABCE，1）} | {ABC，BCE} | 裁*BC;BCE |
| 4 | BDE | {（ABCD，1），（ABCE，1），（BDE，1）} | {ABC，BCE，BD} | |
| 5 | ABCD | {（ABCD，2），（ABCE，1），（BDE，1）} | {ABCD，BCE} | 裁*ABC;BD;ABCD |

# 小结

- 关联规则挖掘问题可以划分成两个子问题：
  - 1. 发现频繁项集：
    Apriori, Close, FP-tree, 项目集格
  - 2、从频繁项集中生成关联规则：通过用户给定Minconfidence，在频繁项目集中，寻找关联规则。

# 生产关联规则

规则S→Y，项集S为该规则的前件，Y为后件。

对于任一个频繁项集X和它的一个非空真子集Y，假设S=X-Y，检验S→Y 的置信度confidence(S→Y)是否大于等于最小置信度minconf。

· 如果满足，则规则S→Y成立，输出该规则；

· 如果不满足，则规则S→Y不成立，不予输出。

- “不重复，不遗漏”，“高效”

## 基于子规则的关联规则生成算法

关联规则是数据挖掘中重要的课题之一.传统的由频繁项目集产生关联规则的方法由于要考虑频繁项目集的每一个非空真子集,当频繁项目集的长度较长时代价较大.文中提出...

任小龙， 吴耿锋， 赵朔 - 《计算机工程》 - 被引量: 10 - 2004年

来源： 知网 / 维普 / 万方 / 爱学术 / 掌桥科研

收藏 引用 批量引用

## 利用广义相关系数改进的关联规则生成算法

提出了一种改进的关联规则生成 算法 ,其目的是在大型数据库中能够高效的发现关联知识.为了达到这个目标 ,将泛逻辑中的广义相关系数与 Apriori算法相结合. Apriori...

周延泉， 何华灿， 李金荣 - 《西北工业大学学报》 - 被引量: 12 - 2001年

来源： 维普 / 万方 / 知网 / 爱学术 / 掌桥科研

收藏 引用 批量引用

## 利用广义相关系数改进的关联规则生成算法

提出了一种改进的关联规则生成算法,其目的是在大型数据库中能够高效的发现关联知识.为了达到这个目标,将泛逻辑中的广义相关系数与Apriori算法相结合.Apriori算法本...

周延泉， 何华灿， 李金荣 - 《西北工业大学学报》 - 被引量: 8 - 2001年

来源： wanfangdata.com.cn

收藏 引用 批量引用

## Source Association Rules Generation Algorithm源关联规则生成算法

一,引言 IBM科学家Rakesh Agrawal于1993年提出了用于交易的关联规则数据挖掘算法,该算法把基于关联规则的数据挖掘分为两大步,第一步,从交易中发现频繁项目集;第二...

李学明， 张伟， 彭军， ... - 《计算机科学》 - 被引量: 7 - 2002年

来源： OALib / jourlib.org / 爱学术

时间
2018年以来 ×
领域
计算机科学与... ( 136 )
信息与通信工程 ( 12 )
电气工程 ( 11 )
+
核心
中国科技核心... ( 243 )
北大核心期刊 ( 155 )
CSCD 索引 ( 46 )
+
获取方式
免费下载 ( 47 )
登录查看 ( 0 )
付费下载 ( 0 )
+
关键词
关联规则
APRIORI算法
关联规则挖掘
+
类型
期刊
学位
会议
+
作者
王莉
唐兴宏

## 一种改进的关联规则挖掘算法研究

传统的关联规则Apriori算法在产生频繁项集的过程中,需要多次扫描事务数据库以及多次扫描频繁项集,从而造成算法性能下降.为了减少扫描事务数据库以及频繁项集的次数...

