



# 数据分析算法

北京理工大学计算机学院 孙新

2019年1月

## 4、经典的机器学习方法

---

- 4.1 分类算法原理
- 4.2 决策树算法
- 4.3 K-近邻分类算法 (KNN算法)
- 4.4 K-均值聚类算法 (K-means算法)
- 4.5 Apriori关联规则算法

- ❑ **关联规则**是反映一个事物与其他事物之间的相互依存性和关联性。
  - 如果两个或者多个事物之间存在一定的关联关系，那么其中一个事物就能够通过其他事物预测到。
- ❑ 超市购物篮分析
  - 典型的关联规则发现问题是对超市中的货篮数据进行分析。通过发现顾客放入货篮中的不同商品之间的关系来分析顾客的购买习惯。

算法输入

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

算法输出

发现的关联规则:

$\{\text{Milk}\} \rightarrow \{\text{Coke}\}$

$\{\text{Diaper, Milk}\} \rightarrow \{\text{Beer}\}$

# 什么是关联规则挖掘？

## 4.5 Apriori关联规则算法

- ❑ **关联规则挖掘：**在事物、关系数据库中的项集和对象中发现频繁模式、关联、相关性或者因果结构
  - **频繁项集：**数据库中频繁出现的项集
- ❑ **目的：**发现数据中的规律
  - 超市数据中的什么产品会一起购买？——啤酒和尿布
  - 在购买铁锤之后下一步会购买什么？——铁锤和铁钉

据报道，美国中西部的一家连锁店发现，男人们会在周四购买尿布和啤酒。



### □ 银行业务分析：若想扩大信用卡业务

#### 银行信用卡产品覆盖率明细表

项目名称	用户数	信用卡数量	信用卡覆盖率	目标覆盖率
个人消费贷款	133,085	25,693	19%	70%
个人住房贷款	3,500	1,198	34%	70%
VIP理财业务	67,055	18,070	27%	70%
基金业务	25,875	15,636	60%	70%
外汇交易业务	15,660	9,737	62%	70%
个人活期存款	160,406	43,602	27%	30%
个人定期存款	454,163	19,634	4%	30%
代发工资业务	301,670	46,083	15%	50%
合计(平均覆盖率)	1,161,414	179,653	15%	43%

### □ 银行业务分析：若想扩大信用卡业务

假设：个人住房贷款业务=1、信用卡业务=2、VIP理财业务=3、基金业务=4、外汇交易业务=5

交易事务数据表

银行信用卡产品覆盖率明细表

项目名称	用户数	信用卡数量	信用卡覆盖率	目标覆盖率
个人消费贷款	133,085	25,693	19%	70%
个人住房贷款	3,500	1,198	34%	70%
VIP理财业务	67,055	18,070	27%	70%
基金业务	25,875	15,636	60%	70%
外汇交易业务	15,660	9,737	62%	70%
个人活期存款	160,406	43,602	27%	30%
个人定期存款	454,163	19,634	4%	30%
代发工资业务	301,670	46,083	15%	50%
合计(平均覆盖率)	1,161,414	179,653	15%	43%

算法输入

TID	项目列表
T1	1, 2
T2	1, 3, 5
T3	2, 4
T4	2, 3, 5
T5	1, 4
T6	2, 4, 5

算法输出

$2 \Rightarrow 4, confidence = 50\%$   
 $4 \Rightarrow 2, confidence = 67\%$   
 $2 \Rightarrow 5, confidence = 50\%$   
 $5 \Rightarrow 2, confidence = 67\%$   
 $3 \Rightarrow 5, confidence = 100\%$   
 $5 \Rightarrow 3, confidence = 67\%$

### 项目(item)与项集(item-set)

设 $I=\{i_1, i_2, \dots, i_m\}$ 是 $m$ 个不同项目的集合，每个 $i_k (k=1, 2, \dots, m)$ 称为一个项目(*Item*)。

项目的集合  $I$  称为项目集合(*Itemset*)，简称为项集。其元素个数称为项集的长度，长度为 $k$ 的项集称为 $k$ -项集( $k$ -*Itemset*)。

### 交易

每笔交易 $T$ (*Transaction*)是项集 $I$ 上的一个子集，即 $T \subseteq I$ ，但通常 $T \subset I$ 。每一个交易有一个唯一的标识——交易号，记作 $TID$

交易的全体构成了交易数据库 $D$ ，交易集 $D$ 中包含交易的个数记为 $|D|$ 。

### 项集支持度:

设项集  $X \subset I$ , 项集X的支持度(support)定义如下:

$$\text{support}(X) = \frac{\text{count}(X \subseteq T)}{|D|}$$

项集的支持度反映了项集在交易集中的重要性

### 最小支持度:

能够发现关联规则的项集必须满足最小支持阈值, 称为项集的最小支持度 (*Minimum Support*), 由用户或专家自定义

### 频繁项集:

支持度大于或等于**最小支持度阈值**的项集称为**频繁项集**, 简称频繁集, 反之则称为**非频繁集**。

通常k-项集如果满足**最小支持度阈值**, 称为k-频繁集, 记作 $L_k$ 。



**关联规则(Association Rule)**是形如  $X \Rightarrow Y$  的逻辑蕴含式, 其中  $X \subseteq I, Y \subseteq I$ , 且  $X \cap Y = \Phi$ 。

- ❖ **支持度**是数据库中包含  $X \cup Y$  的事务占全部事务的百分比  

$$support(X \Rightarrow Y) = P(X \cup Y) = \frac{\text{count}(X \cup Y)}{|D|}$$
- ❖ **置信度**是包含  $X \cup Y$  的事务占包含  $X$  的事务数的比值  

$$confidence(X \Rightarrow Y) = P(Y|X) = \frac{\text{support}(X \cup Y)}{\text{support}(X)}$$

