A study of multilevel association rule mining

Conference Paper · December 2010

CITATION
1

READS
53

1 author:

Faraj A. El-Mouadib
University of Benghazi
20 PUBLICATIONS **21** CITATIONS

SEE PROFILE

A STUDY OF MULTILEVEL ASSOCIATION RULE MINING

Faraj A. El-Mouadib and Amina O. El-Majressi
University of Garyounis, Faculty of Information Technology, Benghazi, Libya
elmouadib@yahoo.com ,
majressi@yahoo.com

ABSTRACT

Recently, the discovery of association rules has been the focus topic in the research area of data mining. For many applications, it is difficult to find strong associations among data items at low or primitive levels of abstraction due to the sparsity of data in multidimensional space. Mining association rules at multiple levels may lead to more informative and refined knowledge from data. Therefore, data mining systems should provide capabilities to mine association rules (refined knowledge) at multiple levels of abstraction.

The objective of this paper is set to explore the concept of the multilevel association rules mining and to study some of the available algorithms for such concept. The work here is carried out in the form of implementing a system for two algorithms, namely; ML-T2 and ML-T2+, for multilevel association rules mining that have been proposed in [7]. UML is used for the analysis and design of our system. The VB6 programming language is used for the implementation. Our system is tested via 66 experiments and the data used in these experiments are mainly synthetic with different sizes ranging from 1.19 MB to 81.8 MB.

***Keywords**: Association rule mining, Concept hierarchies, Data Mining, Knowledge Discovery in Databases, Multi level association rules.*

1. INTRODUCTION

With the wide use of computer applications in many areas such as; business processing (i.e. banking, shopping, etc…) and other applications, massive amounts of data has been accumulated and stored in databases. These massive amounts of data have an implicit hidden knowledge that can not be found via conventional data analysis method and tools. The need to discover such knowledge and make it available for decision makers and business management have called for the development of new tools and techniques in a new field known as Knowledge Discovery in Databases (KDD). According to [3, 4, and 8], KDD is the non-trivial process (is an iterative process) of identifying valid (justified patterns/models, generalized to the future), novel (not something already known), potentially useful (actionable for our task) and ultimately understandable (process leads to human insight) patterns from large amount of data. KDD is also known by other names such as knowledge mining from databases, data mining, knowledge extraction, data/pattern analysis, data archaeology, and data dredging.

In recent years, data mining has become one of the most active and interesting research areas in the field of Artificial Intelligence (AI). According to [4 and 8], Data Mining (DM) is considered to be one of the most essential steps in the KDD process where intelligent methods (algorithms) are applied in to extract patterns or regularities from the data. Not all of the found patterns or regularities are considered to be novel and valid knowledge until they pass some certain threshold (user defined) condition in the pattern evaluation step.

2. KDD AND DATA MINING TASKS

According to [4, 8, 9 and 16], the KDD process consists of a number of iterative sequences of steps. These steps start with the selection of the task relevant data of the current inquiry followed by data preprocessing (i.e. data cleansing, data transformation, etc…). The most important steps are; the data mining step (finding patterns and/or regularities) and pattern evaluation (identifying the truly interesting patterns based on some interesting measures). The final step is to present the knowledge to the user in one form or another.

According to [12], the success of data mining depends largely on the amount of discovered knowledge. Knowledge can come in many different forms. The functionalities of data mining determine the patterns to be mind via one of the different mining tasks. In general and according to [8, 9, 12, 13, 16 and 20], data mining tasks can be classified into two main categories; descriptive and predictive mining tasks.

Descriptive mining tasks are processes that work to characterize the general properties of the data subset (target data) in unsupervised fashion to discover the "natural" structure in the target data. Characterization and discrimination, association analysis, clustering analysis, evolution and deviation analysis are some examples of descriptive mining tasks. Predictive mining tasks are processes of inferring models or functions that governs the properties of the target data to be used to classify new data objects in the appropriate classes. Classification and prediction are some examples of predictive mining tasks.

3. ASSOCIATION RULE BASIC CONCEPTS

In general and according to [2, 8, 12, 13, 20, 21 and 22], Association Rule Mining (ARM) is the process of finding association rules. An association rule is an expression on the form X Þ Y. This rule is read as: "IF X THEN Y". A more formal definition of association rule is given in [1]. The definition states "Let $I = \{i_1, i_2,$

$i_m\}$ be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. Associated with each transaction is a unique identifier, called its *TID*. We say that a transaction T contains X , a set of some items in I , if $X \subseteq T$. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \varnothing$. The rule $X \Rightarrow Y$ holds in the transaction set D with *confidence c* if c% of transactions in D that contain X also contain Y. The rule $X \Rightarrow Y$ has *support s* in the transaction set D if s% of transactions in D that contain $X \cup Y$.".

