Validation Model for Discovered Web User Navigation Patterns

Mieczysław Owoc[1] and Paweł Weichbroth[2]

[1] Wroclaw University of Economics, Department of Artificial Intelligence Systems, Poland
mieczyslaw.owoc@ue.wroc.pl
[2] University of Economics in Katowice, Department of Knowledge Engineering, Poland
pawel1739@gmail.com

Abstract. Information society with huge number of everyday acting participants becomes valuable source of surfer behavior research. Especially web server logs can be used for discovering knowledge useful in different areas. Knowledge acquired from web server logs in order to generate solutions has to be validated. The aim of this paper is presentation of web usage mining as a rather complex process and in this context elaboration of validation model. On a basis of iterative and hybrid approach to discover user navigation patterns the concept of generated knowledge validation model is proposed. Some experiments on real website allow to define a new method of generated association rules refinement including specific knowledge validation techniques. Some constraints as discovered knowledge validation criteria are defined.

1 Introduction

Evolution of designing and developing Web sites from static to dynamic approach has enabled easy updates. Furthermore, intensive development and proliferation of WWW network resulted in other new modeling methods. It's obvious - being recognized and visited in the Web means that the content is up-to-date and satisfies its visitors. Widespread scope of content topics shared and presented on the Web site affects the size and depth level of its structures. This results in negative impression of presented content and weaker usability.

Usability of delivered information and knowledge depends on dynamically created user profiles as a result of Web mining and particularly user navigation patterns discovery process. Knowledge acquired from web server log files in order to generate navigation patterns embraces useful as well as non-relevant association rules and has to be validated. Therefore the ultimate goal of this research is presentation of the method allowing for generated knowledge refinement.

These are very specific features of Web Mining procedures: temporal and massive data input, big differentiation of user types. They have direct impact on generated knowledge about website content and forms and in turn should be considered in Knowledge Validation (KV) framework. List of constrains useful in knowledge validation processes includes: page-view duration, total session time, include and exclude items, support and confidence and time and date. All the mentioned constraints should

E. Mercier-Laurent and D. Boulanger (Eds.): AI4KM 2012, IFIP AICT 422, pp. 38–52, 2014.

be considered as potential criteria of discovered knowledge about web surfer navigation patterns. On the other hand the considered constraints are useful in search space reduction in a case of frequent files. Actually the application supporting the discussed problem with mentioned above constraints has been implemented.

The paper is structured as follows. After presentation of related work, crucial definitions essential for this survey are proposed. Results of prepared experiments consisting of relationships between components are demonstrated in the next section. Crucial procedure for formulating knowledge discovery in such environment is investigated later. The paper ends with itemizing of conclusions and considering future research.

2 Related Works

Recommendation systems help to address information overload by using discovered web users navigation patterns knowledge gained from web server log files. A problem for association rule recommendation systems (RS) is placed in dataset. It is often sparse because for any given user visit or object rank, it is difficult to find a sufficient number of common items in multiple user profiles. As a consequence, a RPS system has difficulty to generate recommendations, especially in collaborative filtering applications.

In [1] to solve above problem, some standard dimensionality reduction techniques were applied due to improved performance. This paper presents two different experiments. Sarwar et al. have explored one technology called Singular Value Decomposition (SVD) to reduce the dimensionality of recommender system databases. The second experiment compares the effectiveness of the two recommender systems at predicting *top-n* lists, based on a real-life customer purchase database from an *e*-Commerce site. Finally, results suggest that SVD can meet many of the challenges of recommender systems, under certain conditions.

Ad hoc exclusion of some potential useful items can be one of known deficiencies of this and other reduction of dimensions solutions hence they will not appear in the final results. Two solutions that address this problem were proposed by Fu et al. in [2]. The first solution assumes to rank all the discovered rules based on the degree of intersection between the left-hand side (antecedent) and active session of the user. Then, the *SurfLen* (client-server system) generates the top *k* recommendations. In addition to deal with sparse datasets - if users browsing histories intersect rarely, the system is unable to produce recommendations - the algorithm for ranking association rules was presented. The second solution takes advantage of collaborative filtering. The system is able to find “close neighbors” who represent similar interest to an active user. Then, based on that, a list of recommendations is generated.

Many collaborative filtering systems have few user ranks (opinions) compared to the large number of available documents. In this paper [3], Sarwar et al. define and implement a integration model for content-based ranks into a collaborative filtering. In addition, metrics for assessing the effectiveness filter bots and a system were identified and evaluated.

