Research Paper Computer Science

Comparison of Data Mining Algorithms for Enhancing The Web Usage Mining

R.SUBA LAKSHMI | Dept. of Computer Science, P.S.G College of Arts & Science, Coimbatore

ABSTRACT

World Wide Web is the large source of online data, which includes text, images, videos, audio, etc., and is currently facing explosive growth of information. The information thus present in WWW lacks integrated structure or scheme, which poses serious drawbacks. From the user's perspective, it is very difficult to extract useful knowledge from the huge amount of information and secondly, it is also difficult to extract for the users to access relevant information efficiently. One way to extract such information from WWW is to use Web Usage Mining, which is the process of extracting knowledge from web user's access data by using data mining technologies. The extracted information can be used by many applications like personalization, website redesign or site improvement, business intelligence, navigation pattern behavior prediction, etc. This research work is focused on the discovery of knowledge about the people using the web, their interest and expectations, using association-based analysis. For this purpose, two frequently used algorithms, namely, Apriori and FP growth algorithm are selected.

KEYWORDS Apriori and FP growth algorithm Apriori, Web usage mining, pattern discovery,association rules, support – confidence framework.

2. INTRODUCTION

Data mining is a multidisciplinary field that was mainly introduced to process large databases and discovers useful information that could help in decision-making (Washio et al., 2008). Web mining, an extension of data mining, employs techniques of data mining to documents on the Internet (Heydari et al., 2009). Web Usage Mining, a part of web mining and data mining, is involved in mining usage characteristics of the users of web applications (Kolari and Joshi, 2004). Frequent pattern mining is the discovery of relationships or correlations between items in a transaction database. This research work considers two types of datasets as transaction database to perform frequent pattern mining is based on association rules (Park et al., 1995; Agrawal et al., 1996). They are market basket data and web log data.A set of market basket transactions is a common dataset used in frequent pattern analysis (Gionis et al., 2007). A web log file is a text file that is automatically created by the server as records of website activity (Chitraa and Davamani, 2010). The present research work uses an IIS format of log file. The present research work also focus on identifying frequent patterns from web log files, which can be used by web masters and business to improve their websites to suit the needs of their customersThe first studied approach in the field of frequent pattern Apriori is the best-known algorithm to mine association rules (Agrawal and Srikant, 1994). It uses a breadth-first search strategy to counting the support of itemsets and uses a candidate generation function which exploits the downward closure property of support. . s. As both the algorithms are the frequently used techniques, they were selected for study in this research work

III - REVIEW OF LITERATURE

The most frequently used basic frequent itemset mining methodologies are Apriori, FP-growth and Eclat. Since there are usually a large number of distinct single items in a typical transaction database, and their combinations may form a very huge number of itemsets, it is challenging to develop scalable methods for mining frequent itemsets in a large transaction database. Agrawal and Srikant (1994) observed an interesting downward closure property, called Apriori, among frequent kitemsets: A k-itemset is frequent only if all of its sub-itemsets are frequent. This implies that frequent itemsets can be mined by first scanning the database to find the frequent 1-itemsets, then using the frequent 1-itemsets to generate candidate frequent 2-itemsets, and check against the database to obtain the frequent 2-itemsets. This process iterates until no more frequent k-itemsets can be generated for some k. This is the essence of the Apriori algorithm and its alternative (Mannila *et al*., 1994).From the point of introduction, there have been extensive studies on the improvements or extensions of Apriori, e.g., hashing technique (Park *et al*., 1995), partitioning technique (Savasere *et al*., 1995), sampling approach (Toivonen, 1996), dynamic itemset counting (Brin *et al*., 1997), incremental mining (Cheung *et al*., 1996), parallel and distributed mining (Agrawal and Shafer, 1996; Zaki *et al*., 1997), and integrating mining with relational database systems (Sarawagi *et al*., 1998). Geerts *et al*. (2001) derived a tight upper bound of the number of candidate patterns that can be generated in the level-wise mining approach. This result is effective at reducing the number of database scans.In many cases, the Apriori algorithm significantly reduces the size of candidate sets using the Apriori principle. However, it can suffer from two-nontrivial costs: (1) generating a huge number of candidate sets, and (2) repeatedly scanning the database and checking the candidates by pattern matching. Han *et al*. (2000) devised an FP-growth method that mines the complete set of frequent itemsets without candidate generation.FP-growth works in a divide-and-conquer way. The first scan of the database derives a list of frequent items in which items are ordered by frequency descending order. According to the frequency-descending list, the database is compressed into a frequent-pattern tree, or FP-tree, which retains the itemset association information. The FP-tree is mined by starting from each frequent length-1 pattern (as an initial suffix pattern), constructing its conditional pattern base (a "subdatabase", which consists of the set of prefix paths in the FP-tree co-occurring with the suffix pattern), then constructing its conditional FP-tree, and performing mining recursively on such a tree.The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-tree.The FP-growth algorithm transforms the problem of finding long frequent patterns to searching for shorter ones recursively and then concatenating the suffix. It uses the least frequent items as a suffix, offering good selectivity. Performance studies demonstrate that the method substantially reduces search time. This research work conducted in the field of data mining, in particular, regarding frequent pattern mining, two algorithms, Apriori and FP-Growth are selected for this purpose.