刘林东， 齐德昱 - 《广东第二师范学院学报》 - 被引量: 1 - 2018年

来源： 知网 / wanfangdata.com.cn / 维普 / 爱学术 / kns.cnki.net

收藏 引用 批量引用

## 基于存储改进的分区并行关联规则挖掘算法

针对现有算法存储结构简单,生成大量冗余的候选集,时间和空间复杂度高,挖掘效率不理想的情况,为了进一步提高关联规则算法挖掘频繁集的速度,优化算法的执行性能,提出...

王永贵， 谢南， 曲海成 - 《计算机应用研究》 - 被引量: 0 - 2020年

来源： 知网 / 掌桥科研 / chinaxiv.org

收藏 引用 批量引用 免费下载

## 基于频次有效长度的加权关联规则挖掘算法研究

[目的]通过对数据库中项在重要程度上存在的差异性进行分析,解决传统关联规则挖掘算法挖掘大量冗余无价值规则的问题.[方法]在具有时态约束的序列上,结合频次有效长...

张勇， 李树青， 程永上 - 《现代图书情报技术》 - 被引量: 0 - 2019年

来源： 掌桥科研

收藏 引用 批量引用

## 集群环境下企业应用系统的关联规则算法研究

大数据在各个领域的快速发展,推动着企业不断地发展新业务和创造新的发展模式,企业大数据的应用和挖掘,成为企业提高竞争力的关键因素之一.关联规则作为数据挖掘研究...

赵向兵， 张景安 - 《山西大同大学学报:自然科学版》 - 被引量: 0 - 2018年

来源： 维普 / wanfangdata.com.cn / 知网 / 掌桥科研 / kns.cnki.net

收藏 引用 批量引用

# 第三章 关联规则挖掘理论和算法

内容提要

- 基本概念与解决方法
- 经典的频繁项目集生成算法分析
- Apriori算法的性能瓶颈问题
- Apriori的改进算法
- 对项目集格空间理论的发展
- 基于项目序列集操作的关联规则挖掘算法
- 改善关联规则挖掘质量问题
- 约束数据挖掘问题
- 关联规则挖掘中的一些更深入的问题
- 数量关联规则挖掘方法

# 衡量关联规则挖掘结果的有效性

- 应该从多种综合角度来考虑：
  - 准确性：挖掘出的规则必须反映数据的实际情况。
  - 实用性：挖掘出的规则必须是简洁可用的。
  - 新颖性：挖掘出的关联规则可以为用户提供新的有价值信息。

- 改善关联规则挖掘质量是一件很困难的工作。必须采用事先预防、过程控制以及事后评估等多种方法，其中使用合适的机制（如约束），让用户主动参与挖掘工作是解决问题的关键。粗略地说，可以在用户主观和系统客观两个层面上考虑关联规则挖掘的质量问题。

# 用户主观层面

- 一个规则的有用与否最终取决于用户的感觉。
- 用户可以在不同的层面、不同的阶段、使用不同的方法来主观设定约束条件。
- 从被约束的对象来看，有下面几种常用的方法：
  - **知识类型的约束：**针对应用问题选择有效的知识表达模式。例如，如果一个商业企业希望根据客户特点进行有针对性地销售，那么使用分类或聚类形式可以帮助用户形成客户群。
  - **数据的约束：**对数据的约束可以起到减少数据挖掘算法所用的数据量、提高数据质量等作用。
  - **维/层次约束：**限制聚焦的维数或粒度层次，也可以针对不同的维设置约束条件。
  - **知识内容的约束：**可以通过限定要挖掘的知识的内容，如指定单价大于10的交易项目，减少探索的代价和加快知识的形成过程。
  - **针对具体知识类型的约束：**针对具体知识类型的进行约束挖掘形式和实现机制的研究。

# 系统客观层面

- 使用“支持度-可信度” 的关联规则挖掘度量框架，在客观上也可能出现与事实不相符的结果。例如，“计算机游戏和录象产品是负相关的”问题。

- 重新考虑关联规则的客观度量问题。例如，
  - Brin等考虑的蕴涵规则（Implication Rule）；
  - Chen等给出的R-兴趣（R-Interesting）规则度量方法等。
  - 这些工作都期望通过引入新的度量机制和重新认识关联规则的系统客观性来改善挖掘质量。