In the paper [4], Lin et al. a collaborative recommendation system based on association rules framework was proposed. The framework provides two measures for evaluating the association expressed by a rule: confidence and support. Moreover, the system generates association rules among users as well as among items.

3 Definitions

The web site space can be considered as universe U which consists of a sequential sets (P_i) where i=1...M. Each of sets P_i corresponds to unique user session where as each element of P_i is user's request for single page shared by a web site. We consider only such subsets A of P_i ($A \subseteq P_i \subseteq U$) which appeared often enough. The frequency of subset A is defined as *support* (or support ratio) denoted as $support(A)=|\{i\ ;\ A \subseteq P_i\}| / M$. A "frequent" set is a set which support satisfies the minimum support value, denoted as *minsupport*. It is a user dependent value, often defined as cut-off as well.

We developed and applied an Apriori-like algorithm to mine frequent itemsets that is based on level-wise search. It scans a database recursively – a growing collections A are generated using smaller collections, especially the subsets of A of cardinality $|A|$-1, called hipersubsets. Formally, a set B is a hipersubset of a set A if and only if $\exists_{a \in A}\ A=B \cup \{a\}$.

An *association rule* is an implication of the form $A \rightarrow B$, where $B \subseteq P_i \subseteq U$ and $A \cap B = \emptyset$. The support of a rule is the percentage sessions in P that contains $A \cup B$. It can be seen as an estimate of the probability $Pr(A \cup B)$. The rule support is computed as follows: $support(A \rightarrow B) = support(A \cup B) / M$. The confidence of a rule $A \rightarrow B$ is the percentage of sessions in P that contain A also contain B. It can be seen as an estimate of the conditional probability $Pr(B \backslash A)$. The rule confidence is computed as follows: $confidence(A \rightarrow B) = support(A \cup B) / support(A)$. Confidence of the disjoint couple of sets $\{A,B\}$ can be read B under the condition A. If its value exceeds an arbitrarily determined level of minimum confidence (also defined by user), denoted as minconfidence, the pair $\{A,B\}$ will express an association rule $A \rightarrow B$.

Given a session data set D, the problem of mining association rules is to discover *relevant* (or *strong*) association rules in D which satisfy support and confidence greater than (or equal) to the user-specified minimum support and minimum confidence. Here, the keyword is "relevant" (or "strong") which means, taking into account a user's point of view, association rule mining process is complete.

Theorem 1: *Each subset of the frequent set is a frequent set.*

Proof 1: Let A be an arbitrary frequent set and $B \subseteq A$ be a subset of A. The set A is frequent, thus $support(A)=|\{i\ ;\ A \subseteq P_i\}| / M \geq$ minsupport. Since B is a subset of A, we have that $A \subseteq P_i \Rightarrow B \subseteq P_i$, thus $support(B) = |\{i\ ;\ B \subseteq P_i\}| / M \geq |\{i\ ;\ A \subseteq P_i\}| / M \geq minsupport$. Therefore the set B is also frequent which ends the proof. □

Theorem 2: *If $A \rightarrow B \cup C$ is a relevant association rule (A, B, C – pair wise distinct), then also $A \rightarrow B$, $A \rightarrow C$, $A \cup B \rightarrow C$ and $A \cup C \rightarrow B$ are relevant association rules.*

Proof 2: Let $A \rightarrow B \cup C$ be an user- relevant association rule. Then *minconfidence* ≤ *confidence*(A,B∪C) = *support*(A∪B∪C) / *support*(A). □

Let us consider $A \cup B \subseteq A \cup B \cup C$. Having in mind theorem 1, we get *support*(A) ≥ *support*(A∪B) ≥ *support*(A∪B∪C). As a result, *confidence*(A,B) = *support*(A∪B) / *support*(A) ≥ *support* {A∪B∪C} / *support*(A) ≥ *minconfidence*. The set {$A \cup B$} is frequent, so $A \rightarrow B$ is a relevant association rule. On the other hand, *confidence*(A∪B,C) = *support*(A∪B∪C) /*support*(A∪B) ≥ *support* {A∪B∪C}/*support*(A) ≥ *minconfidence*, because of a frequency of a set A∪B∪C, A∪B→C is also a relevant association rule. Consequently, proofs for rules $A \rightarrow C$ and $A \cup C \rightarrow B$ are similar.

