Available online at www.sciencedirect.com

ScienceDirect

Expert Systems with Applications 35 (2008) 1422–1429

Expert Systems with Applications

www.elsevier.com/locate/eswa

Towards personalized recommendation by two-step modified Apriori data mining algorithm

Enrique Lazcorreta [a,b], Federico Botella [a,b], Antonio Fernández-Caballero [c,d,*]

[a] Instituto Universitario Centro de Investigación Operativa (CIO), Universidad Miguel Hernández de Elche, 03202 Elche, Spain
[b] Departamento de Estadística, Matemáticas e Informática, Universidad Miguel Hernández de Elche, 03202 Elche, Spain
[c] Instituto de Investigación en Informática de Albacete (I3A), Escuela Politécnica Superior de Albacete,
Universidad de Castilla-La Mancha, 02071 Albacete, Spain
[d] Departamento de Sistemas Informáticos, Escuela Politécnica Superior de Albacete, Universidad de Castilla-La Mancha, 02071 Albacete, Spain

Abstract

In this paper a new method towards automatic personalized recommendation based on the behavior of a single user in accordance with all other users in web-based information systems is introduced. The proposal applies a modified version of the well-known Apriori data mining algorithm to the log files of a web site (primarily, an e-commerce or an e-learning site) to help the users to the selection of the best user-tailored links. The paper mainly analyzes the process of discovering association rules in this kind of big repositories and of transforming them into user-adapted recommendations by the two-step modified Apriori technique, which may be described as follows. A first pass of the modified Apriori algorithm verifies the existence of association rules in order to obtain a new repository of transactions that reflect the observed rules. A second pass of the proposed Apriori mechanism aims in discovering the rules that are really inter-associated. This way the behavior of a user is not determined by "what he does" but by "how he does". Furthermore, an efficient implementation has been performed to obtain results in real-time. As soon as a user closes his session in the web system, all data are recalculated to take the recent interaction into account for the next recommendations. Early results have shown that it is possible to run this model in web sites of medium size.

Keywords: Personalization; Data mining; Apriori-like algorithm; Recommendation

1. Introduction

In recent years, companies have concentrated on understanding the needs and expectations of their customers and grouping the existing and potential customers into classes with the purpose of improving the efficiency of their marketing strategies and increasing their market share (Saglam, Salman, Sayin, & Türkay, 2006). Personalization has become a reality and is possible by using efficient methods of data mining and knowledge discovery (Kim & Cho, 2007). To date, a variety of recommendation techniques has been developed (Cho, Kim, & Kim, 2002). Through analyzing user related information, it is possible to make a more accurate analysis of customer's interest or preference. In most cases, recommendation can be classified according to (1) whether customers for whom we want recommendations are all customers or selective customers, (2) whether the objective of recommendation is to predict how much a particular customer will like a particular product, or to identify a list of products that will be of interest to a given customer (top-*N* recommendation problem), and (3) whether the recommendation is accomplished at a specific time or persistently.

The abundance of large data collections and the need to extract hidden knowledge within them has triggered the

[*] Corresponding author. Address: Departamento de Sistemas Informáticos, Escuela Politécnica Superior de Albacete, Universidad de Castilla-La Mancha, 02071 Albacete, Spain. Tel.: +34 967599200; fax: +34 967 599224.

E-mail addresses: enrique@umh.es (E. Lazcorreta), federico@umh.es (F. Botella), caballer@dsi.uclm.es (A. Fernández-Caballero).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.08.048

development of algorithms to detect unknown patterns in data sets (Han & Kamber, 2001). A paper (Ozmutlu, Spink, & Ozmutlu, 2002) reports results from a study using Poisson sampling to develop a sampling strategy to demonstrate how sample sets selected by Poisson sampling statistically effectively represent the characteristics of the entire data set. Moreover, clustering analysis is a data mining technique developed for the purpose of identifying groups of entities that are similar to each other with respect to certain similarity measures. In the past, different ways to discover groups using clustering techniques have been proposed (Schafer, Konstan, & Riedl, 2001). Very often, they are based on different definitions of similarity measure to represent the closeness between users. Users can also be grouped based on the transactions they perform (Wang, Lim, & Hwang, 2006). In Perkowitz and Etzioni (2000) a cluster mining algorithm – an unsupervised algorithm for efficiently identifying a small set of high-quality (and possibly overlapping) clusters with limited coverage – is introduced.