# 第三章 关联规则挖掘理论和算法

内容提要

- 基本概念与解决方法
- 经典的频繁项目集生成算法分析
- Apriori算法的性能瓶颈问题
- Apriori的改进算法
- 对项目集格空间理论的发展
- 基于项目序列集操作的关联规则挖掘算法
- 改善关联规则挖掘质量问题
- 约束数据挖掘问题
- 关联规则挖掘中的一些更深入的问题
- 数量关联规则挖掘方法

# 约束在数据挖掘中的作用

- **聚焦挖掘任务，提高挖掘效率**：利用约束，可以把具体的挖掘任务转换成对系统工作的控制，从而使挖掘工作按着我们期望的方向发展。约束的使用可以在知识发现的任何阶段进行，快速聚焦挖掘任务，进而提高挖掘效率。

- **保证挖掘的精确性**：挖掘结果的精确性，不仅体现在它的可信程度，而且取决于它的有用性。约束的使用可以帮助我们发现问题，并及时加以调整，使知识发现的各个阶段按着正确的方向发展。

- **控制系统的使用规模**：数据挖掘和知识发现应用最常犯的错误就是无限制的扩大规模。约束数据挖掘的思想为系统的增量式扩充提供条件。当基本的原则和目标确定后，可以把一些有待验证和优化的问题以约束参数的形式交互式输入，通过实验找到最佳值。

# 约束的类型

- 一些常用的名词和符号：
  - 定义3-12　设项目集I={$i_1$，$i_2$，…，$i_m$}，事务数据库T=<tid，$I_t$>，模式S和S*都是项目集I的子集，如果S*⊆S，则称S*是S的子模式（Subpattern）；S是S*的超模式（Superpattern）。
  - 定义3-13　一个约束C是作用于项目集I的幂集（Powerset）上的谓词。约束C对于一个模式S的结果用布尔变量来表示，即C（S）=True/False：C（S）=True表示S满足约束条件；C（S）=False表示S不满足约束条件。
  - 定义3-14 对于被讨论的项目集I，满意模式集（Satisfying Pattern Set），记为SATc（I）是指那些完全满足约束C的项目集的全体。

- 将约束条件用于频繁集的查询无非是找出那些满足C的频繁集。
- 约束的常见类型有：
  - 单调性约束（Monotone Constraint）；
  - 反单调性约束（Anti-monotone Constraint）；
  - 可转变的约束（Convertibale Constraint）；
  - 简洁性约束（Succinct Constraint）。

# 单调性约束

- 定义3-15 所谓一个约束Cm是单调性的约束是指满足Cm的任何项目集S的超集也能满足 Cm。

- 例如，“sum（price）>100”的约束是一个单调性约束。因为一个项目集满足这个条件，那么它的超集也一定满足这个条件。现实世界中有许多条件满足这样的性质，如“属于”、“包含”等。

# 反单调性约束

- 定义3-16 约束Ca是反单调的是指对于任意给定的不满足Ca的项目集S，不存在S的超集能够满足Ca。
- 例如，“sum（price）<100”是反单调的，因为一个人买了大于或等于100元以上的东西，再加上新的物品只能使总和增大，也不可能满足这个条件。
- 这样的约束可以用来裁减不必要的探索，提高挖掘效率。

