文章编号: 1001-4543(2022)02-0172-05 DOI: 10.19570/j.cnki.jsspu.2022.02.012

基于改进 Apriori 算法的保险产品推荐

朱天宇, 谭文安

(上海第二工业大学 计算机与信息工程学院, 上海 201209)

摘 要: 为了加强保险业务员业务能力, 提高被推荐保险产品的成交率, 基于经典关联规则挖掘算法 Apriori, 提出了一种改进算法 Apriori based on difference set(DS_Apriori) 来计算符合条件的保险产品。贡献包括: ① 通过以键值方式重新组织数据、根据时序划分数据集、降低迭代搜索的次数等方式来提高算法执行效率; ② 通过以客户为主键聚合事务数据来挖掘潜在关联规则, 实现精准推送。实验表明所提出的 DS_Apriori 算法执行效率优于 Apriori 等算法, 可以给保险业务员提供业务指导, 精准推荐最符合推荐条件的保险产品。

关键词: 关联规则; 差集; Apriori 算法; 保险推荐

中图分类号: TP311.1; F842.4 **文献标志码**: A

Insurance Product Recommendation Based on Improved Apriori

ZHU Tianyu, TAN Wen'an

(School of Computer and Information Engineering, Shanghai Polytechnic University, Shanghai 201209, China)

Abstract: In order to enhance the business capability of insurance salesmen and improve the turnover rate of recommended insurance products, an improved algorithm Apriori based on difference set (DS_Apriori) is proposed to calculate the eligible insurance products based on the classical association rule mining algorithm Apriori. The contributions include ① improving the algorithm execution efficiency by reorganizing the data in key-value way, dividing the data set according to the time sequence, and reducing the number of algorithm iterations; ② mining the potential association rules by aggregating the transaction data with the customer as the main key to achieve accurate pushing. The experiments show that the proposed DS_Apriori algorithm performs better than algorithms such as Apriori. It can provide business guidance to insurance salesmen and accurately recommend the insurance products that best meet the recommended conditions.

Keywords: association rule; difference set; Apriori algorithm; insurance recommendation

0 引言

清华大学国家金融研究院中国保险与养老金研究中心发布的《2021 中国互联网保险消费者洞察报告》显示, 传统线下网点、保险代理人仍然是主要代理渠道, 调查人群中 88% 的人通过线下渠道购买保险。然而, 目前保险公司线下推荐保险产品时, 主要依靠保险经理人的经验, 对于新员工而言, 经验的缺乏和普遍的能力不足导致了推荐的产品不满足客户需求的问题, 因而销售成功率低、客户满意度低。近年来, 数据规模呈几何级数高速成长。海量数据已经成为企业资产的一部分 [1], 数据分析技术也日渐成熟 [2]。在这种情况下, 商务智能技术的应用给问题的解决提供了可行方案。数据分析表明, 多家公

收稿日期: 2022-01-05

通信作者: 谭文安 (1965–), 男, 湖北公安人, 教授, 博士, 主要研究方向为协同计算与系统进化、协同学习与知识管理等。
E-mail: watan@sspu.edu.cn

基金项目: 国家自然科学基金项目 (61672022, U1904186), 上海第二工业大学电子信息类专业硕士协同创新平台建设项目资助

司采用商务智能优化业务方法之后, 在规避风险和保留客户方面取得了更好的效果 [3]。

Apriori 算法是由 Agrawal 和 Skrikant 于 1994 年提出的关联规则算法, 用于挖掘频繁项集间的关联规则。但是在大规模数据库中应用 Apriori 算法的效率非常低, 因为 Apriori 算法使用逐层搜索的迭代方法寻找频繁项集。自 Apriori 算法出现至今, 不少学者从不同角度对该算法提出了优化和改进的方法。周发超等 [4] 提出了一种 Apriori 的改进方案 I_Apriori, 通过减少扫描数据库次数, 降低候选项集计算复杂度以及减少预剪枝步骤计算量等途径提高了算法的执行效率。张继荣等 [5] 基于新的数据结构, 利用 Hash 表的存储技术以及对 Apriori 算法的优化提高了查找频繁项集的效率。胡世昌等 [6] 提出以二进制编码的项集作为载体载入内存, 并在二进制编码的基础上有效地进行等效的集合之间的运算, 有效提高了 Apriori 算法的执行效率和空间利用率。陈晨等 [7] 提出的 Gra-Apriori 算法, 解决了经典的 Apriori 无法考虑属性类别关系的问题。郭凯等 [8] 改进后的 Apriori 算法大大减少了无效规则的产生, 通过关联分析可得到对断面进行调整的有效强关联规则。高海洋等 [9] 通过数据压缩的方法减少了数据库扫描次数的同时, 对生成的候选集进行了多次验证, 大大减少了无效候选集的数量。赵学健等 [10] 对 Apriori 算法复杂的自连接和剪枝过程进行了优化, 简化了频繁项目集的生成过程, 提高了 Apriori 算法的时间效率。Supriyono 等 [11] 将 Apriori 算法应用到农业领域, 提高了农产品销售额。