Two additional interpretation of introduced earlier constraints should be presented: page view duration and total session time. *Page view duration* is calculated as a difference between the page view start time of the next page minus the page view start of the previous page. *Total session time* is calculated as a sum of duration times referencing to particular elements of frequent sets without the last element.

4 Phases of Web Usage Mining Process

Tools commonly used in data mining are applied in Web Internet resources analysis. The main fundamental task is pattern usage discovering available in the Internet. Such knowledge can be the basics of recommendation systems in order to improve functionality (cross-selling) and usability (easier access) to Internet portals and services. First holistic and integrated (incl. different KDD steps) model of Web Usage Mining process was proposed by Cooley et al. in [5]. Afterwards, Srivastava et al. in [6] extended this model and apparently distinguished three phases: (*1*) preprocessing, (*2*) pattern discovery and (*3*) pattern analysis. This model has been widely applied – the references can be found e.g. in [7-14].

This model is complex and practice- oriented solution of solving problem of knowledge discovery from Web data repositories. The following particular tasks are subordinated to the mentioned phases. It allows for clear defining of user requirements from its point of view including data format and expected analysis results. However it seems to be reasonable to extend the model by data collection (see Fig. 1). During this phase data sources are defined based on data entry type and the same selection of necessary variables necessary variables including planned analysis are performed.

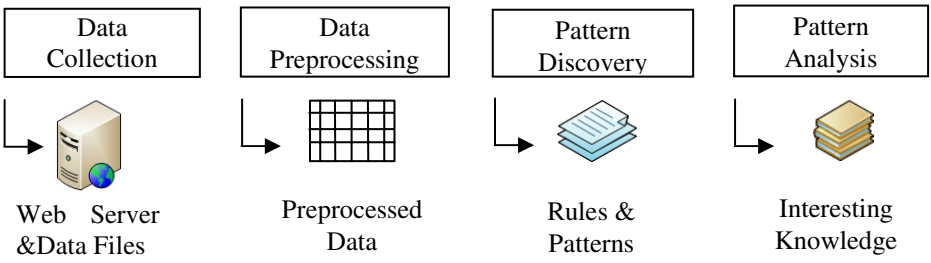

Fig. 1. Extended Web Usage Mining process

Two dimensions can be defined in this model: e.g.:

- phase – definition of data-oriented actions (tasks),
- object – definition of data types.

The whole process consists of the four phases; first three can be considered as machine-aided while the last one is human - expert oriented. In this frames, similarly to CRISP-DM, specialized tasks are separate. We can distinguish four phases in web usage mining process:

1. Data collection, usage data can be collected from various sources like Web Server (e.g. log files), Client side (e.g. cookie files) or proxy servers.
2. Data preprocessing, this phase concerns raw data files which often includes web server log files. There can be distinguish following tasks: (a) data cleaning and filtering, (b) de-spidering, (c) user identification, (d) session identification and (e) path completion.
3. Pattern discovery, it can be seen as the data mining step when comparing to KDD. The following methods and techniques can be applied: (a) statistical techniques, (b) association rules, (c) sequential patterns, (d) clustering and (e) classification.
4. Pattern analysis, where domain expert evaluates discovered knowledge and optionally does performance assessment. If necessary, some modeling methods and techniques can be applied, such as clustering and classification.

In the next section, we refer to each phase briefly, emphasizing the most important issues which can be seen as a part of knowledge validation challenge.

5 Experiments on Real Web Sources

In Web Usage Mining the major sources of data are Web Server and application server log files. We only used Web Server log files which are a set of Web users' activity record, registered by onet.pl - one of the most recognized Web portals in Poland. Web Server log files contain full history of access requests to files, shared on the Web Server. Usually, http server vendors apply Common Log Format (CLF) to the log files associated with http protocol service. This format was developed by CERN and NCSA as the component of http protocol. According to this format, there are seven variables which were selected to be recorded. Complete description of CLF format and its variables can be found e.g. in [15].

5.1 Data Collection

The data format of Web Server log files is specific in onet.pl (see Tab. 1) and includes seven variables. Sample entries, which represent Web users' activity, are presented below.

Table 1. The structure of raw Web Server log files

	Variable					
Row	◊	♠	○	□	⌂	♣
1	140229	8654	2166	2	5723	724
2	140229	8654	2166	2	5723	725
3	140229	8655	2227	124	5086	8052
4	140229	8641	2310	26	1587	1007
5	140229	8654	2227	124	5086	8151

◊ time of the session, ♠ session ID, ○ user ID, □ service ID, ⌂ subservice ID, ♣ html file ID.