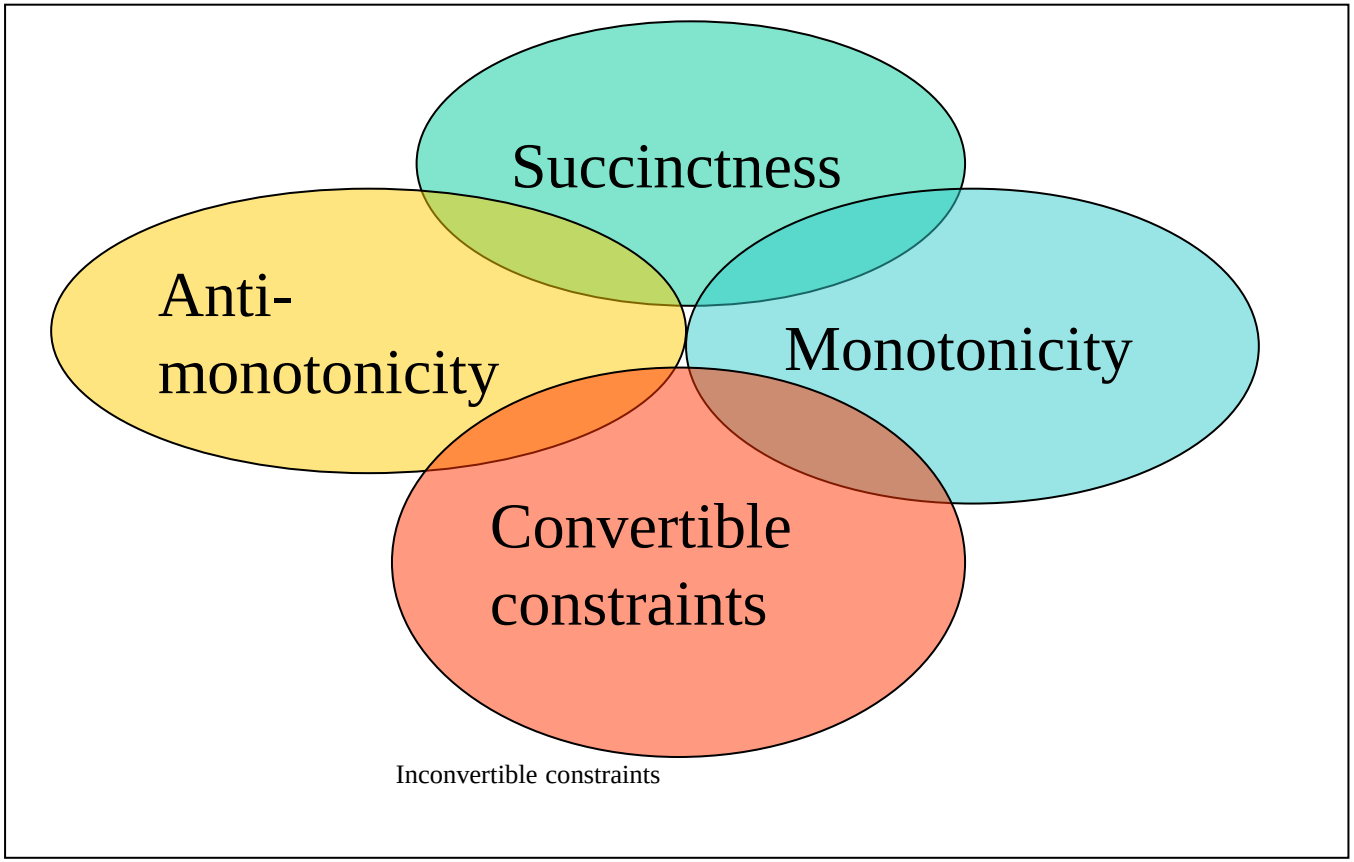
# 可转变的约束

- 定义3-17 如果一个约束C满足下面的条件，那么称它是反单调可转变的：
  - C（S）既不是单调性约束，也不是反单调性约束；
  - 若存在顺序R，使得经R排序后的I具有如下满足：任给S*∈{suffix_S}，有C（S）=>C（S*）。
- 例如，对于Avg（S）≥v，令I为一组以升序排列数值的项目集。如果S满足约束，那么S的后缀S*也满足avg（S*）≥v。
- 定义3-18 如果一个约束C满足下面的条件，那么称它是单调可转变的：
  - C（S）既不是单调性约束，也不是反单调性约束；
  - 若存在顺序R，使得经R排序后的I满足：任给 S*∈{suffix_S}，有C（S*）=>C（S）。
- 例如，对于Avg（S）≥v，令I为一组以降序排列数值的项目集。如果S的后缀S*满足约束avg（S*）≥v，那么S也满足avg（S）≥v。

