The European Proceedings of
Social & Behavioural Sciences
EpSBS

Future Academy

ISSN: 2357-1330

https://dx.doi.org/10.15405/epsbs.2018.11.02.62

ICPE 2018
International Conference on Psychology and Education

ALGORITHMS AND DATA STRUCTURES FOR ASSOCIATION RULE MINING AND ITS COMPLEXITY ANALYSIS

Alexander Prutzkow (a)*
*Corresponding author

(a) Ryazan State Radioengineering University, Department of Computational and Applied Mathematics, 390005, Gagarina str., 59/1, Ryazan, Russia, mail@prutzkow.com

Abstract

Algorithms are characterized by the time of solution of the problem and the data structure used. We propose a method for estimating the time complexity and space complexity and analyze algorithms for association rules mining. To compare the algorithms, we estimate the time complexity of each iteration of processing of all transactions in the worst case. The space complexity is estimated from the size of data structure used by the algorithm, without taking into account its physical implementation in the development of software. The worst case is also used. In accordance with the method, we estimate the time and space complexities of following algorithms: the Apriori algorithm, the FP-Growth algorithm, the matrix algorithm, the SimpleARM algorithm, the algorithm using multi-layer matrix quadrants, and algorithms using the transposition of transaction table. These algorithms use various types of the data structure such as: matrices, trees, and lists. Analyzed result of estimating we conclude that no algorithm surpasses the others by compared complexity. We also survey problems related to association rule mining: sequential pattern mining and association rule hiding.

© 2018 Published by Future Academy www.FutureAcademy.org.UK

Keywords: Association rule, association rule mining, algorithm, time complexity, space complexity.

The Author(s) 2018 This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (https://creativecommons.org/licenses/by-nc-nd/4.0/).

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

1. Introduction

There are some events. Every event has characteristics. The characteristic is a element from a finite set. So the event is characterized by a set of elements. We need to discover the most frequent sets of elements in events. The events are named transactions, elements are named items, and itemset are used for association rule constructing. Association rule mining is finding the most frequent itemsets in transactions.

Association rule mining or market-basket analysis is most often used in marketing to discover the most frequent sets of goods in checks (transactions) and make marketing strategies for sales.

Other subjects where the association rule mining are used:

1) education: transaction – subject marks of pupil; item – the mark.

2) text analysis: transaction – text fragment; item – lexical, morphological, syntactic and semantic attribute;

3) banking services: transaction – bank services for the client; item – deposits, loans, bank cards;

4) medicine: transaction – result of surveys and monitoring received before, during and after impact to the patient; item – the actual result of the surveys.

2. Problem Statement

2.1. Notation and Terminology

We introduce the following notation and terminology.

$E = \{e_1, e_2, \dots, e_m\}$ is the set of some elements or items. An item of the set E can be a commodity, an attribute, a property.

$D = \{T_1, T_2, \dots, T_n\}$ is the transaction set (or transaction database) of $T_i \subseteq E$, $i = 1, 2, \dots, n$. Usually in practics the set D is one or more relational tables of a database. Each transaction has a TID number (in this case, an index).

$X \Longrightarrow Y$ is the association rule, $X \subset E$ is the left side of the rule, $Y \subset E$ is the right side of the rule, such that $X \cap Y = \emptyset$. Consider the association rule $\{x_1, x_2\} \Longrightarrow \{y_1, y_2\}$. The association rule means the following: «If x_1, x_2 are in the transaction, then we can predict that y_1, y_2 will also be in the transaction».

k is the size of the itemset or the $X \Longrightarrow Y$ association rule ($k = |X| + |Y|$).

$kmax$ is the maximum size of the discovered association rule.

$tmax = \max\{|T|: T \in D\}$ is the maximum transaction length.

For this sets, the following relation holds:

$$n \gg m \gg tmax > kmax. \quad (1)$$

We imply $n \geq 1{,}000{,}000$, $m \geq 1{,}000$.

2.2. Measures of Interestingness of Association Rule

The interestingness of an association rule is determined mainly by the frequency in the set D. There are the following measures of the interestingness of association rules:

$supp$ – support for the rule; $supp(X \Longrightarrow Y) = P(X \Longrightarrow Y) = \frac{|\{T:(X \cup Y) \subseteq T\}|}{|D|}$. Support characterizes the frequency of the rule in the set of transactions. In some cases, the support count is used as well: $suppcount(X \Longrightarrow Y) = |\{T: (X \cup Y) \subseteq T\}|$.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

$conf$ – the confidence of the rule; $conf(X \Longrightarrow Y) = P(Y|X) = \frac{P(X \Rightarrow Y)}{P(X)} = \frac{|\{T:(X \cup Y) \subseteq T\}|}{|\{T:X \subseteq T\}|}$. The confidence characterizes the share of transactions with the left side of the rule, which also contains the right-hand side.