Nevertheless, the existing researches could not afford to give a formal way for capturing individual customer's preference or associations among products through web usage mining. Given a set of transactions where each transaction is a set of items (itemset), an association rule implies the form $X \Rightarrow Y$, where X and Y are itemsets; X and Y are called the body and the head, respectively. The support for the association rule $X \Rightarrow Y$ is the percentage of transactions that contain both itemset X and Y among all transactions. The confidence for the rule $X \Rightarrow Y$ is the percentage of transactions that contain itemset Y among transaction that contain itemset X. The support represents the usefulness of the discovered rule and the confidence represents certainty of the rule. Association rule mining is the discovery of all association rules that are above a user-specified minimum support *minsup* and minimum confidence *minconf* (Tseng & Lin, 2007). Apriori algorithm is one of the prevalent techniques used to find association rules (Agrawal, Imielinski, & Swami, 1993; Agrawal & Srikant, 1994). Apriori operates in two phases. In the first phase, all itemsets with minimum support (frequent itemsets) are generated. This phase utilizes the downward closure property of support. In other words, if an itemset of size k is a frequent itemset, then all the itemsets below $(k-1)$ size must also be frequent itemsets. Using this property, candidate itemsets of size k are generated from the set of frequent itemsets of size $(k-1)$ by imposing the constraint that all subsets of size $(k-1)$ of any candidate itemset must be present in the set of frequent itemsets of size $(k-1)$. The second phase of the algorithm generates rules from the set of all frequent itemsets.

Association rule mining, as originally proposed in Agrawal et al. (1993) with its Apriori algorithm, has developed into an active research area. Association rule discovery and classification are analogous tasks in data mining, with the exception that classification main aim is the prediction of class labels, while association rule mining discovers associations between attribute values in a data set (Thabtah, Cowling, & Hammoud, 2006). Many additional algorithms have been proposed for association rule mining (e.g. Pujari, 2001; Lin & Kedem, 2002). End users of association rule mining tools encounter several well-known problems in practice. First, the algorithms do not always return the results in a reasonable time. A fuzzy mining algorithm based on the AprioriTid approach to find fuzzy association rules from given quantitative transactions has been proposed for reduced time complexity (Hong, Kuo, & Wang, 2004). Further, the association rules sets are sometimes very large. In Palshikar, Kale, and Apte (2007) a concept called a heavy itemset is proposed to compactly represent the association rules. An algorithm named as BitTableFI (Dong & Han, 2007) has significant difference from the Apriori and all other algorithms extended from Apriori. It compresses the database into BitTable, and with the special data structure, candidate itemsets generation and support count can be performed quickly. Also, mining association rules with multiple minimum supports is an important generalization of the association rule mining problem. Instead of setting a single minimum support threshold for all items, (Liu, Hsu, & Ma, 1999) allow users to specify multiple minimum supports to reflect the natures of the items, and an Apriori-based algorithm, named MSapriori, is developed to mine all frequent itemsets. In a recent paper (Hu & Chen, 2006), the same problem is suited but with two additional improvements. Another approach is the cluster-based association rule (CBAR) method (Tsay & Chiang, 2005), aimed to create cluster tables by scanning the database once, and then clustering the transaction records to the kth cluster table, where the length of a record is k. Moreover, the large itemsets are generated by contrasts with the partial cluster tables. This not only prunes considerable amounts of data reducing the time needed to perform data scans and requiring less contrast, but also ensures the correctness of the mined results. In Lee, Hong, and Lin (2005) another point of view about defining the minimum supports of itemsets when items have different minimum supports is provided. The maximum constraint is used, and then a simple algorithm based on the Apriori approach to find the large-itemsets and association rules under this constraint is introduced. Another interesting proposal is to utilize methods and techniques from Information Retrieval (IR) in order to assist data mining functions (Kouris, Makris, & Tsakalidis, 2005).

In this paper a new method towards automatic personalized recommendation based on the behavior of a single user in accordance with all other users in web-based information systems is introduced. The proposal applies a modified version of the well-known Apriori data mining algorithm to the log files of a web site to guide the users to the selection of the best user-tailored links. The paper mainly analyzes the process of discovering association rules in this kind of big repositories and of transforming them into user-adapted recommendations.

2. The Apriori2 approach

As stated before, the discovery of association rules between items of a transaction set has been undertaken by many researchers. However, when defining user behavior patterns for one service (like a supermarket, an e-learning site, or simply any website) we should not only base on the analysis of the items composing their transactions. For instance, the behavior of a user in a website can not be well-measured by the pages he visits (what); we also need to know the way the user visits these pages (how), in order to differentiate him from other users or to group him to other users with similar behavior. This leads to take a step beyond from simple association rules discovery generated by the users of the service, that is to say, the relationship existing between the whole set of users and each one of their individuals has to be analyzed.

To perform this analysis the following steps are performed: (1) discovering existing association rules in the original repository of transactions; (2) rewriting the original transactions to reflect all association rules that verify each one of them; (3) analyzing differences existing between both sets of transactions by means of statistical analysis; and, (4) discovering and using the existing relations between rules discovered in step 1 in order to automatically divide the set of transaction rules obtained in step 2.

We can define the behavior of a user at a web service with the data obtained according to the manner he acts in the service, i.e. according to particular rules verified by the user whilst he is interacting with the service. Next the four steps of our approach are explained in detail.

2.1. Discovering existing association rules in the original repository of transactions

Firstly, we use the original algorithm (Agrawal & Srikant, 1994) for discovering the association rules between items generated by users, although we have added some modifications to their implementation in order to obtain an acceptable performance without loosing any of the discovered data. The original pseudocode by Agrawal and Srikant is offered in Algorithm 1.