IV. METHODOLOGY

The amount of data stored in databases has increased tremen-

dously with the widespread use of databases and the rapid adoption of information systems and data warehousing technologies. The process of obtaining this information is called Frequent Pattern Mining and can be discovered using various data mining techniques, like clustering, classification, and prediction and association analysis. Association rules are repeatedly used for identifying frequent patterns from transaction database. Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items. Let 'D' be a set of transactions, where each transaction 'T' is a set of items such that $T \subseteq I$. An association rule is an implication of the form, $X \rightarrow Y$, where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ holds in the transaction set *T* with confidence 'c', if c% of transactions in T that support 'X' also support 'Y'. The rule has support's' in *T* if s% of the transactions in *T* contains $X \cup Y$.

Given a set of transactions *D* (the database), the problem of mining association rules is to discover all association rules that have support and confidence greater than the user-specified minimum support (called *minsup*) and minimum confidence (called *minconf*). An association rule 'r' is a relation between itemsets of the form

$r: X \Rightarrow (Y - X)$

where X and Y are frequent itemsets and $X \subset Y$. The itemsets X and (Y – X) are called, respectively, antecedent and consequence of the rule 'r'. The valid association rules are those for which the measure of support and confidence is greater than or equal to the minimal thresholds of support and confidence, called minsup and minconf (Webb, 2000). Support and confidence are calculated as in Equation (1.1) and (1.2).

$$\text{Support}(X) = \frac{|\{t \in D \mid X \subseteq t\}|}{|D|} \qquad (1.1)$$

$$\text{Confidence}(r) = \frac{\text{Support}(Y - X)}{\text{Support}(X)} \qquad (1.2)$$

A typical association rule is of the form

$A \Rightarrow B, C$ [Support = 60%, Confidence = 80%]

"50% of visitors who accessed URLs B and C also visited A"

The process of finding all the association rules with confidence and support above the respective thresholds is a two stage process. The first stage consists in finding all sets of items with support above the support threshold; these sets are called large item sets. The second step consists in computing for each large itemset the confidence for all its expressions with the form of a rule; the expressions whose confidence is above the confidence threshold are the rules. The second step of the process is trivial; therefore, the problem of mining association rules can be viewed as the problem of finding all the itemsets with support above a given threshold. The algorithms are analyzed based on memory usage, time and scalability.. The most well known algorithms to perform such task are the Apriori algorithm and FP Growth, which are described by Agrawal *et al*. (1996). In this research work two solutions are compared for this purpose. The first is to use a Apriori algorithm and the second is to use FP-Growth algorithm.
Two types of datasets are selected to analyze the applicability of Apriori and FP-Growth for identify usage behaviour. They are (i) transactional database and (ii) web log data. The transaction database consists of customer buying details and the web log data consists of user navigation details. Preprocessing is a very crucial and challenging step as its result has a direct impact on the rules and patterns generated by mining algorithms are shown in the Figure 1. The purpose of preprocessing is to transit various input such as content, structure and usage information into the format which data mining algorithms can handle easily (Han *et al*., 2000).

1. Perform cleaning and filtering
   a. Remove all image, video, audio entries
   b. Remove all unsuccessful HTTP entry codes (Status code 200)
   c. Retain only those entries have GET and POST in re quest method field
   d. Remove all entries with blank IP addresses
   e. Remove all entries whose depth level is more than five

2. User Identification and Session Identification
   a. Identify unique uses in a session who have in the website for more than 30 minutes. Each session is assumed to be of duration 30 minutes and a new session starts after each 30 minutes

3. Assign sequential unique codes to each individual URL
4. Reformat the log file, by assigning the URLs in each session with the unique code formulated in Step 5
5. Prepare a URL-Session Binary Table, which is created by assigning a value of 1, if a URL has been visited in a session, else a 0 value is inserted.
6. Calculate the total number of visits in made to each URL, session wise, and construct the final data.