In our research, we used a web server log file which covers five hours of system activity - from 2PM to 7PM on 29th December 2008. We also want to notice the fact that dedicated solutions (e.g. scripts, services) on the Web Server side were implemented. They were intended to reduce globally useless data (e.g. spiders activity). In this case, we can determine that in the phase of data collection, some preprocessing tasks took(or have taken) place.

5.2 Data Preprocessing

We used Linux operating system to operate on the raw data. To carry out this phase, a simple script in awk language was implemented to process the log files. Firstly, from six available variables just two were selected (♠ ♣) and separated from others. Secondly, these two were sorted accordingly to the session time (◊) and session identifier (♠). Thirdly, the sessions, where only one html page was requested by the users, were deleted from the log files. In such way, there were 2.132.581 unique sessions observed which would be taken into account in our experiments. The size of the log file was reduced from 4.018.853 KB to 512.934 KB. Again, it can be noticed that necessary tasks which typically had to be performed in the second phase were very limited. In addition, the source data was supplemented with text files which contained dictionaries corresponding to the URL name (e.g. 724 denotes to [www]). Finally, the processed file had the following format (session id, URL name):

- 8654 [www]
- 8654 [sport.html]
- 8654 [games.html]

5.3 Pattern Discovery

In this phase, frequent Web users' access patterns are discovered from sessions. A few algorithms have been proposed to discover frequent itemsets. During the literature study of KDD, we came across the algorithms such as AprioriAll [16], GSP [17], SPADE [18], PrefixSpan [19], ItemsetCode [20].

We have decided to use Apriori algorithm, although it is not the most efficient and recent one. It is suitable solution for our problem since we can make it more efficient by pruning most of the candidate sequences generated in each iteration. This can be

done because of given constraints and requirements. We cannot assume that for every subsequent pair of pages in a sequence the former one must have a hyperlink to the latter one. The justification of that assumption is the fact that we do not have full access to the Web Server repository. Some files could have been deleted, moved to the archives or simply changed by the provider. On the other hand, the frequency of content update can be seen as very high - we have to keep in mind that approximately 7 million users per day visit this Web portal. So, it is obvious that the content hosted on the portal must be up-to-date. Furthermore and what seems to be the most important, the engine of recommendation system is able to generate links directly on web pages and does not require knowledge of inner-oriented structure and relationships of individual web pages.

Let $P = \{p_1, p_2, \ldots, p_m\}$ be a set of web pages, called items. Database D represents a set of reconstructed sessions $S = \{s_1, s_2, \ldots, s_n\}$. Each session s_n is a subset of P

where $s_n \subseteq P$, called itemset.

We implemented Apriori algorithm (Algorithm 1) where input is a database of sessions D and output is a set of frequent sequences L.

```
(1) L_1 = find_frequent_1-itemsets(D);
(2) for (k = 2; L_{k-1} = ∅; k++) {
(3)   C_k = apriori_gen(L_{k-1});
(4)   for each transaction T ∈ D
(5)      C_t = subset(C_k, t);
(6)     for each candidate c ∈ C_t
(7)     c.count++; }
(8)   L_k = {c ∈ C_k | c.count ≥ min_sup} }
(9)   return L = ∪_k L_k
```

```
procedure apriori_gen(L_{k-1} : frequent(k - 1) -itemsets
(1) for each itemset l_1 ∈ L_{k-1} and
(2) for each itemset l_2 ∈ L_{k-1}
(3)   if(l_1[1] = l_2[1]) ∧ (l_1[2] = l_2[2]) ∧ … ∧ (l_1[k - 2] =
l_2[k - 2]) ∧ (l_1[k - 1] < l_2[k - 1])
(4)   then {
(5)     c = l_1 ⊠ l_2;
(6)     if has_infrequent_subset(c, L_{k-1}) then
(7)       delete c;
(8)     else add c to C_k; }
(9) return C_k;
```

```
procedure has_infrequent_subset(c: candidate k- itemset);
L_{k-1}: frequent(k - 1)- itemsets;
(1)   for each(k - 1)-subset s ∈ c
(2)     if s ∉ L_{k-1} then
(3)       return True else
(4)       return False;
```

As a result the program returns frequent sequences with support for each. A simple example of a three element sequence is given below.