Algorithm 1. The original Apriori algorithm.

```
$L_1 = \{\text{large 1-itemsets}\}$;
for ($k = 2; L_{k-1} \neq \emptyset; k++$) do begin
    $C_k$ = apriori-gen($L_{k-1}$); //New candidates
    forall transactions $t \in \mathscr{D}$ do begin
        $C_t$ = subset($C_k$,$t$); //Candidates contained in $t$
        forall candidates $c \in C_t$ do
            $c$.count++;
    end
    $L_k = \{c \in C_k | c.\text{count} \geqslant \text{minsup}\}$;
end
Answer $= \cup_k L_k$;
```

In the original algorithm `apriori-gen` is a function made up of two phases: union and pruning. In the union phase (see Algorithm 2), all k-itemsets candidates are generated.

Algorithm 2. Union phase of the original Apriori.

```
insert into $C_k$
select $p.\text{item}_1, \ldots, p.\text{item}_{k-1}$, $q.\text{item}_{k-1}$
from $L_{k-1} p, L_{k-1}\ q$
where    $p.\text{item}_1 = q.\text{item}_1, \ldots, p.\text{item}_{k-2} = q.\text{item}_{k-2}$,
$p.\text{item}_{k-1} < q.\text{item}_{k-1}$;
```

Now in the pruning phase (see Algorithm 3), which gives the name to the algorithm, all candidates generated in the union phase with some non-frequent $(k-1)$-itemset are removed.

Algorithm 3. Pruning phase of the original Apriori.

```
forall itemsets $c \in C_k$
    forall $(k-1)$-subsets $s$ of $c$ do
        if $s \notin L_{k-1}$ then
            delete $c$ from $C_k$;
```

In Algorithm 4 our particular new implementation of the union and pruning phases for the Apriori algorithm is given. When joining the union and pruning phases in a same function, many insert and delete operations in a dynamic vector C_k are saved. Also by relaxing the pruning many search operations in tree L of frequent k-itemsets are saved.

Algorithm 4. Union and pruning phases of the modified Apriori.

```
insert into $C_k$
select $c = \{p.\text{item}_1, \ldots, p.\text{item}_{k-1}, q.\text{item}_{k-1}\}$
from $L_{k-1} p, L_{k-1} q$
where    $(p.\text{item}_1 = q.\text{item}_1, \ldots, p.\text{item}_{k-2} = q.\text{item}_{k-2}$,
$p.\text{item}_{k-1} < q.\text{item}_{k-1})$
    and
    $(p.\text{item}_{k-1}, q.\text{item}_{k-1}) \in L_2$
```

Another important adjustment to the original Apriori algorithm is the extraction of the existing rules (function `genrules`) in the repository of transactions. Again, we offer the code for the original Apriori in Algorithm 5.

Algorithm 5. The original Apriori genrules.

```
// Simple Algorithm
forall large itemsets $l_k$, $k \geqslant 2$ do
    call genrules($l_k$, $l_k$);
// The genrules generates all valid rules $\tilde{a} \Rightarrow (l_k - \tilde{a})$, for
all $\tilde{a} \subset a_m$
procedure genrules($l_k$: large $k$-itemset, $a_m$: large $m$-
itemset)
    $A = \{(m-1)\text{-itemsets } a_{m-1} | a_{m-1} \subset a_m\}$;
```

```
(2)  forall $a_{m-1} \in A$ do begin
(3)    $conf = \text{support}(l_k)/\text{support}(a_{m-1})$;
(4)    if ($conf \geqslant minconf$) then begin
(7)      output the rule $a_{m-1} \Rightarrow (l_k - a_{m-1})$, with confidence = $conf$ and support=support($l_k$);
(8)      if ($m - 1 > 1$) then
(9)        call genrules($l_k$, $a_{m-1}$); // to generate rules with subsets of $a_{m-1}$ as the antecedents
(10)   end
(11) end
```

To denote a rule, we use the notation

$$R_i = \{a_i \Rightarrow c_i\} \tag{1}$$

where R_i is the ith rule, a_i the antecedent k-itemset and c_i is the consequent k-itemset.

By

$$l_{k_i} = a_i \cup c_i \tag{2}$$

we denote the k-itemset that defines rule R_i.

`genrules` takes two k-itemsets as parameters: the first one is always l_{k_i} and the second is a subset of l_{k_i}, from which the antecedents of the rules derived from l_{k_i} are extracted. Each time `genrules` is called, we obtain the confidence of a rule as $a_m \Rightarrow c_i = l_k - a_m$, so we need two calls to the `support` function. This function has to pass over L until the level determined by the number of items of the k-itemset received: $k + (k - m)$ searches.