# 简洁性约束

- 定义3-19 一个项目子集ls是一个简洁集（Succinct Set），如果对于某些选择性谓词p，该项目子集能够表示为$\sigma_p(l)$的形式，其中σ是选择符。
- 如果一个约束是简洁的，那么我们就可以直接使用SQL查询来得到满足条件的集合。在挖掘的不同阶段可以尽量尝试简洁性的约束，这样可以避免不必要的测试。
- 定义3-20 $SP \subseteq 2^I$是一个强简洁集（Succinct Power Set），如果有一个数目不变的简洁集$l_1$，$l_2$，…，$l_k \subseteq I$，SP能够用$l_1$，$l_2$，…，$l_k$的并、差运算表示出来。
- 定义3-21 约束Cs是简洁的，假如SATCs（I）是一个强简洁集。

# 约束之间的关系


Succinctness
Anti-
monotonicity
Monotonicity
Convertible
constraints
Inconvertible constraints

# 第三章 关联规则挖掘理论和算法

内容提要

- 基本概念与解决方法
- 经典的频繁项目集生成算法分析
- Apriori算法的性能瓶颈问题
- Apriori的改进算法
- 对项目集格空间理论的发展
- 基于项目序列集操作的关联规则挖掘算法
- 改善关联规则挖掘质量问题
- 约束数据挖掘问题
- 关联规则挖掘中的一些更深入的问题
- 数量关联规则挖掘方法

# 多层次关联规则挖掘

- 根据规则中涉及到的层次，多层次关联规则可以分为：
  - **同层关联规则：**如果一个关联规则对应的项目是同一个粒度层次，那么它是同层关联规则。
  - **层间关联规则：**如果在不同的粒度层次上考虑问题，那么可能得到的是层间关联规则。

- 多层次关联规则挖掘的度量方法可以沿用“支持度-可信度”的框架。不过，多层次关联规则挖掘有两种基本的设置支持度的策略：
  - **统一的最小支持度：**算法实现容易，而且很容易支持层间的关联规则生成。但是弊端也是显然的：

# 多层次关联规则挖掘

- 多层次关联规则挖掘的度量方法可以沿用“支持度-可信度”的框架。不过，多层次关联规则挖掘有两种基本的设置支持度的策略：
  - **统一的最小支持度：**算法实现容易，而且很容易支持层间的关联规则生成。但是弊端也是显然的：
  - **不同层次使用不同的最小支持度：**每个层次都有自己的最小支持度。较低层次的最小支持度相对较小，而较高层次的最小支持度相对较大。这种方法增加了挖掘的灵活性。但是，也留下了许多相关问题需要解决：
    - **首先，不同层次间的支持度应该有所关联，只有正确地刻画这种联系或找到转换方法，才能使生成的关联规则相对客观。**
    - **其次，由于具有不同的支持度，层间的关联规则挖掘也是必须解决的问题。例如，有人提出层间关联规则应该根据较低层次的最小支持度来定。**

# 多维关联规则挖掘

- 多维关联规则可以有：
  - **维内的关联规则：**例如，“年龄（X，20~30）^职业（X，学生）=>购买（X，笔记本电脑）”。这里我们就涉及到三个维：年龄、职业、购买。
  - **混合维关联规则：**这类规则允许同一个维重复出现。例如，“年龄（X，20~30）^购买（X，笔记本电脑） => 购买（X，打印机）”。由于同一个维“购买”在规则中重复出现，因此为挖掘带来难度。但是，这类规则更具有普遍性，具有更好的应用价值，因此近年来得到普遍关注。