Transaction-id	Items bought
10	A, B, C
20	A, C
30	A, D
40	B, E, F

最小支持度 50%  
最小置信度 50%

Frequent pattern	Support
{A}	75%
{B}	50%
{C}	50%
{A, C}	50%

**强关联规则：**规则  $X \Rightarrow Y$  必须满足：

$support(X \Rightarrow Y) \geq \text{最小支持度}$   
 $confidence(X \Rightarrow Y) \geq \text{最小置信度}$

规则  $A \Rightarrow C$ :

支持度 =  $support(\{A\} \cup \{C\}) = 50\%$

置信度 =  $support(\{A\} \cup \{C\}) / support(\{A\}) = 66.6\%$

## 4.5 Apriori关联规则算法

---

- ▣ 关联规则挖掘的基本思想
- ▣ Apriori 算法的基本过程
- ▣ Apriori关联规则算法步骤
- ▣ Apriori关联规则算法的特点
- ▣ R语言实现Apriori算法示例

- ❖ **找出所有的频繁项集：找出支持度大于最小支持度的项集，即频繁项集。**
- ❖ **由频繁项集产生关联规则：根据定义，这些规则必须满足最小支持度和最小可信度。**

Apriori的基本思想：

频繁项集的任何非空子集也一定是频繁的

- Apriori算法命名源于算法使用了频繁项集性质的先验 (Prior) 知识
  - Apriori算法基本过程：
    - (1) 生成候选集：找出候选集，即有可能成为频繁集的项集
    - (2) 频繁集：生成通过数据库扫描筛选出满足条件的候选集形成又一层频繁集

**频繁集的子集也一定是频繁的**  
例如, 如果 $\{AB\}$  是频繁集, 则  $\{A\}$   $\{B\}$  也一定是频繁集

**从1到k (k-频繁集) 递归查找频繁集**

  - (3) 生成关联规则：用得到的频繁集生成关联规则
- 
- 挖掘或识别出所有频繁项集是该算法的核心**

**连接:** 用  $L_{k-1}$  自连接得到候选  $k$ -项集  $C_k$ , 连接时只能将只差最后一个项目不同的项集进行连接

**修剪:** 一个候选  $k$ -项集  $C_k$ , 如果它的一个  $k-1$  项集 (它的子集) 不是频繁的, 那它本身也不可能是频繁的。

假定  $L_{k-1}$  中的项按顺序排列

第一步: 自连接  $L_{k-1}$

如何生成候选集

insert into  $C_k$

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

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

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

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  $C_k$

### 生成候选集的例子

$L_3 = \{abc, abd, acd, ace, bcd\}$

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

$C_4 = \{abcd\}$

伪代码:

$C_k$ : k候选集

$L_k$ : k频繁集

$L_1 = \{ \text{频繁项} \};$

**for** ( $k = 2; L_{k-1} \neq \emptyset; k++$ ) **do begin**

$C_k$  = 由 $L_{k-1}$ 产生的候选集;

**for** 遍历交易数据库D计算每个 $C_k$ 候选集在交易数据库中出现的  
支持度

$L_k$  = 从所有候选集 $C_k$  中选择具有最小支持度的项集构成 $L_k$  频繁  
项集

**end**

**return**  $\cup_k L_k$ ;



# Apriori关联规则算法步骤

## 4.5 Apriori关联规则算法

数据库 D

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

扫描 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

minsup=1

$C_2$

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

扫描 D

$C_2$

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

$L_2$

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

$C_3$

itemset
{2 3 5}

扫描 D

$L_3$

itemset	sup
{2 3 5}	2

由频繁项集产生关联规则的产生步骤如下：

对于每个频繁项集 $L$ ，产生 $L$ 的所有非空子集

对于 $L$ 的每个非空子集 $s$ 和它的补集 $(L-s)$ ，如果条件概率大于最小置信度阈值，则输出规则 “ $s \Rightarrow (L-s)$ ”

Apriori算法的核心:

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集的支持度

*Apriori* 的瓶颈:

巨大的候选集:

$10^4$  个频繁1-项集要生成  $10^7$  个候选 2-项集

要找长度为100的频繁模式, 如  $\{a_1, a_2, \dots, a_{100}\}$ , 你必须先产生  $2^{100} \approx 10^{30}$  个候选集

多次扫描数据库:

如果最长的频繁模式是 $n$ 的话, 则需要  $(n+1)$  次数据库扫描

提高Apriori效率的方法:

不用生成候选集的关联规则挖掘算法FP-growth