Figure 1: Preprocessing Procedure

Pattern discovery is performed using the two selected association-based algorithms, namely, Apriori and FP-Growth.The apriori algorithm finds the frequent itemsets. Frequent itemsets are the set of items that have minimum support. Apriori algorithm works on the principal that a subset of a frequent itemset must also be a frequent itemset. That is, if {AB} is a frequent itemset, both {A}. The apriori algorithm finds the frequent itemsets. Frequent itemsets are the set of items that have minimum support. Apriori algorithm works on the principal that a subset of a frequent itemset must also be a frequent itemset. That is, if {AB} is a frequent itemset, both {A} and {B} should be a frequent itemset. The frequent itemsets are iteratively found with cardinality from 1 to k (k-itemset). The itemsets thus formed are used to generate the association rules. The apriori pseudo code are shown in the Figure 2.

Join Step: C_k *is generated by joining* L_k *-1with itself*
Prune Step: *Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset*
Pseudo-code
C_k: Candidate itemset of size k
L_k: frequent itemset of size k
L_1= {frequent items};
For (k= 1; L_k!= ∅; k++) do begin
C_{k+1}= candidates generated from L_k;
for each transaction tin database do
increment the count of all candidates in C_{k+1} that are contained in t
L_{k+1}= candidates in $C_{k+1}$with min_support
end
return $\cup_k L_k$;
Figure 2 : Apriori Pseudo Code

The FP-tree construction takes exactly two scans of the transaction database: The first scan collects the set of frequent items, and the second scan constructs the FP-tree. The cost of inserting a transaction Trans into the FP-tree is O(|freq(Trans)|), where freq(Trans) is the set of frequent items in Trans. There are several important properties of FP-tree that can be derived from the FP-tree construction process are shown in the Figure 3. The final stage is the analysis of the patterns discovered in the previous step.

Finally this research work is focused on the discovery of knowledge about the people using the web, their interest and expectations, using association-based analysis. For this purpose, two frequently used algorithms, namely, Apriori and FP growth algorithm are selected. Both the algorithms follow common steps, namely, preprocessing, generation of association rules, pattern discovery and analysis. Data preprocessing involves steps like data cleaning, user identification, session identification and conversion of the resultant

web log dataset into a form that is ready for association rule generation.

5. EXPERIMENTAL RESULTS

Finally The experiments were conducted in two stages, (i) General Pattern Analysis and (ii) Performance analysis in terms of execution time and memory usage. Both the algorithms selected were able to discover access pattern and user behaviors using support and confidence thresholds accurately.

The final stage is the analysis of the patterns discovered in two stages,Validation, to identify relevant rules or patterns, from which interesting patterns can be discovered.Interpretation, Mathematical interpretations which can be used to discover knowledge. Web log data from microsoft elearning website was used during experiments.The server log file was extracted for 15 days, from 01 July 2010 until 15 July 2010 and was retrieved from the www.microsoftelearning.com server using tools like Web log Expert and Web log Explorer. Web log Expert and Web log Explorer are freeware software tools which can be used to download server log files of a website. The e-learning portal site of Microsoft contains various features for teachers and students that are suited for distance or e-learning environment. The selected facilities from the e-learning portal include the following and a sample of the Microsoft e-learning log file are profile creation (Registration),Lessons with download option, Exam assessment,Calendar,Email,Chat,Assignment and Forum. The result of the present study can be used to improve the site design by identifying patterns that frequently used by the users and reorganizing the website to reflect the most commonly used patterns. The results are discussed with preprocessing,pattern Analysis(association rules),Performance Analysis(comparison of Apriori and FP-Growth). Three web log files were used during testing, each with different number of records and varying file size. The number of records and file size are shown in Table 1.

Input : A transaction database DB and a minimum sup port threshold ξ .
Output : FP-tree, the frequent-pattern tree of DB.
Method : The FP-tree is constructed as follows.