The calculation of the *support(s)* and *confidence(c)* is performed as follows:

$$\text{Confidence } (X \Rightarrow Y) = \frac{\text{Support } (X \cup Y)}{\text{Support } (X)}$$

$$\text{Support } (X) = \frac{\text{Number of transactions containing item X}}{\text{Total number of transactions in the database}}$$

$$\text{Support}(X \cup Y) = \frac{\text{Number of transactions containing items X and Y}}{\text{Total number of transactions in the database}}$$

4. CONCEPT HIERARCHIES

The idea of different levels of abstraction in data mining is expressed by concept hierarchies. A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts. [8]. A concept hierarchy is a tree-like shape that organizes concepts where the low-level concepts are subset of the higher-level concept that is related to it. For example, figure-1[1] has four levels that are labeled as; 0, 1, 2, and 3. Level 0 is labeled with "All" to represent the most general concept (i. e. all computer related items). Level 1 presents some specific computer related items such as computer systems, software, printer and camera and etc … Level 2 presents more specific computer related items such as laptop computers and desktop computers for computer systems and Office software and antivirus software for Software. Level 3 presents name of manufacturing companies.

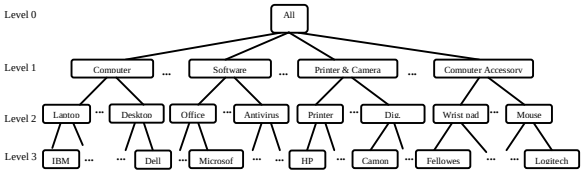


Figure-1: A concept hierarchy for computer related items.

According to [1, 2, 6, 10, 14, 15, 16, 17 and 18], most of the previous work on association rule mining has been focused on single level association rules since it began in the 1990s. On the other hand there are many applications which need to find association rules at multiple levels as well. In this case the use of concept hierarchies becomes an essential tool. Association rules discovered only at very high concept levels represent common sense knowledge. Also it is difficult to find association rules only at low or primitive levels of abstraction due to the sparsity of data. Mining association rules at multiple levels lead to finding useful and more definite clear knowledge. According to [5, 7 and 8], concept hierarchies can be provided by the users or could be deduced automatically or semi-automatically from the data in the database.

5. MULTILEVEL ASSOCIATION RULE MINING

In single level association rule mining there is only one threshold for support and one for confidence. On the other hand in multilevel association rule mining there are as many support and confidence thresholds as there are levels of abstraction except for level 0 (the root node). When working with multilevel association rules, the support and confidence are called minimum support and minimum confidence and these are defined for each level of the concept hierarchy. The minimum support and minimum confidence must be specified before starting the mining process. Any itemset occurs more than or equal to pre-determined minimum support is called frequent itemset thus minimum support is a threshold parameter for the frequency of itemset. Any association rule generated from the frequent itemsets and satisfies the minimum support and the minimum confidence is called strong rule thus minimum confidence is a threshold parameter for the strength of an association rule in a given level. The multilevel association rules discovery is accomplished via the following two steps:

1. Finding all frequent itemsets in every level: A top-down strategy is employed to accumulate counts for the calculation of frequent items at each level. Starting at level 1 and preceding downward in the hierarchy toward the more specific levels, until there are no more frequent itemsets can be found. [8].
2. Generating strong rules for every level.

According to [7], the process of mining association rules at multiple levels progressively refines the knowledge that is found from the given data.

5.1 MULTI LEVEL ARM ALGORITHMS

Most of the work on ARM was focused on single level rules. In the literature there is only one algorithm that deals with multi level association rules mining named ML-T2.

The ML-T2 was first introduced in [7]. The ML-T2 uses two encoded transaction tables to accomplish its' task and it is governed by the following process:

Input: The input to ML-T2 consists of two parts:

1. An encoded database (T[1]) that is the result of coding the transactional database by the use of the concepts and levels of the concept hierarchy used. Each transaction of T[1] is on the form of: <TID, coded items>
2. The minimum support threshold (minsup[l]) for each level of the concept hierarchy (l).