$lift$ – lift of the rule; $lift(X \Longrightarrow Y) = \frac{supp(X \Rightarrow Y)}{supp(X) \cdot supp(Y)}$. The lift characterizes the relationship between the left and right hands of the rule.

lev – leverage of the rule; $lev(X \Longrightarrow Y) = supp(X \Longrightarrow Y) - supp(X) \cdot supp(Y)$. The leverage also characterizes the relationship between the left and right hands of the rule.

When solving the problem of association rule mining, limiting value of measure is set, which determines whether the association rule is interesting or not. Typically, it is $minsupp$ (in some cases $minsuppcount$) or $minconf$.

The measures discuss in more detail in (Bremer, 2016).

2.3. Itemsets and Association Rules

Let $R \subseteq E$ be an itemset. The algorithms for association rule mining, as a rule, first discover the itemsets, and then construct the rules from them.

Example. Let $\{e_1, e_2, e_3\} \subset E$, $supp(e_1) \neq supp(e_2) \neq supp(e_3)$.

From the elements e_1, e_2, e_3, the following association rules can be constructed:

$\{e_1\} \Longrightarrow \{e_2, e_3\}$, $\{e_1, e_2\} \Longrightarrow \{e_3\}$, $\{e_1, e_3\} \Longrightarrow \{e_2\}$, $\{e_2\} \Longrightarrow \{e_1, e_3\}$, $\{e_2, e_3\} \Longrightarrow \{e_1\}$, $\{e_3\} \Longrightarrow \{e_1, e_2\}$.

All these rules will have equal support, since all rules consist of the same elements. However, the confidence of these rules will be different, because their left parts are different, which affects the calculation of confidence.

Among the constructed rules, those whose confidence is greater than $minconf$ are selected. □

We formulate the problem of association rule mining in the following way. Find all $X \Longrightarrow Y$ rules such that $supp(X \Longrightarrow Y) > minsupp$ or $conf(X \Longrightarrow Y) > minconf$ (depending on which input data is specified).

There are many algorithms for association rule mining. However, none of them became classical. Algorithms use different data structures, have different time and space complexities.

3. Research Questions

The following research questions guide the current study:

Question 1: What data structures are algorithms for association rule mining used?

Question 2: How estimate time and space complexities of the algorithms?

Question 3: What algorithm has the best complexity?

4. Purpose of the Study

The purpose of the study is to develop a method for analyzing algorithms for association rule mining, identifying the data structures they use, estimating time and space complexities, and other characteristics of these algorithms.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

There are surveys of algorithms by other researchers. In (Hipp, Gontzer, & Nakhaeizadeh, 2000) the time for association rule mining by the algorithms Apriori, DIC, Partition, and Eclat in the sets D of different sizes was compared. In (Kotsiantis & Kanellopoulos, 2006) various approaches to improving mining efficiency and implementing these approaches in various algorithms were considered.

5. Research Methods

Our research has the following stages:

- To analyze algorithms for association rule mining and select the algorithms which have an original data structure.
- To develop a method for estimating the time and space complexities.
- To estimate complexities of the selected algorithms.

6. Findings

In the result of our research we have got the following findings.

6.1. Method for Estimating the Time and Space Complexities of Algorithms for Association Rule Mining

We estimate complexity in the following way.

6.2. Estimate of the Time Complexity

Estimates of time and space complexity are measures of comparison of algorithms. All the algorithms for association rule mining, considered in this paper, process every element of the set D.

The time complexity of the search for TC can be written as follows:

$TC = TC_D + TC'$,

where TC_D – time complexity of processing of the elements of the set D; TC' is the time complexity of other steps.

Starting from the relation (1), we can conclude that

$O(TC) = O(TC_D)$.

Therefore, to compare the algorithms, we estimate the time complexity of each iteration of processing of the elements of the set D in the worst case.

6.3. Estimate of the Space Complexity

The space complexity is estimated from the size of data structure used by the algorithm, without taking into account its physical implementation in the development of software. The worst case is also used.