1. Scan the transaction database DB once. Collect the set of frequent items F and their supports. Sort F in support descending order as L, the list of frequent items.
2. Create the root of an FP-tree, T, and label it as "root". For each transaction Trans in DB do the following.

a. Select and sort the frequent items in Trans according to the order of L. Let the sorted frequent item list in Trans be [p | P], where p is the first element and P is the remaining list. Call insert_tree([p | P], T).
b. The function insert_tree([p | P], T) is performed as follows. If T has a child N such that N.item-name = p.item-name, then increment N's count by 1; else create a new node N, and let its count be 1, its parent link be linked to T, and its node-link be linked to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert_tree(P, N) recursively.

Figure 3: FP-Tree Construction Procedure

STARTING STATISTICS OF WEB LOG FILES
TABLE 1

Filename	No. of transactions	File size	Period
WebLog1.log	3000	0.8 MB	1 Day
WebLog2.log	25000	2.3 MB	3 Days
WebLog3.log	50000	4.5MB	10 Days

TABLE 2

Step No.	No. of Records		File size (MB)	
	Before	After	Before	After
WEBLOG1.LOG				
1	3000	1200	0.80	0.30
2	1200	800	0.30	0.21
WEBLOG2.LOG				
1	25000	11857	2.3	1.20
2	11857	8018	1.1	0.78
WEBLOG3.LOG				
1	50000	22024	3.5 MB	1.40
2	22024	15934	1.4 MB	1.08

EFFECT OF PREPROCESSING

The results obtained in terms of number of records and file size are shown in Table 2.

STATISTICS OF WEB LOG FILES AFTER USER AND SESSION IDENTIFICATION
TABLE 3

	Weblog1	Weblog2	Weblog3
No. of Users	49	693	1847
Size (MB)	0.12	0.32	0.47
Number of Sessions	140	362	891
Number of Repeated users	13	81	169

After preprocessing stage 3, the number of users in each web log file is 49,693 and 1847 respectively. The number of unique users thus identified is 36, 612 and 1678 respectively for the 3 web log files selected for study are shown in Table 3.

URLS THAT APPEAR IN EACH SESSION URL-SESSION BINARY TABLE
TABLE 4

Session	URLS in a Session					
1	0	3	5	0	0	…
2	0	0	1	0	0	…
3	1	6	2	0	0	…
⋮	⋮	⋮	⋮	⋮	⋮	⋮

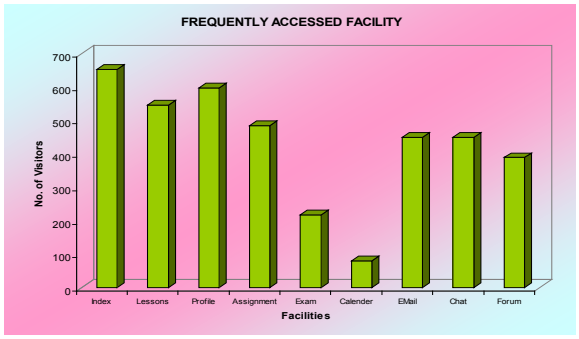
TABLE 5

Session	URLS in a Session					
1	0	3	5	0	0	…
2	0	0	1	0	0	…
3	1	6	2	0	0	…
⋮	⋮	⋮	⋮	⋮	⋮	⋮

TABLE 6
FINAL RESULT

Session	Attribute 1	Attribute 2	…
1	123	200	…
2	150	75	…
3	2	0	…
⋮	⋮	⋮	⋮

Figure 4: Frequently Accessed facility



To assign unique codes to each URL in the web log file, as a first step, all the unique URLs were found. In the next step, the URLS in each session was found and a URL-Session Table as shown in Table 5.4 is created. From Table 4, the binary table is created and a sample is shown in Table 5. Each page visited is considered as a user request and in the present study and the number of times each page is visited in a session is calculated only for cell regions having a value of one and are shown in Table 6.

The 8 options provided on the e-learning portal were again considered for this analysis and the total transactions for the options path were 1,00,578 transactions and all of them were selected for the experiment.