Output: Frequent item sets for mining strong multi level association rules for the relevant set of transactional data.

[1] This figure is borrowed from [8].

Method: A top-down progressively deepening process which collects frequent item sets at different concept levels. The actual algorithm[2] is:

```
for ( l := 1;  L[l, 1] # 0  and  l < max_level;  l ++)
 do begin
   if  l = 1 then
     begin
      L [ l, l] := get_large_1_itemsets (T [l] , l );
      T [2] := get_filtered_transaction_table (T [l], L [1,1] );
     end
     else L [l, 1] := get_large_1_itemsets (T [2], l );
     for ( k := 2; L [l , k - 1] # 0; k++)
      do begin
        Ck:= get_candidate_set (L [l, k - 1] );
        foreach transaction t ∈ T [2]
         do begin
           Ct := get_subsets(Ck, t); /  Candidates contained in t
             foreach candidate c ∈ Ct  do c.support++;
          end
         L [ l, k] := {c ∈ Ck | c.support ≥ minsup[l]}
       end
       L L [l] := Uk  L [l , k];
 end
```

5.2 PERFORMANCE IMPROVEMENTS OF ML-T2 ALGORITHM

As it has been stated in [7], there have been some suggestions to improve the performance of the ML-T2 algorithm on: the sharing of the data structures, intermediate results and maximally generation of results at each database scan. Such ideas had lead to the introduction of three variant algorithms of the original one namely; ML-T1, ML-Tmax and ML-T2+.

The third variation of the ML-T2 algorithm is ML-T2+ which uses the same two encoded transaction tables T[1] and T[2] as in algorithm ML-T2. This algorithm avoids the generation of a group of new filtered transaction tables. It scans T[1] twice to generate T[2] and the large 1-itemset tables for all the levels. Then it scans T[2] once for the generation of each large k-itemset for all the level l ($l \geq 1$). This algorithm is considered as refined technique using two encoded transaction tables. The input and output are the same for both algorithms; ML-T2 and ML-T2+. The ML-T2+ algorithm[3] is:

```
L[1,1] := get_large_1_itemsets(T[1], 1);
{T[2],L[2,1],...,L[max_l,1]}:=get_filtered_transaction_t
able_and_large_1_itemsets(T[l], L[1,1]);
more_results := true;
for  (k := 2; more_results; k++)
  do begin
    more_results := false;
    for  (l := 1; l < max_l; l++)  do
      if  L[l, k - 1] # 0  then
        begin
          C[l] := get_candidate_set (L [l, k - 1]);
          foreach transaction t ∈ T[2]
            do begin
              D[l] := get_subsets(C[l], t);   / /
Candidates contained in t
              foreach candidate c ∈ D[l] do
c.support++;
              more_results := true;
            end
        end
     L [l, k] := {c ∈ C[l] c.support ≥ minsup[l]}
  end
 for (l := 1; l < max_l; l++) do L L [l] := Uk L [l, k];
```

The algorithm ML-T2+.

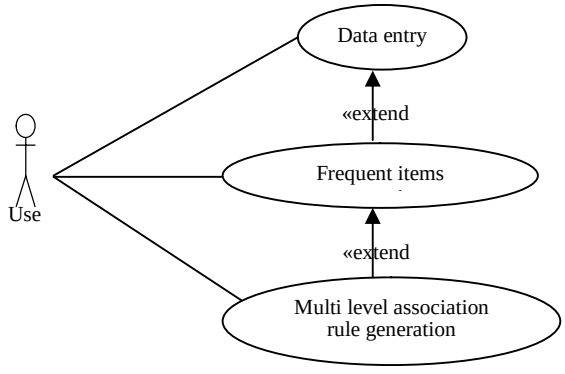
5.3 GENERATING MULTI LEVEL ASSOCIATION RULES

Multi level association rule mining consists of two steps the first step is finding all frequent k-itemsets for all levels of the concept hierarchy via one of the above mentioned algorithms. The second step is to generate multi level association rules for all levels where each frequent k-itemset in any level can produce up to $2^k - 2$ multi level association rules in k^{th} level. The generation of multi level association rules is a straight forward step.

6. DESIGN AND IMPLEMENTATION OF OUR SYSTEM

Our system consists of an implementation of ML-T2 and ML-T2+ algorithms. We used Visual Basic version 6.0 to implement our system and the method used in the analysis and design of the system using Unified Modeling Language (UML). According to [11 and 19], the UML is a graphical language that is suitable to express software or system requirements, architecture, and design. In describing our system, we use only some diagrams of the static and dynamic groups of the UML diagrams.