[www];[info]/science/item.html;[info]/world/item.html;0,02011

Interpretation of this particular sequence may be expressed in these words: "*Over 2 percent of anonymous users between 2PM and 7PM on 29th December 2008, first requested access to the home page, next opened science section then world section*".

Based on the set of frequent sequences L we are able to generate sequential association rules (SAR). In this scenario, a simple algorithm was developed (Algorithm 2). Let $R = \{ r_1, r_2, \ldots , r_m \}$ be a set of SAR. In each rule antecedent (body of the rule, left side) and consequent (head of the rule, right side) must not be replaced. In other words, it guarantees the precise order of each element in the sequence. Also, a user is requested to provide the minimum confidence (*minconfidence*) threshold.

```
(1) R={}
(2) for all I ∈ L do
(3)   R = R ∪ I →{}
(4)   C1 = {{i} | i ∈ I};
(5)   k:= 1;
(6)   while Ck ≠ {} do
(7)     //extract all consequents of confident association
rules
(8)     Hk := {X ∈ Ck | confidence(I \ X ⇒ X, D) ≥
min_conf}
(9)     //generate new candidate consequents
(10)    for all X, Y ∈ Hk, X[i] = Y[i] for 1 ≤ i ≤ k - 1
and X[k] < Y[k] do
(11)      I = X ∪ {Y[k]}
(12)      if ∀J ⊂ I, |J| = k : J ∈ Hk then
(13)        Ck+1 = Ck+1 ∪ I
(14)      end if
(15)    end for
(16)    k++
(17)  end while
(18)  //cumulate all association rules
(19)  R := R ∪ {I \ X →X | X ∈ H1 ∪ ⋯ ∪ Hk}
(20) end for
```

The set of sequential association rules *R* is achieved as a result. We give a simple example of such results, based on previously given frequent sequence.

r_1:{www};{science.html}⟶{world.html}; 0,02011; 0,964

r_2:{www}⟶{science.html}{world.html};
0,02011; 0,652

r_3:{www}⟶{science.html}; 0,0305; 0,00209

r_4:{www}⟶{world.html}; 0,0294; 0,00152

Interpretation of the first rule may be expressed in these words: "*There is a 96,4% chance that a user who visited {www} and {science.html} pages, after them would also visit {world.html}*". In other words, the value of confidence ratio shows degree of rule reliability.

In this way we have shown that attractiveness of discovered knowledge is evaluated by two measures: support and confidence ratio. They are both specified by a user - it guarantees a definitive result.

5.4 Results of Pattern Analysis

The aim of the research was to discover frequent sequences which represent web user navigation paths. In first iteration the level of support and confidence ration was respectively 0,01 and 0,9. Table 2 shows top ten (taking into account the highest support, denoted by percentage) one element frequent itemsets.

Table 2. Itemsets and its support

Itemset	Support	Itemset	Support
www	78.48%	email/logout.html	14,09%
email/login.html	27.08%	email/folder/delete.html	10,80%
email/folder.html	25.49%	sport/football/news.html	10,22%
info/world/item.html	20.27%	sport/formula one/news.html	9,88%
email/folder/open.html	15,01%	info/homeland/item.html	9,18%

For instance, human interpretation of such knowledge might be expressed in these words: "Over 27 percent of sessions include a page enabling user to log on the email service".

Entire volume of frequent sequences draws picture of the most popular content of the web portal. First conclusion which arises from this analysis to divide and group discovered knowledge to two different categories: (1) service- oriented and (2) content- oriented. First category relates to the hosted services via the web portal like email, advertisements, auctions, games, video on demand, dates. Second category relates to the shared content like information, news and plots. In this kind of the web portal, we are not able to recommend anything on the home page unless we violate user's privacy. On the other hand, keeping in mind high level of usability and visibility, the objects' arrangement should remain static. Therefore service- oriented content will not be considered to be recommended. Also, over 60% of discovered knowledge concerns users' actions while using email service. As an example let us deliberate on sequence below:

- [www];[email]/login.html;[email]/inbox.html;[email]/new-message.html;[email]/logout.html

Such knowledge is useless for recommendation engine since it simply presents obvious and feasible actions during common session, restricted to single service. In this

case, we decided to decrease the confidence ratio to four additional levels: 0,8; 0,7; 0,6 and 0,5. Table 3 shows the volume of discovered sequential association rules for five different levels of confidence ratio.