以上介绍的相关算法实施比较复杂, 或者虽有改进但不能较好适用于应用场景问题, 本文通过采取 Python 字典列表 [12] 以键值方式重新组织数据、根据时序划分数据集、仅仅计算频繁 2-项集的差集 [13] 等方式来提高算法执行效率, 通过以客户为主键聚合事务数据来挖掘潜在关联规则, 从而达到算法优化的目的, 改进了 Apriori 算法, 提出了一种改进的 DS_Apriori 算法, 计算出关联规则, 以得出已经停售的保险产品的最佳替代产品, 来指导保险经理人线下展业, 在最大程度上保留客户。

本文将详细阐释改进的 DS_Apriori 算法和具体实现, 并对 DS_Apriori 算法与 Apriori 等算法进行实验和对比分析, 最后得出总结及工作展望。

1　改进的 Apriori 算法

1.1　算法改进方案

1.1.1　问题定义

设 $J=\{I_1,I_2,\cdots,I_m\}$ 是项的集合, 设数据 D 是数据库事务的集合, 其中每个事务 T 是一个非空项集, 使得 $T\subseteq J$。每个事务都有一个标识符, 称为 TID。设 A 是一个项集, 事务 T 包含 A, 当且仅当 $A\subseteq T$。关联规则是形如 $A\Rightarrow B$ 的蕴涵式, 其中 $A\subset J$, $B\subset J$, $A\neq\varnothing$, $B\neq\varnothing$, 并且 $A\cap B=\varnothing$。规则 $A\Rightarrow B$ 在事务集 D 中成立。具有支持度 s, 其中 s 是 D 中事务包含 $A\cup B$ 的百分比。它是概率 $P(A\cup B)$。规则 $A\Rightarrow B$ 在事务集 D 中具有置信度 c, 其中 c 是 D 中包含 A 的事务同时也包含 B 的事务的百分比。这是条件概率 $P(B|A)$, 即:

$$\text{support}(A\Rightarrow B)=P(A\cup B) \tag{1}$$

$$\text{confidence}(A\Rightarrow B)=P(B|A) \tag{2}$$

同时满足最小支持度阈值 (min_sup) 和最小置信度阈值 (min_conf) 的规则为强规则, 项的集合称为项集, 包含 k 个项的项集称为 k 项集, 如集合 $X=\{I_1,I_2\}$ 是一个 2 项集。

1.1.2　数据准备

保险产品通常具有时效性, 保险产品的推陈出新已是司空见惯 [14]。传统的 Apriori 算法没有对数据进行针对性的划分, 由于数据中包含已经停售的产品, 在产生关联规则时, 会产生停售产品之间的关联规则, 既浪费了宝贵的计算性能, 又没有意义。这里按照保险产品 A 的停售时间, 将完整数据集 CompleteSale 集中停售之前的数据划分出来, 称为 BeforeStopSale 集。通过重新划分数据集, 为差集的计算提供数据准备, 以此来挖掘新旧产品之间的关联规则。

传统的算法是以单次保险销售记录为一个事务, 一条销售记录往往只包含一份保单的内容; 如果直接在此基础上采用 Apriori 算法, 单次销售记录的不连续忽略了用户所购买产品在时序上的相关性; 不能够很好地将同一用户的多次购买记录关联起来, 导致了潜在关联规则的丢失, 关联规则挖掘效果受到影响。因此, 本文重新组织了数据, 将同一用户的所有购买记录整合为一条事务记录加入运算, 建立起用户所购买产品之间的时序联系; 使用 Python

的字典列表数据结构, 通过键值对保存用户购买的产品以及该产品的购买次数, 以此保留了原始数据作为多重集合的数据特征如表 1 所示; 同时将购买次数也加入支持度计数的计算, 用户重复购买这一行为在数据中得到了体现 (见表 2), 更能体现用户的购买偏重。采用内存数据库 Redis 保存键值对数据, 以此优化数据加载速度, 进一步提高算法性能。

表 1 Apriori 算法输入数据格式
Tab. 1 The input data form for Apriori algorithm

TID	商品 ID 列表
T100	I_1, I_2, I_5
T200	I_2, I_4
...	...