# 第三章 关联规则挖掘理论和算法

内容提要

- 基本概念与解决方法
- 经典的频繁项目集生成算法分析
- Apriori算法的性能瓶颈问题
- Apriori的改进算法
- 对项目集格空间理论的发展
- 基于项目序列集操作的关联规则挖掘算法
- 改善关联规则挖掘质量问题
- 约束数据挖掘问题
- 关联规则挖掘中的一些更深入的问题
- 数量关联规则挖掘方法

# 数值属性的描述

- **定义3-25** 设$A=\{A_1, A_2, \ldots, A_m\}$是数据集D中的属性集合，属性$A_i$（$A_i \in A$，$1 \leqslant i \leq m$）的属性值集为$R_i$，$I=\{<A_i, u, v> \in A \times R_i \times R_i\}$为数据集D中的项目组合，$<A_i, u, v>$称为项目（Item）。
  - 若$A_i$为连续属性，则$u \leq v$，且$<u, v>$构成$A_i$的属性值区间；
  - 若$A_i$为分类属性，则$u= v$为分类属性的一个属性值，此时项目$<A_i, u, v>$可以简记为$<A_i, u>$。
- 对数据集D中的连续属性$A_i$，若其属性值空间$[u_i, v_i]$被划分成为n个子区间$[u_1, v_1]$、$[u_2, v_2]$、…、$[u_n, v_n]$，则属性$A_i$将产生n个项目$<A_i, u_1, v_1>$、$<A_i, u_2, v_2]>$、…、$<A_i, u_n, v_n>$。
- 将连续属性的属性值空间划分为若干个子区间并产生项目的过程称为连续属性的离散化。

# 项目集的泛化与特化

- **定义3-26** 设$X \subseteq I$是一个项目集，若$|X|=k$则称项目集为k-项目集。项目集X的属性集合记为attribute（X），即attribute（X）$=\{A_i | A_i \in A, <A_i, u, v> \in X\}$。
  - 对项目集X和X^，若attribute（X）= attribute（X^），且对于任意$A_i \in$attribute（X），都有$\{< A_i, u, v> \in X\} \cap \{<A_i, u^, v^ > \in X^\} \neq \Phi \Rightarrow u^<u<v< v^$，则称项目集X^为X的泛化（Generalization），X为X^的特化（Specialization）。

# 项目集的支持度

- **定义3-27** 对K-项目集$X=\{<A_1, u_1, v_1>, <A_2, u_2, v_2>, \ldots, <A_k, u_k, v_k>\}$和数据集D中的记录T，T的$A_1, A_2, \ldots, A_k$属性的属性值为$T_{A1}, T_{A2}, \ldots, T_{Ak}$，若对所有的$i=1, 2, \ldots, k$，有$u_i \le T_{Ai} \le v_i$，，则称记录T包含在K-项目集X中。
- **定义3-28** 数据集D中包含在项目集X的记录数称为项目集X的支持数，记为$\sigma_x$，项目集X的支持度记作support（X），并满足关系：

$$support(X) = \sigma_x / |D| \times 100\%,$$

其中|D|是数据集D的记录数。若support（X）不小于用户指定的最小支持度，则称X为频繁项目集（或大项目集），否则称X为非频繁项目集（或小项目集）。

# 关联规则的置信度

- 定义3-29 若X，Y为项目集，且X∩Y=Φ，蕴涵式X⇒Y称为数量关联规则，X和Y分别称为X⇒Y的前提和结论。项目集（X∪Y）的支持度称为关联规则X⇒Y的支持度，记作support（X⇒Y），其中X∪Y的含义是在数据库中同时包含X和Y，即support（X⇒Y）= support（X∪Y）。数值关联规则X⇒Y的**置信度**记作confidence（X⇒Y）：
  confidence（X⇒Y）= support（X∪Y）/support（X）×100%。
- 定义3-30 给定用户的最小支持度minsupport和最小置信度minconfidence，如果 support（X⇒Y）≥minsupport且confidence（X⇒Y）≥minconfidence，则称数量关联规则X⇒Y为**强规则**。