6.4. Algorithms for Association Rule Mining

In the following sections, algorithms for association rule mining will be described. The main criterion for choosing these algorithms was the originality of the data structures they used. Data structures can be considered as a method of compression (usually with losses) of the set D.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

The description of algorithms has the following structure:

- the main steps and a short description of the data structure used, links to the articles of the authors who proposed this algorithm, and other researchers, in which the algorithm is described in more detail;
- a description of the actions at each iteration of the process of elements of the set D;
- estimating of the time complexity of one iteration the process of elements of the set D;
- estimating of the space complexity of the data structure used;
- features of the algorithm and the data structure.

6.5. Apriori Algorithm

The Apriori algorithm (Agrawal & Srikant, 1994) counts for all sets of elements of the set E of length k ($k = 1, 2, ...$) in the list L_k the number of their occurrences in the transaction, excludes from the list of sets whose support is less than the specified one, and generates a new list L_{k+1} sets of length $k + 1$ based on the remaining ones. The algorithm is executed until new sets are generated ($L_{k+1} = \emptyset$).

At each iteration, you need to get from the transaction a set of elements of length k and increase by 1 those sets in the list L_k that have an entry in the transaction. Therefore, the estimate of the time complexity of one iteration is $O(C_{tmax}^{kmax})$.

The Apriori algorithm uses a list of sets of elements of length k. Estimate of its size $O(C_m^{kmax})$.

The Apriori algorithm at each step stores the minimum possible amount of data. However, the algorithm requires a large number of transaction passes, which increases its time complexity.

A step-by-step implementation of the Apriori algorithm is presented in (Bremer, 2016). The implementation of the Apriori algorithm in Apache Spark is presented in (Mehta, 2017).

6.6. Modifications of the Apriori Algorithm

The Apriori algorithm is the first solution of the problem of association rule mining. After publication of the algorithm, it was modificated in different ways. We review some modifications.

The AprioriTID algorithm (Agrawal & Srikant, 1994) uses for storing a list. Every element of the list consists of the transaction number TID and the collection of itemsets discovered in it.

The Direct Hashing and Pruning (DHP) algorithm (Park, Chen, & Yu, 1995) uses a hash table to store L_k, which increases the required memory, but shortens the execution time of the algorithm. Note that the itemset and association rule can be represented as a combination of their elements. Each combination can be put in a one-to-one correspondence with a natural number. The combination of elements is uniquely transformed into a number and back (Knuth, 2005). These transformations can be used as hash functions without collisions. However, in transformations it is necessary to calculate the number of combinations C_n^m, and hence factorials, that implies a large amount of computation.

The K-Apriori algorithm (Loraine Charlet Annie & Ashok Kumar, 2012) uses clustering by the k-means method and the Wiener linear transformations.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

6.7. FP-Growth Algorithm

The FP-Growth (Frequent Pattern Growth) algorithm (Han, Pei, & Yin, 2000) includes the following steps. At the first transaction pass, for each element of the set E, the number of its entries in the set D is counted. In the second pass, the elements of transactions are sorted in the order determined at the first pass, they are added to the tree, forming a path. If an element is already present in the tree, then the value in the corresponding node increases, otherwise a new node with a value of 1 is added to the tree.

At each iteration of the second pass, the transaction elements are sorted (time complexity of sorting: $tmax \cdot log_2 tmax$) and add the sorted elements to the tree (m). Proceeding from the relation (1), the estimate of the time complexity of one iteration is $O(m)$.

Let's estimate the size of the tree being built. At the zero level, there is a single empty node. At the first level, the tree has $C_m^1 = m$ nodes. The first node of the first level of the tree has $m - 1$ child nodes, the second vertex – $m - 2$ child nodes, etc. The penultimate node of the first level has one child, and the last node has no child nodes. In total on the second level $C_m^2 = \frac{m!}{2(m-2)!}$ nodes. The third level of tree has C_m^3 nodes, etc. At the level $kmax$ there are C_m^{kmax} nodes. Then the tree size estimate is $O(C_m^{kmax})$.

Step-by-step execution of the FP-Growth algorithm see in (Bremer, 2016). The implementation of the FP-Growth algorithm in Apache Spark is presented in (Mehta, 2017).

The tree most compactly represents the set D. However, bypassing it and searching for necessary nodes requires extra time computing (in comparison with arrays).