Static diagrams are also known as structural diagrams. They represent the building blocks of a system features that don't change with time. For our system we used only use case diagram from the group of static diagrams. The Use case diagram is concerned with modeling the functionality of the system.



Figur2: Use case diagram for ML-T2 and ML-T2+ system

Dynamic diagrams are used to show how a system responds to requests and how the system evolves over

[2] This algorithm is borrowed from [7].
[3] This figure is borrowed from [7].

time. From the group of dynamic diagrams, we have only used the activity diagram. An activity diagram is concerned with modeling the activities and the responsibilities of each of the elements of the system.

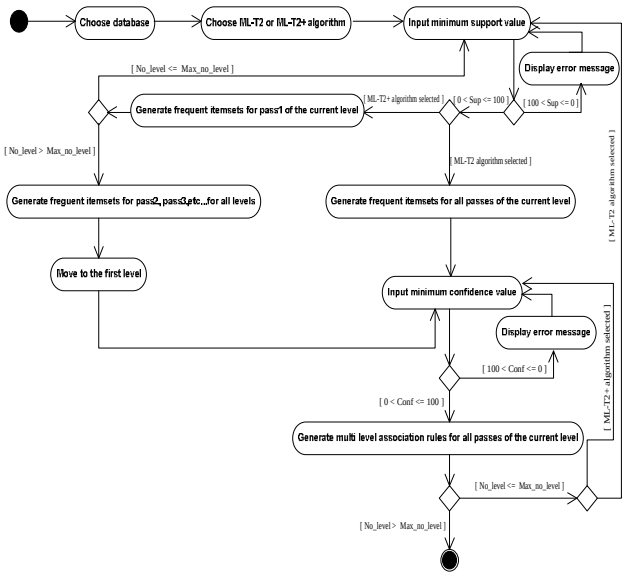


Figure-3: The activity diagram for ML-T2 and ML-T2+

7. EXPERIMENTS AND RESULTS

We tested our system by a number of databases that differ in size. Due to the lack of actual data, we had used databases that are computer generated (synthetic) given the appropriate parameters. All the generated databases are in the form of multi level transactional databases. Table-1 depicts the basic parameters for generating the synthetic multi level transactional databases.

Table 1: Parameters of synthetic multi level transactional databases.

Parameter	Description
I	Represents the total number of items in the database.
D	The total number of transactions in the database.
S	The number of levels presenting an itemset (default=3).
T	The number of items in a transaction (maximum=5).

All of the generated transactional databases are encoded in a transactional table called *T*[1] according to the used concept hierarchy prior to the system execution. All of the experiments are conducted on a laptop computer with 512MB Memory, Celeron processor type, 1.86MHz with Microsoft Windows *XP* professional operating system platform.

7.1 SMALL DATA EXPERIMENT

The medium synthetic data experiment is conducted on a database that consists of 100 transactions and 244 different items (D100.I244). The size of this database is about 7.25 MB. This experiment is performed with eight different minimum support values of 1, 1.5, 1.75, 2, 2.5, 2.75, 3, 4 and 5. The final obtained results are depicted in table 2 and table 3.

Table 2: Tabulation of the medium synthetic data experiment results of: Minimum support values vs. CPU time.

Minimum support values	Algorithm	
	ML-T2	ML-T2+
1	690.058	2254.17
1.5	73.460	79.137
1.75	73.410	76.911
2	73.288	74.698
2.5	63.770	34.320
2.75	62.978	37.171
3	61.338	34.272
4	57.861	22.557
5	56.519	16.730

Table 3: Tabulation of the medium synthetic data experiment results of: Minimum support values vs. number of frequent items.