# 数量关联数规则的分类

- **根据数值属性的处理方式，主要技术有：**
  - **数值属性的静态离散化：**数值属性使用预定义的概念分层，在挖掘之前进行离散化，数值属性的值用相应的区间来替代。例如，income的概念分层可以用于用区间值“0…20K”，“21…30K”，“31…40K”来替换属性原来的数值。
  - **数值属性的动态离散化：**动态离散化过程需要考虑数据的关联和程度，使用相应的度量手段，如数据点之间的距离，来跟踪数据的语意。
  - **基于特定的技术进行离散化：**常用的方法是根据数据的分布，将数值属性离散化到“箱（Bin）”。由于这些箱可能在挖掘过程中进一步组合，因此有动态的特征。离散化的动态体现在分区合并等过程中。三种常用的分箱策略是等宽分箱、等深分箱和基于同质的分箱。
- **根据使用的规则模板，主要技术有：**
  - **复杂的挖掘模板形式：**类似于“数值属性∩分类属性⇒数值属性∩分类属性”这样的规则。例如，sex=‘female’∩age∈[20，30] =>wages∈[$5，$10]。这类规则较为复杂，是一般性的数量关联规则。
  - **分类规则的挖掘模板：**简单的挖掘模板形式类似于“数值属性∩分类属性⇒分类属性”这样的规则。例如，smoke=Yes∩age∈[60，80] =>heart-desease=Yes。此类规则的左端通常表示的是数据库的的几个连续或分类属性，右端则是一个预定义的类。这样的模板很适合用于分类规则的挖掘。
  - **其他挖掘模板：**例如，挖掘模板形式类似于“分类属性⇒数值属性∩分类属性”这样的规则。这和第二类规则形式恰好相反，但对有些问题，这类规则非常有意义。

# 数量关联规则挖掘的一般步骤

- 以Boolean关联规则算法及其理论为基础，是解决数量关联规则的挖掘问题的最有效途径之一。较典型的数量关联规则挖掘的主要步骤有：
  - 1．**对每个数值属性进行离散化：**选取或设计合适的离散化算法，对数据库中的所有数值属性进行离散化。
  - 2．**离散区间整数化：**对分类属性或数值属性的离散区间，将其值映射成连续的整数标识。这样做的目的是使数据归整以利于挖掘。
  - 3．**在离散化的数据集上生成频繁项目集：**此步和前面介绍的生成频繁项目集的步骤类似。
  - 4．**产生关联规则：**和前面介绍的方法类似。
  - 5．**确定感兴趣的（Interesting）关联规则作为输出**

# 数值属性离散化问题

- **数值属性离散化是关键问题，处理不当会带来如下问题：**
  - 过小置信度问题：若区间的数目过少，则支持区间的元组数目增加，包含区间的强项目集的支持度上升，但在强项目集的子集支持度不变的情况下，将导致右端包含该子集的规则置信度下降，若不能达到置信度阈值，就会造成信息丢失。
  - 过小支持度问题：划分的区间数目过多，则区间的支持度下降，不能有效的生成期望的频繁项目集。
  - 离散化可能会把本来很紧密的数据分割开来，结果就会生成大量没有意义的无用规则。
  - 离散化会带来区间的组合爆炸。假设某数值属性划分成n个基区间，由于相邻的两个连续属性又可以合并生成更大的区间，这样反反复复的合并，最终的区间数目就有0（$n^n$）个。所以具体离散化过程中必须有一种停机条件，当区间合并到一定程度能够自动停下来。离散区间的组合爆炸带来的直接后果是算法效率低下，而且生成成千上万条规则，用户很难从中找到有趣的规则。

