A Personalized Commodities Recommendation Procedure and Algorithm Based on Association Rule Mining

Jianyi Zhang
Yunfeng Wang
Jie Li

Follow this and additional works at: https://aisel.aisnet.org/iceb2004

This material is brought to you by the International Conference on Electronic Business (ICEB) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ICEB 2004 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
A Personalized Commodities Recommendation Procedure and Algorithm Based on Association Rule Mining

Jianyi Zhang, Yunfeng Wang, Jie Li
School of Management, Hebei University of Technology, Tianjin 300130, China
jyzhang214@eyou.com, {ywang,lijie}@hebut.edu.cn

ABSTRACT

The double-quick growth of EB has caused commodities overload, where our customers are no longer able to efficiently choose the products adapted to them. In order to overcome the situation that both companies and customers are facing, we present a personalized recommendation, although several recommendation systems which may have some disadvantages have been developed. In this paper, we focus on the association rule mining by EFFICIENT algorithm which can simple discovery rapidly the all association rules without any information loss. The EFFICIENT algorithm which comes of the conventional Apriori algorithm integrates the notions of fast algorithm and predigested algorithm to find the interesting association rules in a given transaction data sets. We believe that the procedure should be accepted, and experiment with real-life databases show that the proposed algorithm is efficient one.

Keywords: Association Rules, Data Mining, Procedure, Algorithm, Personalized Recommendation

1. INTRODUCTION

Electronic Business has been growing rapidly, keeping pace with the Web [1]. According to the investigations of Angus Reid Group in March 2000, there have been more than three hundred million users using the Internet in the world, and there will be three billion ones by 2005[2]. Its rapid growth has made both companies and customers face a new situation in which companies find it harder to survive due to more and more competition, the opportunity for customers to choose among more and more commodities has increased the burden of information processing before they select which ones meet their needs. In order to solve the problem, researches and practitioners have stressed the need for marketing strategies, for example one-to-one marketing, CRM and mass customization engineering. So under this condition a personalized commodities recommendation which helps our customers find the ones they prefer are very important and useful.

The great amount of data not only gives the statistics, but also offers the resources of experiences and knowledge. Data mining, as one of the promising technologies since 1990s, is to some extent a non-traditional data-driven method to discover novel, useful, hidden knowledge from massive data sets [3]. The discovery of interesting association among business decision making processes, such as catalog design, cross marketing, and loss-leader analysis [4]. Data mining is one of the most popular techniques that can find potential business knowledge from enterprise databases in support of making better decision, it can discovery potentially significant patterns and rules underlying the database. It can be categorized into several interesting areas, such as association rules, clustering, decision tree analysis and so on. Association rule mining finds interesting association or correlation relationships among a large set of data items. Market basket analysis is the typical example of association rule mining [4].

However, on one hand rules explosion is a problem of concern, as conventional mining algorithms often produce too many rules for decision makers to digest [5]; on the other hand, some rules can miss important information; Addition, the rules of many algorithms are limited to single item. To overcome the shortcomings, we improve on the Apriori Algorithm with EFFICIENT Algorithm. Such a rule set can mine interesting knowledge rapidly without any information loss. At the same time, it can save a large of time and become simpler.

The paper concentrates on the EFFICIENT Algorithm and the recommend procedure of personalize commodities. We begin by show the procedure in section 2. We make clear the basic concept of association rule mining and the EFFICIENT Algorithm in section 3. And section 4 present an example. Finally, we summarize the conclusion and the future research work in section 5.

2. PROCEDURE

Data mining is an interactive and iterative process [2]. “How are association rules mined from large database?” General speaking, association rule mining includes the following two-step process [4].

STEP I: Find all frequent itemsets: By definition, each of these itemsets will occur at least as frequently as a pre-determined minimum support count.

STEP II: Generate strong association rules from the frequent itemsets: By definition, these rules must satisfy minimum support and minimum confidence.

In this paper, we only focus on the recommendation problem which help customer to find which products they would like to purchase for every one at a specific time; at the same time, the companies can discovery the
rule and recommend related product to their gods. From fig 1, we can see the flow of data mining in EC era.

Fig. 1: Flow of data mining

Fig. 2 outlines the procedure of data mining steps. It includes six steps which are cycle \cite{2} \cite{3}.

2.1 Raw data pre-processing

A database consists of current and historical detailed item, summarized data items and metadata. \cite{2} We select database and create a fact table; Next, selecting attributes and dimensions; Finally, we deal with the data, such as filtering data.

2.2 Data Transforming and Standardization

In this step, we transform data and standardize them to prepare to discover the relations of the commodities which have been recorded in computers.