This characteristic makes a little bit inefficient the original algorithm due to the way the repository L of frequent k-itemsets is stored. Therefore we have to search for the items that compose l_k from the root of the repository. Moreover, we have added the support of l_k as parameter to `genrules` to get a more efficient implementation of the algorithm, which avoids many calls to the `support` function (only one call is performed before processing each k-itemset of L and in the multiple recursive callings this function is not called again with this parameter).

The creators of the Apriori algorithm followed the next statement:

> *If $a \Rightarrow (l - a)$ does not hold, neither does $\tilde{a} \Rightarrow (l - \tilde{a})$ for any $\tilde{a} \subset a$. By rewriting, it follows that for a rule $(l - c) \Rightarrow c$ to hold, all rules of the form $(l - \tilde{c}) \Rightarrow \tilde{c}$ must also hold, where $\tilde{c}$ is a non-empty subset of c. For example, if the rule $AB \Rightarrow CD$ holds, then the rules $ABC \Rightarrow D$ and $ABD \Rightarrow C$ must also hold.*

After that, they proposed a second method, quicker under certain circumstances: when *minconf* is large, if they can detect that $conf(\text{ABC} \Rightarrow \text{D}) < minconf$ then they assume that it is not necessary to check the rules AB ⇒ CD, AC ⇒ BD, BC ⇒ AD, A ⇒ BCD, B ⇒ ACD, C ⇒ ABD. In the case they want to detect rules without high confidence, this is rather a disadvantage that supposes an additional verification and will slow down the execution of the algorithm.

Moreover, we improve the performance of the algorithm by analyzing the characteristics of the rules generated by it. Let's see an example:

Let the k-itemset be $l_k = \{1,2,3,4,5,6\}$.

- `genrules`(l_k, l_k) will recursively call `genrules`(l_k, {1,2,3,4,5}) and this will call recursively `genrules`(l_k, {1,2,3,4}).
- Later, the initial reference will call `genrules`(l_k, {1,2,3,4,6}) and this will make another recursive call to `genrules`(l_k, {1,2,3,4}).

Afterwards, `genrules`(l_k, {1,2,3,4}) is called twice, and thus we will get duplicates of all the performed callings with the subsets of {1,2,3,4} as the second parameter. This generates a great number of searches and repeated operations, which will be of exponential order k.

In our implementation this problem is overcome by the use of a static vector where the analyzed subsets of l_k are stored as second parameter. In order to apply these properties we have redefined the algorithm as depicted in Algorithm 6.

Algorithm 6. The modified Apriori genrules.

```
forall large itemsets $l_k, k \geqslant 2$ do
$supp\_l\_k = \text{support}(l_k)$;
call genrules($l_k$, $l_k$, $supp\_l\_k$);
// The genrules generates all valid rules $\tilde{a} \Rightarrow (l_k - \tilde{a})$, for all $\tilde{a} \subset a_m$
procedure genrules($l_k$: large $k$-itemset, $a_m$: large $m$-itemset, $supp\_l\_k$: double)
(x)  static a_m_processed; //set containing the sets of $l_k$ processed in previous calls to genrules
(x)  if ($m == k$)
(x)    a_m_processed.clear(); //Initialize a_m_processed for each new $l_k$
(1)  $A = \{(m-1)\text{-itemsets } a_{m-1} | a_{m-1} \subset a_m\}$;
(2)  forall $a_{m-1} \in A$ do begin
(x)    if $a_{m-1} \in$ a_m_processed then
(x)      continue;
(x)    a_m_processed.add($a_{m-1}$);
(3)    $conf = supp\_l\_k/\text{support}(a_{m-1})$;
(4)    if ($conf \geqslant minconf$) then begin
(7)      output the rule $a_{m-1} \Rightarrow (l_k - a_{m-1})$, with confidence = $conf$ and support = support($l_k$);
(8)      if ($m - 1 > 1$) then
(9)        call genrules($l_k$, $a_{m-1}$, $supp\_l\_k$); // to generate rules with subsets of $a_{m-1}$ as the antecedents
(10)   end
(11) end
```

The results obtained in several executions of the original vs. our modified Apriori algorithm are shown in Figs. 1 and 2.

As you may appreciate, when getting all existing rules in the repository ($minConf = 0\%$) the execution time is