Table 3. The volume of SAR

	Number of SAR				
	Confidence				
Number of items	◊	□	○	θ	⌂
2	65	79	95	118	142
3	169	197	254	349	430
4	196	225	315	480	585
5	114	132	209	328	407
6	28	32	68	111	146
total	**572**	**665**	**941**	**1386**	**1710**

minimum confidence: ◊ 0.9 □ 0.8 ○ 0.7 θ 0.6 ⌂ 0.5

It can be noticed easily that simple and apparent indication has occurred - on every lower level of confidence the number of rules have increased. Finally, for 0,5 confidence more than 1710 rules were discovered. These 1138 rules (difference between 0,5 and 0,9 confidence) are not likely to bring further knowledge of web data usage. Nevertheless, some interesting itemset groups were observed which will be added to knowledge base. An example of useful sequence is presented below:

- [www];[sport/football/news.html];[sport/formula one/news.html];[info/homeland/item.html]

At this point, previously undiscovered knowledge shall be reviewed and compared to that, which is already stored in knowledge base. Moreover, if discovered knowledge was transferred to knowledge base and its resources are engaged by recommendation engine, we are able to track the process of knowledge validation. It means that two measures recall and precision will determine degree of knowledge adequacy. In other words, we are able to determine quality of recommendations produced by the system.

Another step, which is possible to undertake and promises to discover new rules, is to decrease the value of the minimum support. There is no general instruction with suggested value in any scenario. It is a subjective issue and relies on expert intuition and experience. In the next two iterations, we set up the value of minimum support respectively on 0,05 and 0,001. In our opinion it should be compromise between time-consumption and hardware requirements constraints.

Table 4. Frequent sequences for three different minimum support values

	Frequent sequences				
	Quantity			Change (◊=100)	
Number of items	◊	□	○	□	○
1	64	103	276	61%	331%
2	167	307	1522	84%	811%
3	177	438	3279	147%	1753%
4	418	338	3768	-19%	801%
5	40	154	2625	285%	6463%
6	14	44	1193	214%	8421%
7	0	6	373	-	-
8	0	0	73	-	-
9	0	0	5	-	-
total	**880**	**1390**	**13114**	**58%**	**1390%**

minimum support: ◊ 0.01 □ 0.005 ○ 0.001

The program was executed under Eclipse environment with Java version 1.6.0 on IBM PC x86 computer with an Intel Core2 Quad processor 2.4 GHz and 2 GB Random Access Memory. Execution times are 55 minutes, 2 hours 30 minutes and 28 hours 16 minutes respectively to lower value of the minimum support. The determination of the factors influencing on the effectiveness of algorithm covers an remarkable research question. Unfortunately, we are not able to put forward straightforward answers.

Let us examine new sequences discovered throughout these two independent iterations. Though, we only focus on the set (○) because it is a superset of the other two sets (◊ and □). First of all, we can notice a enormous growth - almost 14 times larger set of frequent sequences was discovered, when value of minimum support was set up on 0,001 comparing to 0,01. This fact is worth deep consideration however our attention would be focused on the longest nine- items sequences which one of them is presented below:

- [www];
- [dates]/visitors.html; [dates]/email.html; [dates]/email/new.html; [dates]/index.html; [dates]/user-page/index.html; [dates]/user-page/index/k.html; [dates]/user-page/album.html; [dates]/album.html

We can definitely state that the results are unsatisfactory. The longest sequences present the interactions within one hosted service. The same situation can be observed taking into account eight- and seven- items sequences. Just six sequences in the latter one would possibly classified to be added to the knowledge base. These sequences present interesting navigation paths like:

- [www];
- [business]/pap.html; [business]/news.html; [business]/company/market.html;
- [info]/homeland/item.html; [info]/world/item.html; [business]/stock market/news.html

We asked ourselves: was it necessary to decrease value of minimum support having in mind program time-consumption? Even some portion of discovered knowledge might be useful, there is a great risk that it would become inadequate. This situation happens when the content update is very often. Discovered sequences are not longer available to reach because its items (represented by links) are simply replaced by others.

In our approach we anticipated the last step which can help to discover relevant knowledge from web server log files. The assumption is simple: the domain expert shall determine service- oriented itemsets and any other irrelevant items and exclude them from the data. Then, the data is preprocessed again and process of web usage mining starts one more. Preliminary experiments have confirmed presented model in the next section.