2.3 Determine Active Customers

Making a commendation only for customers who are likely to buy recommended commodities could be avoid the false positives of a poor recommendation. In this step, we can use decision tree induction to perform the task of selecting active customers.

2.4 Select Algorithm of Mining

In step4, we should select an appropriate algorithm, although a lot of algorithm can solve mining problem. Some ones have limits which can not sit for many conditions, and some others are complex and invalid. Therefore, we need an efficient algorithm which can fast and simple solve the mining work. EFFICIENT Algorithm which be provided with the character will be referred in section 3.

2.5 Discovering Product Affinity & Generate Rules

In this step, we first search for meaningful relationships or affinities among product classes through mining association rule from large transaction.

Finding association rules at level-k by the following steps and the process of mining will be detailed showed in section4. 1) Get the consequents of rules from \( l_k \) with one item in the consequent; 2) Set up minimum support and minimum confidence. At different lever set the min-supp and min-conf different value. Such is distinctness from the traditional threshold;3) Call acceleration (large \( k \)-itemset, set of \( m \)-item consequents); 4)Call the predigestion Algorithm; 5)Obtain the affinity between the stander commodities and target commodities.

2.6 Add Related Commodities

Fig 3: Add commodities to standard plan

<table>
<thead>
<tr>
<th>Related Commodities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

Fig 3: Add commodities to standard plan
We find related commodities are strongly related to the target commodities, add the related commodities to the standard plan and get the specific sales promotion proposal plan which is prepared to perform the recommendation. This last step can be shown in fig 3.

The following is the last procedure, but it is the one which realizes the recommendation. It is the important phase which verifies the mining work is useful and efficiency or not.

In fact, the process of mining is a cycle one. When finish a recommendation, the new sale date will be next mined, the consequence still will be done …… At the same time, the rule will be corrected. We can name this process dynamic process.

3. ALGORITHM

“How can we help or guide the mining procedure to discovery interesting association? What language constructs are useful in defining a data mining query language for association rule mining?”[4] In this section, we will find these answers.

3.1 Preliminary Description

Let $J=\{i_1, i_2, \ldots i_m\}$ be a set of items. Let $D$ be a set of database transactions where each transaction $T$ is a set of items such that $T \subseteq J$. Each transaction is associated with an identifier, called TID. Let $A$ be a set of items. A transaction $T$ is said to contain $A$ if and only if $A \subseteq T$. An association rule is an implication of the form $A \Rightarrow B$, where $A \subseteq J$, $B \subseteq J$, and $A \cap B = \emptyset$. The rule $A \Rightarrow B$ holds in the transaction $T$ that contain $A \cup B$. This is taken to be the probability, $P(A \cup B)^[4]$.

Rule support and confidence are two measure of rule interestingness. They respectively reflect the usefulness and certainty of discovery rules. The following is the description of them:

Support ($A \Rightarrow B$) = $P(AB) = P(A \cup B)$  \hspace{1cm} (1)

Confidence ($A \Rightarrow B$) = $P(B \mid A) = P(A \cup B)/P(A)$  \hspace{1cm} (2)

If both the support and confidence are equal, we can use equation (3). If the reverse confidence ($A \Rightarrow B$) $= 1.0$, when $B$ appears, $A$ must also appear$^[3]$. Reverse confidence ($A \Rightarrow B$) = $P(B \mid A) = P(A \cup B)/P(B)$  \hspace{1cm} (3)

3.2 Apriori Algorithm

Agrawal, Jmielinski, and Swarmi introduced the notion of association rules in 1993. The basic Apriori algorithm regarded as a conventional method, was developed by Agrawal and Srikant (1994) and Uuama Fayyad, and Uthurusamy(1994). Many research efforts have then been made in two directions: one is to extend the notion of association rules, giving rise to various extensions such as generalized association rules, fuzzy association rules, etc; the other is to improve the algorithm in various ways such as fast algorithms, sampling algorithm, and so on.

Apriori is an influential algorithm for mining frequent itemsets for Boolean association rules. It employs an iterative approach known as a level-wise search, where $k$-itemsets are used to explore $(k+1)$-itemsets. First, find the set of frequent 1-itemsets. This set is denoted $L_1$, which is used to find $L_2$, the set of frequent 2-itemsets used to find $L_3$,... That all nonempty subsets of a frequent itemset must also be frequent is the Aprior property$^[4]$.