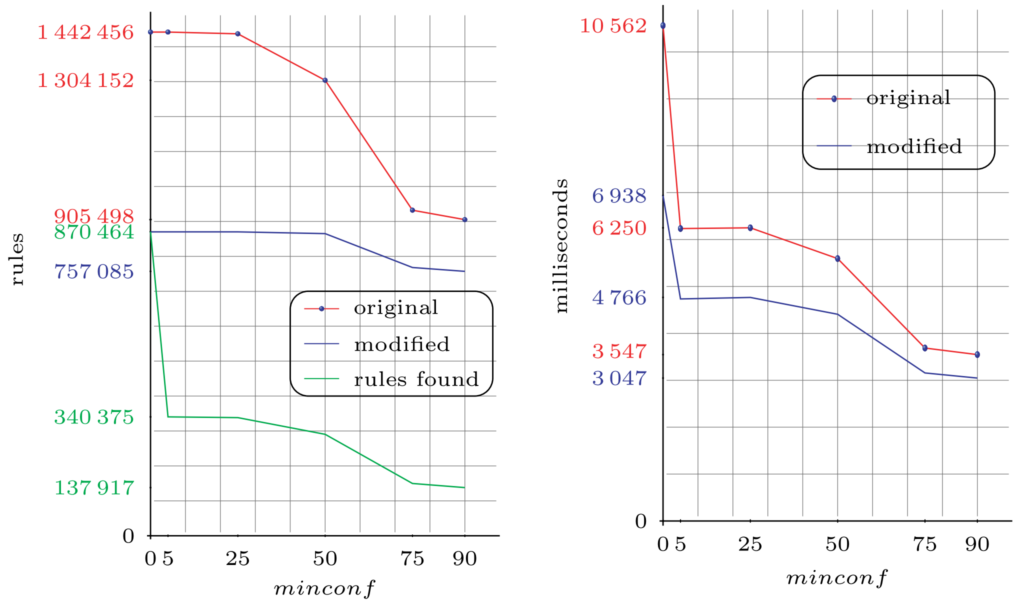

Fig. 1. Rules analyzed by the original vs. our modified Apriori algorithm along time using foodmart database with *minsup* = 0.005%.

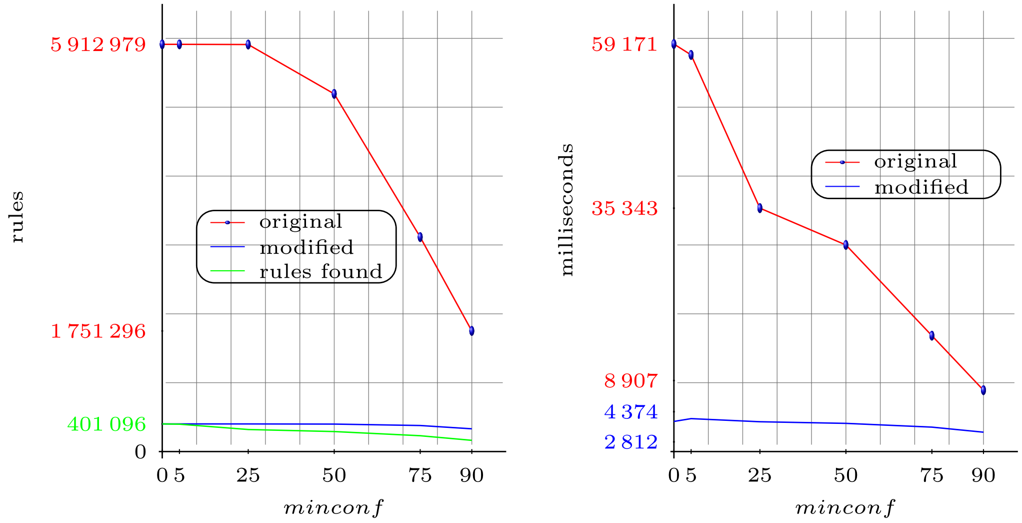

Fig. 2. Rules analyzed by the original vs. our modified Apriori algorithm along time using T40I10D100K database with *minsup* = 1%.

reduced between a 1276% and a 13,459% as we only analyze once the rules when the k-itemsets that generate the rules appear at first time in L. The original algorithm repeats the analysis of rules and results overloaded by a 12,309%, because even with the improvement proposed by the authors no rules are excluded in most of the analysis performed. Our modified algorithm does not check any extra rule when *minconf* is set to 0%. We also want to highlight that the execution time of our modified algorithm is not excessive at all, so it could be executed in real-time. If applied to the transactions performed by same user or group of users with homogeneous behavior in a website, this modified algorithm even gets lower execution times.

2.2. Rewriting transactions to return association rules verified by each transaction

It is more appropriate to study the behavior of a user according to the rules he “verifies” rather than to the items themselves. Suppose that a customer uses to enter in a website on Mondays and visits pages A, B and C and next a series of pages P_1. But he visits pages A,G and H on Fridays and next another series of pages P_2. Suppose also that the pages of P_1 and P_2 are not frequently presented with page A, and as our recommender system is not considering temporal variables, we can not detect if it is Monday or Friday. Thus when a user enters in a website and requests for page

A, probably we will recommend him to visit pages B or G, if we were using the traditional methods of recommendation. If the user then visits page G, the method will suggest a link to page H as the more probable, next some links to pages P_2 and finally another link to page B with less probability to be visited by the user. Therefore the fact to visit page A is not as significant as the possibility to use the behavioral rules already stored (the existing relations between rules containing the k-itemsets AGH and P_2). The cue idea is the capacity of analyzing the rules independently of the number of items they contain, providing a means to get the relations between two rules composed by the different items and using only the transactions supplied by the users.