表 2 DS_Apriori 输入数据格式
Tab. 2 The input data form for DS_Apriori algorithm

TID	商品 ID 列表
T100	$\{I_1: 3\}, \{I_2: 3\}, \{I_5: 2\}$
T200	$\{I_2: 4\}, \{I_4: 3\}$
...	...

1.1.3 算法改进

传统的 Apriori 算法因为效率低下很少用在大规模数据挖掘中。本文放弃其迭代搜索全部的频繁项集的思路, 转而搜索目标项目的频繁 2-项集, 只挖掘目标项目的二项关联规则, 减少连接操作, 以此解决频繁的连接操作造成的性能问题。

$$X = \{I_i, I_j | I_i \in J, I_j \in J \text{ 且 } I_i \neq I_j\} \tag{3}$$

$$\text{ResultSet} = \{X_1, X_2, \cdots, X_n\} \tag{4}$$

式中, X 为频繁 2-项集; ResultSet 为频繁 2-项集的集合。由先验性质的数学原理可得, 产生频繁 2-项集并不需要剪枝, 再次提高了算法的效率。在 BeforeStopSale 数据集和 CompleteSale 数据集产生频繁 2-项集得到结果 $\text{ResultSet}_{\text{BeforeStopSale}}$ 和 $\text{ResultSet}_{\text{CompleteSale}}$ 之后, 分别计算出各自的关联规则集合 $\text{RuleSet}_{\text{CS}}$ 和 $\text{RuleSet}_{\text{BSS}}$

$$\text{RuleSet}_{\text{target}} = \text{RuleSet}_{\text{CS}} - \text{RuleSet}_{\text{BSS}} \tag{5}$$

最终得到的关联规则集合 $\text{RuleSet}_{\text{target}}$, 即可推出保险产品 A 的替代产品。

1.2 DS_Apriori 算法伪代码

改进的 Apriori 算法伪代码描述如下:

Algorithm 1 DS_Apriori

Input: D, min_sup
Output: L, frequent 2-itemset in D.

```
1.  L1 = find_frequent_1_itemsets (D);
2.  C2 = apriori_gen (L1);
3.  for each transaction t ∈ D {
4.  Ct = subset (C2, t);
5.  c.count ++;
6. }
7. for each candidate c ∈ Ct;
8.  L2 = {c ∈ C2| c.count>=min_sup}
9. Return L2;
10. Procedure apriori_gen (L1: frequent 1 itemset)
11.  for each itemset l1 ∈ l1
12.  for each itemset l2 ∈ l1
13.  if (l1[1] = l2[1]) ∧ ··· ∧ (l1[0] = l2[0])∧
     (l1[1] < l2[0]) then {
14.  c = l1 ⋈ l2;
15.  add c to C2;
16.  }
17. return C2;
```

改进的 DS_Apriori 算法创新工作主要包括如何构造频繁 2-项集和关联规则两部分。

1.2.1 频繁 2-项集构造

构造频繁 2-项集目的是找出所有可能存在的二元关联项集, 其构造步骤如下:

(1) 从生产数据库中获取原始数据, 利用 SQL 语句重新组织数据, 以字典列表的数据结构存储以用户 ID 为 TID 的事务数据集合 BeforeStopSale 集和 CompleteSale 集; BeforeStopSale 是保险产品 A 停售之前的所有保险销售数据; CompleteSale 是所有的保险销售数据。

(2) 输入预定的最小支持度计数 min_sup, 扫描事务数据集合, 计算每个 1-项集的支持度计数 (即每个产品的购买次数); 删除小于最小支持度计数的集合, 得到频繁 1-项集的集合 L_1。

(3) 连接步: 使用 L_1 自连接 [15] 产生候选 2-项集的集合 C_2。

(4) 剪枝步: 根据先验性质, 删除子集不是频繁 1-项集的候选 2-项集, 得到剪枝后的 C_2, 需要注意的是, 这一步并没有候选项从 C_2 中删除, 因为这些候选的每个子集也都是频繁的。

(5) 计算每个候选 2-项集的支持度计数, 删除小于最小支持度计数的候选 2-项集, 得到频繁 2-项集的集合 L_2。

(6) 对 BeforeStopSale 和 CompleteSale 分别执行步骤 (2)∼(5), 得到频繁 2-项集 $\text{ResultSet}_{\text{BeforeStopSale}}$ 和 $\text{ResultSet}_{\text{CompleteSale}}$。

1.2.2　关联规则构造

关联规则的作用是确定关联项之间的关联关系, 其构造步骤如下:

(1) 设置置信度阈值 min_conf;

(2) 构造 L_2 中频繁 2-项集的项之间的关联规则 A⇒X, 此处仅构造目标产品 A 与其蕴含的项集之间的关联规则, 并计算置信度;

(3) 删除小于置信度阈值的关联规则, 最终得到满足条件的强关联规则;

(4) 分别对频繁 2-项集 $\text{ResultSet}_{\text{BeforeStopSale}}$ 和 $\text{ResultSet}_{\text{CompleteSale}}$ 执行 (1)∼(3) 步骤, 得到强关联规则集合 $\text{RuleSet}_{\text{BSS}}$ 和 $\text{RuleSet}_{\text{CS}}$;

(5) 依据 $\text{RuleSet}_{\text{target}} = \text{RuleSet}_{\text{CS}} - \text{RuleSet}_{\text{BSS}}$, 求得 $\text{RuleSet}_{\text{target}}$ 集合中关联规则右端即为保险产品 A 的最可能替代产品。

2　实验结果与分析

为了验证所提出算法的有效性和高效性, 将所提出的 DS_Apriori 算法与经典的 Apriori 和 Han 等 [16] 提出的不产生中间项集的 FP-growth 进行了对比试验。实验环境参数如表 3 所示, 算法使用 Python 语言实现。测试数据集 mushroom.dat 来自 UCI 公共数据集 [17]。

表 3　实验环境参数
Tab. 3　Experiment environment parameters

实验环境	实验环境参数
CPU	AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz
RAM	Micron 4ATF11G64HZ-3G2E1 16 GB
GPU	NVIDIA GeForce RTX 3050 Laptop GPU
持久化数据库	MySQL 8.0.27.1
内存数据库	Redis 4.0.11

Mushroom 数据集包括对姬松茸和 Lepiota 家族中 23 种鳃蘑菇的假设样本的描述, 包含 8 124 条数据, 存在缺失值, 用 1 代替; 共 23 个属性。

为了消除单次实验带来误差的影响, 本文采用了多次取平均的方式, 统计在不同最小支持度的情况下算法的执行时间。对 Mushroom 数据集应用 Apriori 等算法和本文算法的执行时间如表 4 所示。

表 4　Mushroom 数据集的实验结果对比
Tab. 4　Experiment results of Mushroom data set

算法	最小支持度不同时算法的执行时间/s				
	0.20	0.22	0.24	0.26	0.28
Apriori	174.733 6	22.918 0	11.913 4	6.835 9	5.899 6
FP-growth	21.563 8	12.523 1	6.813 4	3.278 5	2.914 8
DS_Apriori	0.712 3	0.740 0	0.724 4	0.771 2	0.730 0

为了更加直观地展现 3 种算法执行时间的对比情况, 表 4 中执行时间的可视化如图 1 所示。

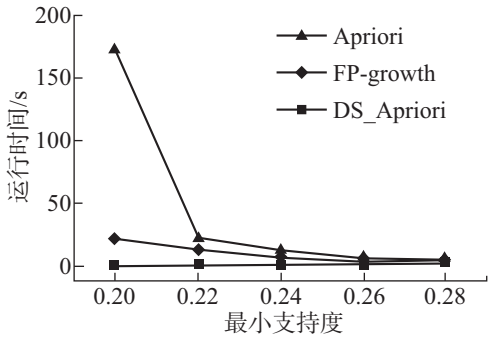

图 1　Mushroom 数据集的实验结果对比
Fig. 1　Experiment results of Mushroom data set

图 1 所示为各算法计算得出 635 个 2 项关联关系的运行时间对比。从表 4 可以看出, DS_Apriori 算法在效率上明显优于 Apriori 算法, 略优于 FP-growth 算法, 在最小支持度较小时尤为明显。随着最小支持度增加, Apriori 算法与 FP-growth 算法的执行时间呈下降趋势, 但是 DS_Apriori 算法的执行时间却基本持平。从图 1 可以直观地看出, DS_Apriori 算法随最小支持度的增大, 其执行时间曲线几乎趋于水平, 而 Apriori 算法的下降幅度却很大, 尤其在最小支持度为 0.20 和 0.22 时表现得尤为明显。但是当最小支持度为 0.28 时, 两种算法的执行时间基本一致。这主要是因为最小支持度越大, 过滤掉的频繁 k-项集 ($k > 2$) 越多, 运算量也越相近。

在以上实验中, 以 Mushroom 数据集为对象, 对 Apriori、DS_Apriori 算法与 FP-growth 算法进行了比较, 改进算法在效率上优于后两者。同时, 从上述实验结果可以看出, 在不同的最小支持度取值下,