6 A Concept of a Validation Procedure

In this section, we present an iterative model for discovering and validating web user navigation patterns. General assumptions of this procedure are as follows:

- Input data are the results of the previous described steps (data are collected then preprocessed to allow for discovering patterns).
- The procedure consists of four steps where defined confidence and support parameters are changed in order to exclude irrelevant sets.
- A domain expert is responsible for the validation generated knowledge base. He plays crucial role in order to exclude irrelevant sets – furthermore called also as an user.
- The procedure is iterative what denotes active participant of the expert on different steps of validation with possibility of starting the procedure from the beginning.

The graphical form of the procedure is presented in Figure 2.

In the first step, initial values for minimum support and confidence are set by a user. Next, the pattern discovery process is performed and thus analyzed and evaluated by a user. Positively assessed knowledge is put into the knowledge base from where inference engine produces a list of recommendations to active user sessions. The effectiveness of this process can be measured by two metrics: recall and precision. Therefore the knowledge base is validated by them, expressing the degree of its adequacy. If the values of metrics are not satisfied, in the second step the confidence value is decreased.

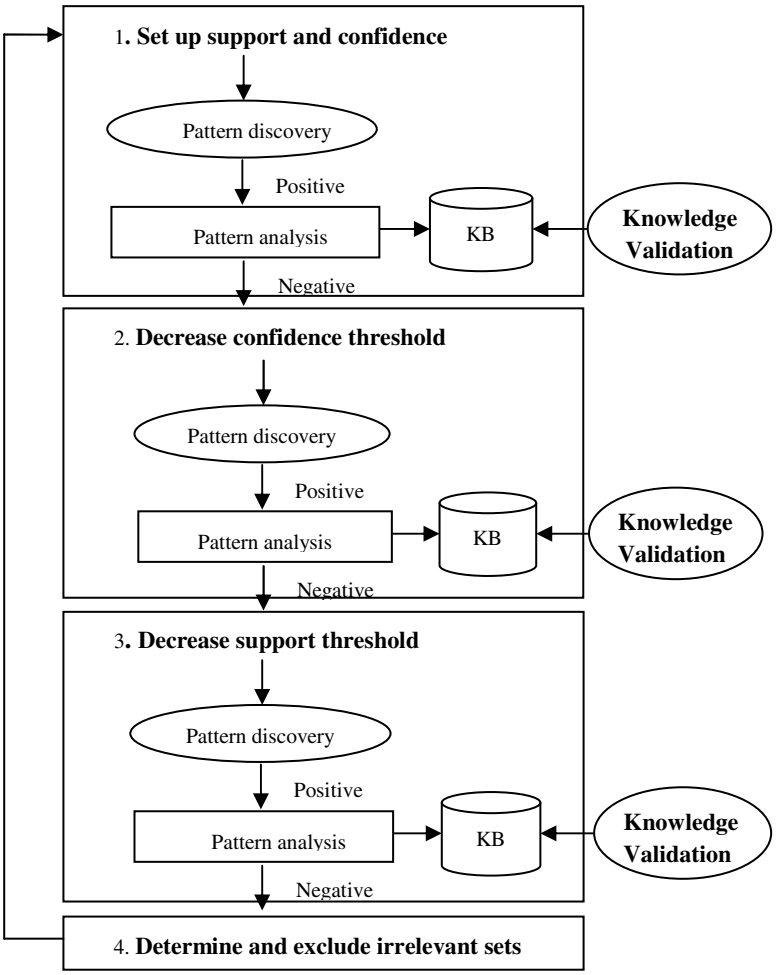

Fig. 2. Iterative model for discovering and validating web user navigation patterns

Similarly, in the third step, a user can decrease the minimum support value. If even in this case, values of evaluation metrics are not satisfied, a user specifies additional constraint by excluding sets which are irrelevant. If results are not satisfactory, the process can be started from the beginning.

7 Conclusions

In this paper, we introduced an interactive a user-driven model which addresses problem of validating web user navigation patterns. Experiments performed on web server file logs show that the model is useful and in some circumstances can be used in large-scale web site environments.

Conclusions based on our presented results can be divided into two groups. In the first group we have the confirmation of the general properties of the method of association rules, and detailed observations of Agrawal and Srikant algorithm and its implementation in the form of application.

Thanks to the cut-off at the level of support we discovered only the strong association rules. The number and size of the discovered association rules increase with the growth of the minimal support ratio. Cutting off the irrelevant sets of frequent factor support was implemented to be done at first, so that level has decisive influence on the uptime. Confidence factor, which determines the level at which the second cut-off is made after pruning of low-level support sets, has a significant impact on the final number of the rules, but the change of this factor does not affect the uptime. Number of the elements of the association rules determines the strength of correlation (the more elements in the antecedent of a rule, the correlation is weaker, the more elements in the consequent of a rule, the correlation is stronger). This, in addition to the actual values of support and confidence, has a decisive influence on ordering rules in the final results.