If we study the rules verified by each original transaction of $\mathscr{D}$, we can define a set of rules ($\mathscr{R}$) that contains one line for each verified rule of each transaction of $\mathscr{D}$. The goal of this conversion $\mathscr{D}\rightsquigarrow\mathscr{R}$ is to study again the group of rules containing $\mathscr{R}$ by means of Apriori algorithm.

At the beginning, this task is simply an exploratory task as we need many hours to convert file $\mathscr{R}$ into huge repositories of data and we have to do it for different support and confidence thresholds. Rule repositories use to be bigger than transaction repositories. Indeed, if a transaction verifies a rule that contains a given k-itemset l_k, then it will verify all rules of the k-itemsets contained in l_k. For example, if transaction $ABCD$ generates one rule, then it can generate the fifty rules inferred by their subsets:

- 14 rules with 4 items: $ABC \Rightarrow D$, $ABD \Rightarrow C, \ldots D \Rightarrow ABC$.
- 24 rules with 3 items: $AB \Rightarrow C, \ldots CD \Rightarrow B$.
- 12 rules with 2 items: $A \Rightarrow B, \ldots D \Rightarrow C$.

If we use a low confidence threshold, one transactio of $\mathscr{D}$ with three items will define twelve rules in $\mathscr{R}$, one transaction of $\mathscr{D}$ with four items will define 50 rules in $\mathscr{R}$, and then $\mathscr{R}$ will grow exponentially according to the length of transactions of $\mathscr{D}$.

At this point, the impressive size of $\mathscr{R}$ leads to an exploratory study of the existing differences between both repositories for proposing a reduction of data contained in $\mathscr{R}$ without losing any information contained in it.

2.3. Analyzing existing differences between both repositories using statistical exploration

In previous section 2.1 a modification of the rule generating algorithm has been introduced, enabling to work with

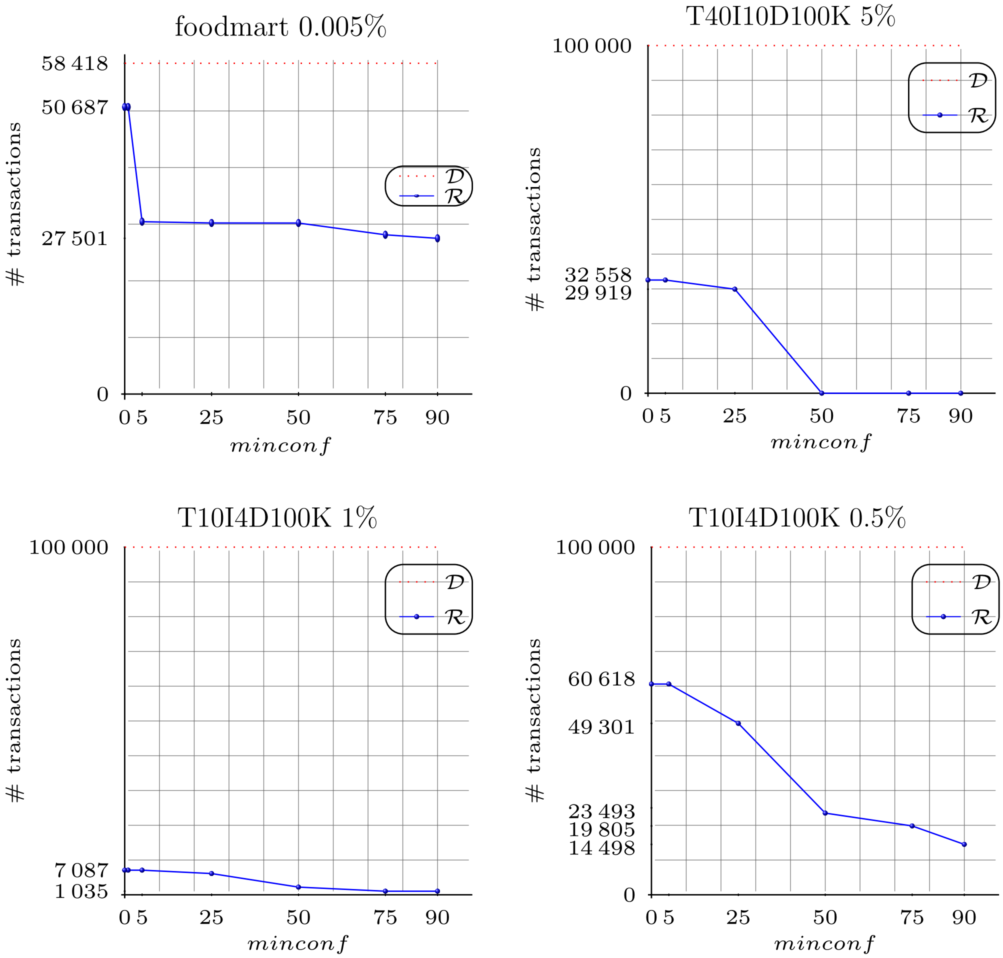

Fig. 3. Number of transaction of $\mathscr{R}$ vs. $\mathscr{D}$

low *minconf* values. The reason of searching this reduction (sometimes it is even a removal) of minimal confidence may be understood by analyzing the results shown in Fig. 3. As the value of *minconf* is increased we lose significance in the obtained rules; for instance, although we use a minimum support of 5% the obtained rules from repository T40I10D100K employ a maximum of 32.6% of the original transactions. That is, the data used in our recommender system ignore more than 67% of the original data.