SUPPORT AND CONFIDENCE FOR DIFFERENT FACILITIES

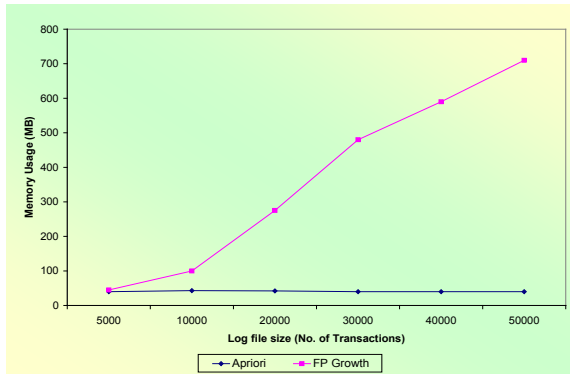
/index ← /profile (15.4/685, 95.7)
/index ← /Lessons (15.4/555, 72.0)
/index ← /EMail (15.4/325, 71.0)
/Lessons ← /Exam (22.0/444, 85.0)
/Lessons ← /Exam (19.5%, 83.9)
/profile ← / Lessons /Exam(23.0/825, 89.1)
/assignment ← /Lessons /Exam (15.0/378, 98.6)
/index ← /Lessons /Email (19.5/244, 83.9)
/index ← /profile /Email (16.5/322, 70.2)
Figure 5: Association Results

TABLE 7

Facility	Support (%)	Confidence (%)
Index	97.32	11.19
Profile	72.32	5.89
Lessons	59.94	56.49
Assignment	42.63	22.35
Exam	8.21	7.73
Calendar	7.16	6.75
Email	33.89	5.99
Chat	33.59	5.94
Forum	27.98	26.37

From Table 7, the results are too obvious, e.g., the support and confidence of the index page and profile page are relatively high (97.32% and 72.32% respectively), followed by lessons, communication facilities, exam and calendar facilities.

The Association Rules along with the support and confidence for identifying the patterns. Based on the figure 5, the rule with higher support (23.0%) means, "if in a session the user selected the /Lesson and /Exam options path, user also selected the /profile option path"; the rule with higher confidence (98.6%) says that "if in a session, the user selected the /Lessons and /exam option path, user also selected the / assignment option path". From the figure, it can be seen that the order of frequency in descending order is index, profile, lessons, assignment, forum, exam, calendar, email and chat.



The ability to produce association rules in a fast and accurate manner occupying minimum memory space is considered very important in web usage analysis. To study the performance of the two algorithms used, (Apriori and FP growth) the following two parameters were used are Memory usage and Execution time .The experiments were conducted for different sized log files, while varying support factor threshold.

The memory usages of both the algorithms are compared as a function of the number of transactions in the log file for support threshold 40% are shown in the Figure 6. The memory requirement of the Apriori algorithm does not depend on the number of transactions The reason for that can be found in the 'candidate generate and test' approach. The number of the candidates does not depend on the number of transactions; it depends only . the item number and on the minimum support threshold. The advantage of the FP-growth algorithm is the quick mining process which does not use candidates. Its drawback is, however, that the memory requirement of the algorithm is huge, Figure 6: Memory usage for Different Sized Log files especially by lower minimum support threshold.

TABLE 8

Web log file size	Apriori	FP-Growth
5000	1.67	1.66
10000	2.58	2.54
20000	3.95	3.94
30000	4.99	4.99
40000	6.02	5.07
50000	7.25	5.97
60000	8.93	7.86
70000	10.12	9.89
80000	11.32	10.42
90000	12.74	11.21
100000	14.25	11.79

TABLE 9

Support	Apriori	FP-Growth
5	6.02	5.07
10	5.99	5.07
15	5.67	5.07
20	5.35	5.05
25	4.17	5.05

30	3.87	5.06
35	3.23	5.07
40	2.98	5.05

RESULTS FOR SUPPORT FACTOR 5% EXECUTION TIME WITH DIFFERENT SUPPORT FACTOR

Table 8 shows the result of apriori and FP-growth algorithms for different sized log files with support factor 5%. The execution time is reported in milliseconds. From the results it is evident that the execution time of the algorithms grows with the dimension of the data set. Both the algorithms behave similarly when the dataset size is small (<=30000), but when the size increases, the best performance is given by the FP-growth algorithm.The results from Table 9 show that the execution time for the FP-growth algorithm is constant when the support factor decreases from 40% to 5% while, at the same time, the execution time of the Apriori algorithm increases dramatically. For a support factor of 25% or greater, the Apriori algorithm has better performances than the FP-growth algorithm . From the experimental data presented it can be concluded that the FP-growth algorithm behaves bet ter than the Apriori algorithm.

CONCLUSION

This paper focused on the discovery of knowledge about the people using the web, their interest and expectations, using Apriori and FP growth algorithm, for web usage mining.