The second group of the conclusions concerns the study – particular data that have been examined and usage of a multistage pruning. The primary study revealed structure of the portal. In particular, there were extracted two independent services which have extremely strong, static navigation paths. These services have their own audience and introduce artifacts which hinders the proper analysis. Furthermore, the hierarchical structure of information service is reflected in the extracted association rules.

Despite the relatively slight reduction in the size of data, allows to significantly reduce the parameters in the secondary study and obtain more relevant knowledge. In summary, the multistage pruning seems to be accurate for use in the analysis that uses method of association rules. In addition, the method itself seems to give satisfactory results in discovering web users' navigation paths.

References

1. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Application of Dimensionality Reduction in Recommender System - A Case Study. In: ACM WEBKDD Workshop. ACM (2000)
2. Fu, X., Budzik, J., Hammond, K.J.: Mining navigation history for recommendation. In: Proceedings of the 5th International Conference on Intelligent User Interfaces, pp. 106–112. ACM, New Orleans (2000)
3. Sarwar, B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B., Riedl, J.: Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. In: Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work, pp. 345–354. ACM, Seattle (1998)
4. Lin, W., Alvarez, S.A., Ruiz, C.: Efficient Adaptive-Support Association Rule Mining for Recommender Systems. Data Mining and Knowledge Discovery 6, 83–105 (2002)
5. Cooley, R., Mobasher, B., Srivastava, J.: Web Mining: Information and Pattern Discovery on the World Wide Web. In: Proceedings of the 9th International Conference on Tools with Artificial Intelligence. IEEE Computer Society (1997)

6. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.N.: Web usage mining: discovery and applications of usage patterns from web data. ACM SIGKDD Explorations Newsletter 1, 12–23 (2000)
7. Kosala, R., Blockel, H.: Web mining research: A survey. Newsletter of the Special Interest Group (SIG) on Knowledge Discovery and Data Mininig SIGKDD: GKDD Explorations 1 (2000)
8. Eirinaki, M., Vazirgiannis, M.: Web mining for web personalization. ACM Trans. Internet Technology 3, 1–27 (2003)
9. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. Data Mining and Knowledge Discovery 6, 61–82 (2002)
10. Pierrakos, D., Paliouras, G., Papatheodorou, C., Spyropoulos, C.D.: Web Usage Mining as a Tool for Personalization: A Survey. User Modeling and User-Adapted Interaction 13, 311–372 (2003)
11. Facca, F.M., Lanzi, P.L.: Mining interesting knowledge from weblogs: a survey. Data Knowledge Engineering 53, 225–241 (2005)
12. Bayir, M.A., Toroslu, I.H., Cosar, A., Fidan, G.: Smart Miner: a new framework for mining large scale web usage data. In: Proceedings of the 18th International Conference on World Wide Web, pp. 161–170. ACM, Madrid (2009)
13. Staś, T.: Wykorzystanie algorytmów mrowiskowych w procesie doskonalenia portali korporacyjnych. Rozprawa doktorska. Akademia Ekonomiczna im. Karola Adamieckiego w Katowicach, Katowice (2008) (in Polish): Staś, T.: The Usage of Ant Colony in the Improvement of corporate portals process. Ph.D. Dissertation. University of Economics in Katowice, Katowice (2008)
14. Markellou, P., Rigou, M., Sirmakessis, S.: Mining for Web Personalization. In: Scime, A. (ed.) Web Mininig: Applications and Techniques, pp. 27–49. Idea Group Reference, USA (2005)
15. `http://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format`
16. Agrawal, R., Srikant, R.: Mining Sequential Patterns. In: Proceedings of the Eleventh International Conference on Data Engineering, pp. 3–14. IEEE Computer Society (1995)
17. Srikant, R., Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements. In: Apers, P.M.G., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 3–17. Springer, Heidelberg (1996)
18. Zaki, M.J.: SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning 42, 31–60 (2001)
19. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.-C.: Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach. IEEE Transactions on Knowledge and Data Engineering 16, 1424–1440 (2004)
20. Ivancsy, R., Vajk, I.: Frequent pattern mining in web log data. Acta Polytechnica Hungarica 3, 77–90 (2006)