We have found many approaches to deal with huge sets of transactions that apply easy statistical techniques based on frequency to reduce the dimension of the problem to solve. However, some details are not considered enough times; hence, the results are only meaningful to a reduced group of individuals from the population in our study.

In this work we use the minimum support and confidence values that a conventional personal computer is able to process without problems of overload and being able to process the highest number of individuals of the population studied, so we can not analyze directly the set $\mathscr{R}$. Then we propose a new algorithm to reallocate the information of $\mathscr{R}$ and nevertheless capable of processing the whole set $\mathscr{R}$.

2.4. Automatic division of the repository of rules

Many approaches about methods to divide the repository of original data exist nowadays, but all of them are based on a highest control and classification of the set of items available to the system. Our proposal does not use this additional information because it is obtained from a system manager and not from system usage data itself. We define a set of families of rules to divide the repository of rules and group those rules that link users when interacting with the system. The proposed algorithm is detailed next.

(1) Select the rule of major support; in case of draw, select the rule of highest confidence and in case of another draw, select the rule that contains more items.

$$R_1|supp(R_1) = max_i\{supp(R_i)\}\wedge (conf(R_1) = max_j\{conf(R_j)|supp(R_j) = supp(R_1)\})$$

This rule will be the principal element of the first family of rules, $\mathscr{F}_1$.

(2) Divide $\mathscr{R}$ into subsets of transactions. The first repository $\mathscr{R}_1$ will contain all transactions with the rule R_1 and the second repository $\mathscr{R}_\infty$ will contain the remaining transactions.

(3) Run the Apriori algorithm on $\mathscr{R}_1$ and apply step 1 again to select R_2, the rule with highest frequency jointly with R_1.

(4) Check the support of R_2 into $\mathscr{R}_\infty$: if the support of R_2 into $\mathscr{R}_1$ is greater than the support in $\mathscr{R}_\infty$, then add R_2 to family $\mathscr{F}_1$; in other case, remove R_2 from $\mathscr{R}_1$.

(5) Go to step 3 while there are rules still not classified in $\mathscr{R}_1$.

When the definition of the first family of rules has been accomplished, remove all rules belonging to $\mathscr{F}_1$ and all empty transactions from the original repository of rules $\mathscr{R}$. Next the second family of rules is constructed in the same manner. The algorithm finishes when $\mathscr{R}_\infty$ has no rules associated with other rules, i.e. when all transactions of $\mathscr{R}_\infty$ possess only one rule.

2.5. Our recommender system proposal

Finally, we have defined a recommender system where our two-step modified Apriori algorithm has been applied. The main goal is to obtain personal recommendations for the users visiting our educational website in link format and in real-time. The recommendation process can be defined as follows:

(1) A user enters into the system and selects item A.

(2) The system can only use the information collected about its proper items, that is to say the original rules with A as antecedent – this is the classical recommendation using association rules. Thus, the first recommendation will be solely based on frequency (the system will recommend the consequents of the rules with the highest confidence that have item A as antecedent).

(3) The user selects a second item B.

(4) Now we can already use the new information about the behaviors of the user population. We obtain the rules derived from the k-itemset l_k and search for the family which belongs to it, $\mathscr{F}_i$.

- If $\mathscr{F}_i$ contains rules derived from the rules accomplished by the user in his current visit, then there is a confidence of 100% between the discovered rules and the rules already verified by the user. In this case the system recommends the items of the discovered rules with the highest support. Unlike in more classical methods, here we discover the rules that have no items selected as antecedents, and this is very important as the performed analysis completely ignores the selection order of the items.
- If $\mathscr{F}_i$ does not contain such a kind of rules, the rules with highest confidence (and support in case of draw) in relation to the rules verified by the user are used. In addition, in order to solve the classical problem of discovering recommendations based on the antecedent, it is possible to find rules with any items already selected by the user. This can not be accomplished with the classical method.

(5) The user selects a third item C.

(6) We proceed as in step 4, but now looking for one or more families according to derived rules from k-itemset $l_k = \{A,B,C\}$.

This approach is guiding to new experiences within our educational systems, where users should feel a little more comfortable in their interactions with the system. They are able to perform at least one task more directly when using the recommended links offered by the system.

3. Conclusions

In this paper a new method towards automatic personalized recommendation based on the behavior of a single user in accordance with all other users in web-based information systems has been introduced.