# 主要的离散化方法

- 现有的离散化方法主要有两种策略：
  - 归并方法：开始将属性的每个取值都当作是一个离散的值，然后逐个反复合并相邻的属性值，直到满足某种条件结束合并。
  - 划分方法：将属性的整个取值区间作为一个离散属性，然后对该区间反复划分成更小的区间，直到满足某种条件结束划分。
- 较好的离散化方法通常是这两种策略的结合，是动态划分归并的过程。
- 目前最典型的数值属性离散化方法主要有：
  - 等宽度划分
  - 等深度划分
  - 基于距离的划分

# 等宽度划分的方法

**算法3-17** 等宽度划分算法描述

输入：数值属性A，区间数n。

输出：离散后的区间。

（1）扫描数据库，得到A的最大值max（A）和最小值min（A）；

（2）求区间的宽度w，w=（max（A）-min（A））/n;

（3）形成离散后的区间$[l_1, v_1]$，$[l_2, v_2]$，...$[[l_n, v_n]$输出，其中$l_i$= min（A）+（i-1）*w，$v_i$= $l_i$+w。

- 等宽度划分的方法是最简单的离散化方法，一般适用分布比较均匀的数据。算法只要一次扫描数据库，因此算法效率较高。等宽度划分的方法单纯从数学角度对数值属性进行划分，不考虑数据分布的特点。由于该方法比较直观，比较适合数值属性的前期处理，因此通常和聚类等方法结合，才能取得好的离散化效果。

# 等深度划分方法

- **定义3-31** 若A的取值区间在$[X_l, Y_m]$。可以将A的取值区间分割为一系列不相交的域$B_i=[X_i, Y_i]$，其中i=1，2，…，m且$X_i \leqslant Y_i \leqslant X_{i+1}$，称$B_i$为A的一个桶（Bucket），并称$B_i$中元组个数为桶的大小，记为$U_i$。若两个桶的大小相等，则称它们为等深桶。

**算法3-18** 等深度划分描述

输入：数值属性A，区间数（桶数）n。

输出：离散后的区间。

（1）数值属性A从小到大或从大到小排序；

（2）扫描数据库，统计数据库的记录数N；

（3）求桶的深度：h=N/n；

（4）逐个扫描排序后的A值，形成离散后的区间$[l_i, v_i]$（i=1，2，…n），每个区间包含h个值。

- 等深度划分一般适用于属性之间关联度比较低的数据集。等深度划分的方法趋向于把具有共性的、支持度很高的相邻值划分到不同的区间去。当数据分布在某个点附近达到峰值时，等深度划分这种机械的方法并不能反映出数据本身的特点，因此对高偏度的数据效果不理想。

# 基于距离的划分的方法

- 等宽度划分的方法和等深度划分的方法都没有充分考虑数据的分布，单纯从几何和数学的角度对数值属性进行划分。基于距离的离散化方法通过数据聚类方法形成数据区间。

**算法3-19**基于距离的划分

输入：数值属性A，区间的数目k。

输出：离散后的区间[$u_1$，$v_1$]，[$u_2$，$v_2$]，…，[$u_k$，$v_k$]。

（1）从数值属性A任意选择k个不同的值作为初始的簇的中心；

（2）REPEAT

（3）　根据簇中A的平均值，将每个A值（重新）赋给最类似的簇

（4）　更新簇的平均值，即计算每个簇中A值的平均值

（5）INTIL 不再发生变化.

# 第三章 关联规则挖掘理论和算法

内容提要

- 基本概念与解决方法
- 经典的频繁项目集生成算法分析
- Apriori算法的性能瓶颈问题
- Apriori的改进算法
- 对项目集格空间理论的发展
- 基于项目序列集操作的关联规则挖掘算法
- 改善关联规则挖掘质量问题
- 约束数据挖掘问题
- 关联规则挖掘中的一些更深入的问题
- 数量关联规则挖掘方法