The first successful algorithm proposed to generate all frequent itemsets in a depth-first manner is the Apriori algorithm and the next best known depth-first algorithm is the FP-growth algorithm. The experiments were conducted in two stages, (i) General Pattern Analysis and (ii) Performance analysis in terms of execution time and memory usage. Both the algorithms selected were able to discover access pattern and user behaviours using support and confidence thresholds accurately. From the association rules generated the algorithm was able to selected featured association rules precisely. While testing the algorithms for their memory usage, it is found that the memory requirement of the Apriori algorithm does not depend on the number of transactions while the memory requirement of the FP-growth algorithm increases significantly with the growth of the number of transactions.

From the experimental data presented it can be concluded that the FP-growth algorithm behaves better than the apriori algorithm. The main drawback of Apriori algorithm is that the candidate set generation is costly, especially if a large number of patterns and/or long patterns exist. This study can be extended by incorporating classification or clustering algorithm to predict future user requests.

REFERENCES

[1] Aggarwal, C.C., Li, Y., Wang, J. and Wang, J. (2009) Frequent pattern mining with uncertain data, International Conference on Knowledge Discovery and Data Mining, Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACMPp. 29-38, ISBN-978-3-642-13657-3. | [2] Heydari, M., Helal, R.A. and Ghauth, K.I. (2009) A graph-based web usage mining method considering client side data, International Conference on Electrical Engineering and Informatics, 2009. ICEEI '09, Vol. 01, Pp.147-153. | [3] Kolari, P. and Joshi, A. (2004) Web Mining: Research and Practice, Computing in Science and Engineering, Vol. 6, No. 4, Pp. 49-53. | [4] Park, J.S., Chen, M.S. and Yu, P.S. (1995) An effective hash-based algorithm for mining association rules. Proceeding of the 1995 ACM-SIGMOD International Conference on Management of Data (SIGMOD'95), San Jose, CA, Pp 175–186. | [5] Agrawal R. and Shafer, J.C. (1996) Parallel mining of association rules: design, implementation, and experience, IEEE Trans Knowl Data Eng., Vol. 8, Pp.962–969, ISBN: 1-58113-061-9.Agrawal, R. and Srikant, R. (1994) Fast algorithms for mining association rules in large databases, In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, Santiago, Chile, Pp. 487-499, isbn = {1-55860-153-8}. | [6] Gionis, A., Mannila, H., Mielikainen, T. and Tsaparas, P. (2007) Assessing Data Mining Results via Swap Randomization, ACM Transactions on Knowledge Discovery from Data (TKDD), Volume 1 , Issue 3, Article No. 14. | [7] Chitraa, V. and Davamani, A.S. (2010) A Survey on Preprocessing Methods for Web Usage Data, International Journal of Computer Science and Information Security, Vol. 7, No. 3, Pp.78-83. | [8] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Verkamo, A. I. (1996) Fast discovery of association rules, Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R., editors, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, Pp. 307–328. | [9] Mannila, H., Toivonen, H. and Verkamo, A.I. (1994) Efficient algorithms for discovering association rules, Proceeding of the AAAI'94 Workshop Knowledge Discovery in Databases (KDD'94), Seattle, WA, Pp 181–192. | [10] Savasere, A., Omiecinski, E. and Navathe, S. (1995) An efficient algorithm for mining association rules in large databases, Proceeding of the 1995 International Conference on Very Large Databases (VLDB'95), Zurich, Switzerland, Pp 432–443. | [11] Toivonen, H. (1996) Sampling large databases for association rules,Proceeding of the 1996 International Conference on Very Large Data Bases (VLDB'96), Bombay, India, Pp. 134–145. | [12] Brin, S., Motwani, R., Ullman, J. and Tsur, S. (1997) Dynamic itemset counting and implication rules for market basket data, Proceedings of the International ACM SIGMOD Conference, Tucson, Arizona, USA, Pp. 255–264. | [13] Sarawagi, S., Thomas, S. and Agrawal, R (1998) Integrating association rule mining with relational database systems: alternatives and implications, Proceeding of the 1998 ACM-SIGMOD International Conference on Management of Data (SIGMOD'98), Seattle, WA, Pp. 343–354 . | [14] Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation, Proceeding of the 2000 ACM-SIGMOD International Conference on Management of Data (SIGMOD'00), Dallas, TX, Pp. 1–12. | [15] Webb, G.I. (2000 Efficient Search for Association Rules, R. Ramakrishnan and S. Stolfo (Eds.), Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2000) Boston, MA. New York: The Association for Computing Machinery, Pp. 99-107. |