Apriori 算法在执行时间上的波动幅度明显大于改进算法。FP-growth 算法的执行时间随着最小支持度的增加较为平稳的下降, 不断逼近 DS_Apriori 算法。综上所述, 本文改进算法 DS_Apriori 的执行效率大幅高于 Apriori 算法与 FP-growth 算法, 是一种行之有效的频繁 2-项集生成算法。

DS_Apriori 算法与 Apriori、FP-growth 算法得出关联规则的原理相同。Apriori、FP-growth 算法得到的是二元及以上的关联规则, DS_Apriori 算法得到的是二元关联规则, 在精准度上一致。在业务要求下, Apriori、FP-growth 算法产生了冗余的关联规则, DS_Apriori 算法效率更高、更符合要求。

3 结 语

本文提出了一种改进的 DS_Apriori 算法, 创新工作表现在: ① 通过数据重组以挖掘出更多潜在的关联规则; ② 通过计算频繁 2-项集的差集代替迭代搜索, 减少搜索次数, 降低了算法的时间复杂度, 为大规模数据实时处理提供了可行方案。依据此关联规则得到某款已停售保险产品的最佳替代产品, 为保险业务员线下展业提供指导。由于算法对数据要求较为严格, 数据预处理花费较多时间。今后工作可考虑实现 DS_Apriori 算法与大数据存储技术以及并行计算相结合, 整体上提高算法的效率。

参考文献:

[1] 韩海庭, 原琳琳, 李祥锐, 等. 数字经济中的数据资产化问题研究 [J]. 征信, 2019, 37(4): 72-78.

[2] 梅宏. 大数据发展现状与未来趋势 [J]. 交通运输研究, 2019, 5(5): 1-11.

[3] 杨婷婷, 汪哲健. 探究保险客户属性对保险销售的影响 [J]. 云南行政学院学报, 2013(z2): 217-218.

[4] 周发超, 王志坚, 叶枫, 等. 关联规则挖掘算法 Apriori 的研究改进 [J]. 计算机科学与探索, 2015(9): 1075-1083.

[5] 张继荣, 王向阳. 基于 X ML 数据挖掘的 Apriori 算法的研究与改进 [J]. 计算机测量与控制, 2016, 24(6): 178-180.

[6] 胡世昌, 李劲华, 王常颖. 基于二进制编码的 Apriori 改进算法 [J]. 计算机应用研究, 2020, 37(2): 398-400.

[7] 陈晨, 侯瑞瑞, 张新梅, 等. 基于危化品事故案例的决策规则提取算法研究 [J]. 中国安全生产科学技术, 2021, 17(4): 85-91.

[8] 郭凯, 韩昳, 王岩, 等. 基于改进 Apriori 算法的直流闭锁受端断面越限调整策略 [J]. 电力建设, 2020, 41(9): 94-101.

[9] 高海洋, 沈强, 张轩溢, 等. 一种基于数据压缩的 Apriori 算法 [J]. 计算机工程与应用, 2013(14): 117-120.

[10] 赵学健, 孙知信, 袁源, 等. 一种正交链表存储的改进 Apriori 算法 [J]. 小型微型计算机系统, 2016, 37(10): 2291-2295.

[11] SUPRIYONO, FERINE K F, PUSPITASARI D, et al. Implementation of data mining with Apriori techniques to determine the pattern of purchasing of agricultural equipment (case study: XYZ store) [J]. Journal of Physics: Conference Series, 2021, 1933: 012029.

[12] 刘雪琳, 章钰琪, 董爱国. 基于 Python 的物理实验数据处理系统设计与实现 [J]. 实验技术与管理, 2021, 38(3): 74-78.

[13] 黄坤, 吴玉佳, 李晶. 基于差集的高效用项集挖掘方法 [J]. 电子学报, 2018, 46(8): 1804-1814.

[14] 李佩, 靳永辉. 我国保险行业发展态势、困境与路径探寻: 基于 “一带一路” 视野的战略分析 [J]. 技术经济与管理研究, 2017(6): 66-70.

[15] 李莹, 代勤. 关系代数运算与 SQL 查询的对应关系 [J]. 内蒙古农业大学学报 (自然科学版), 2003, 24(3): 71-74.

[16] HAN J W, PEI J, YIN Y W. Mining frequent patterns without candidate generation [J]. Sigmod Rec, 2000, 29(2): 1-12.

[17] DUA D, GRAFF C. UCI machine learning repository[DB/OL]. Irvine: University of California, [2021-12-16]. http://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/.