Minimum support values	Algorithm	
	ML-T2	ML-T2+
1	1677	1677
1.5	222	222
1.75	222	222
2	222	222
2.5	123	123
2.75	123	123
3	123	123
4	92	92
5	66	66

From the above results, we found that:

- For the ML-T2 sub system, whenever the minimum support increased, the CPU time decreased.
- For the ML-T2+ there is no pattern that can be commented on as far as the CPU time is concerned due to the fluctuations. This due to the fact that the ML-T2+ algorithm works in a parallel fashion where large frequent itemsets are generated on all levels for the current pass at the same time before moving to the next pass.
- Even though there is no pattern as far as the CPU time is concerned, the ML-T2+ gave a better time than ML-T2.
- For the last eight minimum support values, the ML-T2+ algorithm had shown an improvement over the ML-T2. Such improvement is ranged from 1.89% to 70.4% and on the average; the improvement was about 37.08%.
- For the number of frequent items, both of the sub systems ML-T2 and ML-T2+ gave exactly the same results but ML-T2+ has taken less time to produce such result.
- The ML-T2+ sub system gave a better result than ML-T2 for this database and for the above mentioned minimum support values.

7.2 MEDUIUM DATA EXPERIMENT

Very large synthetic data experiment is conducted on a database that consists of 1000 transactions and 718 different items (D1000.I718). The size of this database is about 7.66 MB. This experiment is performed with five different minimum support values of 1, 2, 3, 4 and 5. The final obtained results are depicted in table 4 and table 5.

Table 4: Tabulation of very large synthetic data experiment results of: Minimum support values vs. CPU time.

Minimum support values	Algorithm	
	ML-T2	ML-T2+
1	9012.03	8901.111
2	2612.02	1419.030
3	1928.18	1912.836
4	1627.22	1501.105
5	3301.72	3170.783

Table 5: Tabulation of very large synthetic data experiment results of: Minimum support values vs. number of frequent items.

Minimum support values	Algorithm	
	ML-T2	ML-T2+
1	2018	2018
2	1960	1960
3	1161	1161
4	774	774
5	507	507

From examining the above results, we found that:

- For the ML-T2 sub system, whenever the minimum support increased, the CPU time decreased.
- In this experiment is no relation between the performance of the ML-T2+ sub system and minimum support values due to the ML-T2+ sub system works in a parallel fashion.
- Even though there is no pattern as far as the CPU time is concerned, the ML-T2+ gave a better time than ML-T2.
- For the last four minimum support values, the ML-T2+ sub system had shown an improvement over the ML-T2. Such improvement is ranged from 0.8% to 45.68% and on the average; the improvement was about 11.89%.
- For the number of frequent items, both of the sub systems ML-T2 and ML-T2+ gave exactly the same results but ML-T2+ has taken less time to produce such result.
- From the above results, the ML-T2 + sub system was superior to ML-T2 sub systems for this database and for the above mentioned minimum support values.

7.3 LARGE DATA EXPERIMENT

The huge synthetic data experiment is conducted on a database that consists of 50,000 transactions and 729 different items (D50000.I729). The size of this database is about 81.8 MB. This experiment is performed with five different minimum support values of 60, 70, 80, 90 and 100. The final obtained results are depicted in table 6 and table 7.

Table 6: Tabulation of the huge synthetic data experiment results of: Minimum support values vs. CPU time.

Minimum support values	Algorithm	
	ML-T2	ML-T2+
60	9212.023	7892.869
70	8112.020	7790.307
80	7928.183	6087.903
90	7227.220	6930.380
100	6301.720	5957.500

Table 7: Tabulation of the huge synthetic data experiment results of: Minimum support values vs. number of frequent items.

Minimum support values	Algorithm	
	ML-T2	ML-T2+
60	2446	1323
70	2446	1323
80	1086	1086
90	1065	1065
100	1063	1063

From examining the above results, we found that:

- The performance of the ML-T2 sub system gave a better result as the minimum support value is increased.
- Due to the fact that the ML-T2+ sub system works in a parallel fashion, there is no relation between the CPU time and minimum support values.
- The ML-T2+ gave a better time than ML-T2 even though the reached reason in the step two.
- For the last four minimum support values, the improvements of the ML-T2+ algorithm over the ML-T2 ranged from 3.97% to 23.22% and on the average the improvement was about 10.21%.
- ML-T2 and ML-T2+ gave exactly the same results for the number frequent items.
- The ML-T2+ sub system out performed the ML-T2 sub system for this database and for the above mentioned minimum support values.

8. CONCLUSION

We had tested our system with total of 66 experiments but here we represented some of them. All of the data we have used in the experiments are synthetic data sets with sizes ranging from 1.19 MB to 81.8 MB. The results that we had obtained from the experiments are summarized in the following points:

1. ML-T2 sub system has shown very low performance when executed with low minimum support.
2. ML-T2 sub system has shown better performance when the minimum support is high.