The Apriori Algorithm finds the frequent itemsets by using candidate generation. In order to find $L_k$ from $L_{k-1}$, a two-step process which consist join and prune should be done. If $(l_1[1]=l_2[1]) \land (l_1[2]=l_2[2]) \land \ldots \land (l_1[k-2]=l_2[k-2]) \land (l_1[k-1] < l_2[k-1])$, $l_1$ and $l_2$ of $L_{k-1}$ can join, where the condition $l_1[k-1] < l_2[k-1]$ simple ensure that no duplicates are generated. $C_k$, the set of candidates, is a superset of $L_k$, that is, its members may or may not be frequent, but all of the frequent $k$-itemsets are included in $C_k$. If any $(k-1)$-subset of a candidate $k$-itemset is not in $L_{k-1}$, then the candidate cannot be frequent either and so can be removed from $C_k$.$^[4][5]$

3.3 EFFICIENT Algorithm

EFFICIENT Algorithm is a fast and simple algorithm, which can save a great deal of time without missing useful information. Now let we see many properties of the algorithm$^[6][9]$. First, EFFICIENT Algorithm is simple, in some cases, it is enough, does not need other complex one to solve simple question, such as we can use association rule to class. Second, it count very fast, can decrease the work of CPU, then save significant time. Next, the algorithm pierces into limits of the single item which is required by some algorithms. Finally, generating the focused and interested rules does not need to obtain all rules without miss information.

The flowing is the EFFICIENT algorithm, which includes “acceleration” and “predigestion” actions$^[3][6][7]$. 

```
// EFFICIENT Algorithm
1) for all large k-itemsets $l_k$, k≥2 do begin
2) C_{l1} = \{ frequent 1-itemsets\} ;
3) If (the value of consequent item = 1) call acceleration ($l_k$, C_k);
4) else call predigestion ($l_k$, C_k);
5) end

//acceleration algorithm
7) Procedure acceleration
8) (l_k, C_m : set of m-item consequents, x)
9) x=x+1;
10) if (k>m+1) then begin
11) C_m+1 = acceleration (C_m);
12) for all C_{m+1}\in C_m do begin
13) conf = supp(C_m)/supp(C_{m+1});
```

13) if (conf ≥ minconf) then
14) output the rule \( l_k \rightarrow c_{m+1} \)
with confident=conf and support=supp\((l_k)\);
15) else delete \( c_{m+1} \) from \( C_{m+1} \);
16) end
17) call acceleration \((l_k, C_{m+1})\);
18) end
//predigestion algorithm
19) Procedure predigestion \((l_k, c_m)\)
20) \( C = [\{m-1\}-\text{itemsets} | c_{m-1} \subset c_m] \);
21) for all \( c_{m-1} \in C \) begin
22) \( \text{conf} = \frac{\text{supp}(l_k)}{\text{supp}(c_{m-1})} \);
23) if (conf ≥ minconf) then begin
24) output the rule \( c_{m-1} \rightarrow (l_k-c_{m-1}) \)
with confidence=conf(l_k) and support=supp(l_k);
25) if \((m-1>1)\) then call predigestion \((l_k,c_{m-1})\);
26) end
27) end

Fig 4: The EFFICIENT Algorithm

To generate rules, for every large itemset \( l \), we find all non-empty subset of \( l \). For every such subset \( c \), we output the rule: \( c \rightarrow (l-c) \), if the support is at least \( \text{minsupp} \). The algorithm does not miss any rule because the support of any supplementary item, \( c' \) of \( c \) must be as great as the support of \( c \). So the confidence of the rule \( c' \rightarrow (l-c') \) cannot be more than the confidence of \( c \rightarrow (l-c) \). By rewriting, if rule \( (l-c) \rightarrow c \) hold, all rules of the form \( (l-c') \rightarrow c' \) must also hold (\( c' \) is non-empty subset of \( c \)). If \( AC \rightarrow BD \) holds, then \( ABC \rightarrow D \) or \( ACD \rightarrow B \) must also hold. When call acceleration \((ABCD, ABCE)\), we can find that \( ACE \rightarrow BD \) is hold, but \( ABC \rightarrow DE \) is not [4][8].

Proof: \( ACE \rightarrow BD \)

\[
\wedge \text{Dconf}(X \Rightarrow YZ) = \text{Dconf}(X \Rightarrow Y) \wedge \text{Dconf}(XY \Rightarrow Z) \\
\text{Dconf}(X \Rightarrow Y) \geq \text{Dconf}(XY \Rightarrow Z) \\
\text{Dconf}(XY \Rightarrow Z) \geq \text{Dconf}(X \Rightarrow YZ), \text{ and} \\
\text{supp}(XY \Rightarrow Z) = \text{supp}(X \Rightarrow YZ) \leq \text{Dsupp}(X \Rightarrow Y) \\
\wedge XY \Rightarrow Z \Rightarrow X \Rightarrow YZ \\
\] Let \( X=ACE, Y=B, Z=D \), then get \( ABCE \rightarrow D \) from \( ACE \rightarrow BD \), that is \( ACE \rightarrow BD \) can be expressed by \( ABCE \rightarrow D \).