The proposal introduces a modified version of the well-known Apriori data mining algorithm to the log files of a web site to guide the users towards the selection of the best user-tailored links. The paper mainly analyzes the process of discovering association rules in this kind of big repositories and of transforming them into user-adapted recommendations. The four main steps of our approach have been presented in detail. These are: (1) Mining the association rules present in the original transactions repository. The modified Apriori algorithm introduced enables mining the association rules in real-time. (2) Rewriting the transactions in such a way that they reflect the association rules verified for each transaction. (3) Analyzing the differences between both repositories by means of a statistical study. (4) Mining the association rules among the new transactions through a second run of the modified Apriori algorithm.

Most association rule mining algorithms suffer from the twin problems of too much execution time and generating too many association rules. Although conventional Apriori algorithm can discover meaningful itemsets and construct association rules, it suffers the disadvantage of also generating numerous candidate itemsets that must be repeatedly contrasted with the entire database. The processing of the conventional algorithm also utilizes large amounts of memory. Performance also is influenced, since the database is repeatedly read to contrast each candidate itemset with all of the transaction records in the database. In this paper, we have proposed a solution to address both problems.

Finally, our proposal has established a basis towards personalized recommendation. Our paper shows that our approach can provide better recommendation services by analyzing the behavior of a single user in accordance with all other users in web-based information systems.

Acknowledgement

This work is supported in part by the Spanish Junta de Comunidades de Castilla-La Mancha PAI06-0093 grant.

References

Agrawal, R., Imielinski, T., & Swami, A. (1993) Mining association between sets of items in massive database. In: *International proceedings of the ACM-SIGMOD international conference on management of data* (pp. 207–216).

Agrawal, R., & Srikant, R. (1994) Fast algorithms for mining association rules. In: *Proceedings of the international conference on very large data bases* (pp. 407–419).

Cho, Y. H., Kim, J. K., & Kim, S. H. (2002). A personalized recommender system based on web usage mining and decision tree induction. *Expert Systems with Applications, 23*, 329–342.

Dong, J., & Han, M. (2007). BitTableFI: An efficient mining frequent itemsets algorithm. *Knowledge-Based Systems, 20*, 329–335.

Han, J., & Kamber, M. (2001). *Data mining: Concepts and techniques*. Morgan Kaufmann Publisher.

Hong, T. P., Kuo, C. S., & Wang, S. L. (2004). A fuzzy AprioriTid mining algorithm with reduced computational time. *Applied Soft Computing, 5*, 1–10.

Hu, Y. H., & Chen, Y. L. (2006). Mining association rules with multiple minimum supports: A new mining algorithm and a support tuning mechanism. *Decision Support Systems, 42*, 1–24.

Kim, K. J., & Cho, S. B. (2007). Personalized mining of web documents using link structures and fuzzy concept networks. *Applied Soft Computing, 7*, 398–410.

Kouris, I. N., Makris, C. H., & Tsakalidis, A. K. (2005). Using Information Retrieval techniques for supporting data mining. *Data & Knowledge Engineering, 52*, 353–383.

Lee, Y. C., Hong, T. P., & Lin, W. Y. (2005). Mining association rules with multiple minimum supports using maximum constraints. *International Journal of Approximate Reasoning, 40*, 44–54.

Lin, D., & Kedem, Z. M. (2002). Pincer-search: An efficient algorithm for discovering the maximum frequent set. *IEEE Transactions on Knowledge Data Engineering, 14*, 553–556.

Liu, B., Hsu, W., & Ma, Y. (1999) Mining association rules with multiple minimum supports. In: *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 337–341).

Ozmutlu, H. C., Spink, A., & Ozmutlu, S. (2002). Analysis of large data logs: An application of Poisson sampling on excite web queries. *Information Processing and Management, 38*, 473–490.

Palshikar, G. K., Kale, M. S., & Apte, M. M. (2007). Association rules mining using heavy itemsets. *Data & Knowledge Engineering, 61*, 93–113.

Perkowitz, M., & Etzioni, O. (2000). Towards adaptive Web sites: Conceptual framework and case study. *Artificial Intelligence, 118*, 245–275.

Pujari, A. K. (2001). *Data mining techniques*. University Press.

Saglam, B., Salman, F. S., Sayin, S., & Türkay, M. (2006). A mixed-integer programming approach to the clustering problem with an application in customer segmentation. *European Journal of Operational Research, 173*, 866–879.

Schafer, J., Konstan, J., & Riedl, J. (2001). E-commerce recommendation applications. *Data Mining and Knowledge Discovery, 5*, 115–153.

Thabtah, F., Cowling, P., & Hammoud, S. (2006). Improving rule sorting, predictive accuracy and training time in associative classification. *Expert Systems with Applications, 31*, 414–426.

Tsay, Y. J., & Chiang, J. Y. (2005). CBAR: An efficient method for mining association rules. *Knowledge-Based Systems, 18*, 99–105.

Tseng, M. C., & Lin, W. Y. (2007). Efficient mining of generalized association rules with non-uniform minimum support. *Data & Knowledge Engineering, 62*, 41–64.

Wang, Y., Lim, E. P., & Hwang, S. Y. (2006). Efficient mining of group patterns from user movement data. *Data & Knowledge Engineering, 57*, 240–282.