3. For the ML-T2+ there is no pattern that can be commented on as far as the minimum support value is concerned due to that the ML-T2+ algorithm works in a parallel fashion where large frequent itemsets are generated on all levels for the current pass at the same time before moving to the next pass.
4. In general, ML-T2+ sub system has better performance than ML-T2 because the ML-T2+ works in a parallel fashion while the ML-T2 works in a sequence fashion where large frequent itemsets are generated for the first level then it go to generating the frequent itemsets for the second level then it go to generating the frequent itemsets for the third level.
5. In most of the experiments that we have conducted, both of the sub systems ML-T2 and ML-T2+ gave the same number of multi level frequent itemsets.

REFERENCES

[1] Agrawal, R., and Srikant, R., "Fast algorithms for mining association rules in large databases", In Proceedings of *20th International Conference on Very Large Databases*, Santiago, Chile, pp.478 – 499, 1994.

[2] Agrawal, R., Imielinski, T., and Swami, A., "Mining association rules between sets of items in large databases", In Proceedings of *International ACM SIGMOD Conference on Management of Data*, Washington, D.C., pp. 207 – 216, 1993.

[3] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P., "The KDD Process for Extracting Useful Knowledge from Volumes of Data", Communications of the ACM, pp. 27 – 34, 1996.

[4] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P., "From Data Mining to Knowledge Discovery in Databases", AAAI/MIT Press, pp. 37 – 54, 1996.

[5] Han, J., "Knowledge at multiple concept levels", In CIKM. pp. 19 – 24, 1995.

[6] Han, J., and Fu, Y., "Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases", In AAAI94 Workshop on knowledge Discovery in Databases (KDD94), Seattle, WA, pp.157–168, 1995.

[7] Han, J., and Fu, Y., "Discovery of Multiple-Level Association Rules from Large Databases", Proceedings of *the 21st International Conference on Very Large Databases*, Zurich, Swizerland, pp. 420 – 431, 1995.

[8] Han, J., and Kamber, M., Data Mining: *Concepts and Techniques*, Second Edition, Morgan Kaufmann, San Francisco, 2006.

[9] Hand, D., Mannila, H., and Smyth, P., *Principles of Data Mining*, The MIT press, Cambridge: MA, 2001.

[10] Hipp, J. Guntzer, U., and Nakhaeizadeh, G., "Algorithms for Association Rule Mining – A General Survey and Comparison", ACM SIGKDD, pp.58 – 64, 2000.

[11] Jesse, M., and Schardt, J., *UML 2 for Dummies*, Wiley Publishing, Inc, 2003.

[12] Kantardzic, M., *Data Mining: Concepts, Models, Methods, and Algorithms*, John Wiley & Sons, 2003.

[13] Larose, D., *Discovering Knowledge in Data an Introduction to Data Mining*, John Wiley & Sons, Inc., Hoboken, New Jersey, 2005.

[14] Mannila, H., Toivonen, H., and Inkeri, A., "Efficient Algorithms for Discovering Association Rules", Seattle, Washington, pp. 181 – 192, 1994.

[15] Park, J. S., Chen, M.S., and Yu, P.S., "An Effective Hash Based Algorithm for Mining Association Rules", In Proc. *ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'95)*, San Jose, CA, pp. 175 – 186, 1995.

[16] Sarker, R. Abbass, H. and Newton, C., *Introducing Data Mining and Knowledge Discovery*, Idea Group Publishing,2002.

[17] Savasere, A., Omiecinski, E., and Navathe, S., "An Efficient Algorithm for Mining Association Rules in Large Databases", The VLDB Journal, Morgan Kaufmann Publishers Inc, pp. 432 – 444, 1995.

[18] Sergey, B., Rajeev M., Jeffrey U., and Shalom, T., "Dynamic Itemset Counting and Implication Rules for Market Basket Data", Proceedings of *the ACM SIGMOD Conference*, pp. 255 – 264, 1997.

[19] Si, S., Learning UML, O'Reilly, 2003.

[20] Two Crows Corporation, *Introduction to Data Mining and Knowledge Discovery*, Third Edition, 1999.

[21] Witten, I., and Frank, E., "Data Mining: practical machine learning tools and techniques with java implementations", Morgan Kaufmann, San Francisco, pp. 371 – 375, 2000.

[22] Ye, N., *The handbook of data mining*, LEA (Lawrence Erlbaum Associates), publishers, Mahwah, New Jersey, London, 2003.