4. EXAMPLE

4.1 Generate Candidate & Frequent Itemsets

This study uses the example to describe the program execution of EFFICIENT Algorithm. The example includes seven transaction records, as show in table 1.

Table 1: The table of database D

<table>
<thead>
<tr>
<th>TID</th>
<th>List of item</th>
<th>TID</th>
<th>List of item</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>ABCDE</td>
<td>T5</td>
<td>ABCE</td>
</tr>
<tr>
<td>T2</td>
<td>BCE</td>
<td>T6</td>
<td>ACE</td>
</tr>
<tr>
<td>T3</td>
<td>ACD</td>
<td>T7</td>
<td>ABCE</td>
</tr>
<tr>
<td>T4</td>
<td>BE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig 5: Generate of candidate and frequent itemsets

The minimum length threshold of large itemsets which at different phase is different value, we let the number of items of the first phase be 3 while the min-sup is defined based on the generation of itemsets in various lengths. In the following phases, we can find that the min-sup is changing, that does not appear in additional algorithm. The result is used to prepare the next step which mines the rules.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>lim1</td>
<td></td>
<td>4/7</td>
<td>80%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lim2</td>
<td></td>
<td>3/7</td>
<td>75%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lim3</td>
<td></td>
<td>3/7</td>
<td>75%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3/7</td>
<td>75%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: All association rules
Table 3: Derived rules

<table>
<thead>
<tr>
<th>No.</th>
<th>Rules Derived from</th>
</tr>
</thead>
<tbody>
<tr>
<td>#18</td>
<td>B (\Rightarrow) CE {3:B (\Rightarrow) C, 9:BC (\Rightarrow) E}</td>
</tr>
<tr>
<td>#19</td>
<td>A (\Rightarrow) CE {1:A (\Rightarrow) C, 6:AC (\Rightarrow) E}</td>
</tr>
<tr>
<td>#20</td>
<td>AB (\Rightarrow) CE {12:AB (\Rightarrow) C, 14:ABC (\Rightarrow) E}</td>
</tr>
<tr>
<td>#21</td>
<td>AE (\Rightarrow) BC {7:AE (\Rightarrow) C, 16:ACE (\Rightarrow) B}</td>
</tr>
<tr>
<td>#22</td>
<td>BC (\Rightarrow) AE {9:BC (\Rightarrow) E, 17:BCE (\Rightarrow) A}</td>
</tr>
<tr>
<td>#23</td>
<td>A (\Rightarrow) BCE {19:A (\Rightarrow) CE, 16:ACE (\Rightarrow) B}</td>
</tr>
<tr>
<td>Replaced by</td>
<td>{1:A (\Rightarrow) C, 6:AC (\Rightarrow) E, 16:ACE (\Rightarrow) B}</td>
</tr>
</tbody>
</table>

4.2 Mine Rule by Using EFFICIENT Algorithm \(^{[3][9]}\)

The following is presented to help illustrate the idea of EFFICIENT Algorithm. If we let the minconf and minsupport have different values, we can get the following table which has some new or different rules.

From the process of mining, we can find that it is more efficient and easier than before. The different between mining for EFFICIENT Algorithm and mining for all association rules is that, to extract the former from any k-frequent itemset \(i_1,i_2,...,i_k\). The frequent itemset consisting of k-items, one only needs to examine certain frequent itemsets, namely its (k-1)-frequent sub-itemsets instead of all j-frequent sub-itemsets \(j-1,2,...,k-1\). In fact, only \(k(k-1)\) sub-itemsets can be checked, but \(2^k-2\) sub-itemsets should be done in all association rules.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we introduce a kind of algorithm, EFFICIENT Algorithm which retains all information in the original rule set and save mass time, but has a very smaller size and discovery all significant association rules between items in a large database of transactions. We can draw a conclusion that all qualified association rules can be derived from the EFFICIENT algorithm. The next work is being conducted on further algorithmic optimizations. Besides, more efficient association rules mining may also be the future work. In addition, we will attempt to use the technology of array to solve the mining work, which is a spick-and-span exploration. We believe that data mining is an important new application area for database, combining commercial interest with diverting research questions.

ACKNOWLEDGEMENT

This research was supported by the National Society Science Foundation of China, the Research on the Informationlize Model and the System of Information Management oriented MC.

REFERENCES