6.8. Matrix Algorithm

Matrix algorithm (Yuan & Huang, 2005) uses the binary matrix $B = \{b_{ij}\},\ i = 1, 2, \dots, n;\ j = 1, 2, \dots, m$ as the data structure, where

$$b_{ij} = \begin{cases} 1, e_j \in T_i, \\ 0, e_j \notin T_i. \end{cases}$$

The estimate of the space complexity of the matrix algorithm is $O(n)$.

The matrix algorithm is similar to the Apriori algorithm, with the difference that transactions are not moved in the set D, but in the binary matrix B and the number of entires of the set in the transaction is counted by multiplying the binary vectors: the vector of size m corresponding to the set and the row of the matrix b_i. The matrix algorithm requires only one transaction scan to form the matrix B. However, the size of the obtained matrix B is commensurable with the size of the set D. Therefore, the actual number of searches is the same as in the Apriori algorithm with an additional search for the matrix B: $kmax + 1$.

At the first pass at each iteration, the transaction is converted to the row of the matrix b_i in m steps. For the next passes, two vectors of size m in m steps are multiplied. Therefore, the estimate of the complexity of one iteration of the transaction search is $O(m)$.

6.9. The SimpleARM Algorithm

The SimpleARM (Simple Association Rules Miner) algorithm (Saraee & Al-Mejrab, 2004) is analogous to the Apriori algorithm. With a single transaction scan, the set D is written into six matrices and arrays:

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

1) GeneralMatrix (size $m \times m$), (i, j)-th value of which is equal to the number of simultaneous occurrences in the transaction of the elements i and j;

2) Before_Matrix (size $m \times m$), (i, j)-th value of which is equal to the number of precedings in the transactions of element i to element j;

3) After_Matrix (size $m \times m$), (i, j)-th value of which is equal to the number of followings in the transactions of the element i in the element j;

4) Items_Occurrences (size $m \times tmax$), (i, j)-th value of which is equal to the number of occurrences of the element i in the transaction of length j;

5) Transaction Length Array (size $tmax$) whose i-th value is equal to the number in the transaction of length i;

6) Transaction Arrays (total size $m \times n \times 2$); each array has size $n \times 2$ in the worst case; each row of the array contains the transaction number and position in this transaction of the element corresponding to the array.

To calculate the support for sets of elements of length 1, only the Items_Occurrences matrix is used. To calculate the support for sets of length 2, only the GeneralMatrix is used. To calculate the support for sets of length 3, the Before_Matrix is also used. Supports sets of all other lengths using element of the transaction arrays.

It is assumed that only one transaction search is needed to fill the matrices and arrays. However, for sets of length greater than 3, it is necessary to enumerate elements of transaction arrays whose size is commensurate with the power of the set D. This leads to an increase in the number of searches.

At each iteration of the transaction search, for each element of the transaction, the corresponding values in matrices and arrays increase. Therefore, the estimate of the time complexity of one iteration of the transaction processing is $O(tmax)$.

The estimate of the space complexity of the Transaction Arrays is $O(n)$, which determines the space complexity of the SimpleARM algorithm.

6.10. Algorithm Using Matrix Quadrants

Formally, a data structure is presented in (Fajriya Hakim, 2013), but not an algorithm for association rule mining. However, for the search, the algorithm can be used the Apriori, modified for the proposed data structure.

The data structure is a $2m \times 2m$ matrix, divided into four quadrants by x and y axes with positive and negative numbers of the E elements. Each cell contains a list of transaction numbers. For the set $\{e_1, e_2, e_3, e_4\}$, the transaction number is added to the list of numbers in the cells (e_1, e_1), (e_1, e_2), (e_1, e_3), (e_1, e_4), (e_2, e_2), (e_2, e_3), (e_2, e_4), (e_3, e_3), (e_3, e_4), (e_4, e_4) in the first quadrant, into cells $(e_{-2}, e_3), (e_{-2}, e_4)$, (e_{-3}, e_4) in the second quadrant, into the cell (e_{-3}, e_{-4}) in the third quadrant. The elements of the set E must be ordered in lexicographic order. Another layer with a matrix of the same size is added to record an itemset of more than 5 elements.

The matrix of size $2m \times 2m$ has the order of space complexity $O(m^2)$. However, each cell in the matrix contains a list of transaction numbers. In the worst case, the list has length n. Therefore, the estimate of the space complexity of the data structure used by the algorithm using matrix quadrants is $O(n)$.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

With a single search at each iteration of the transaction elements, all possible combinations are formed for $\sum_{i=1}^{tmax} C_{tmax}^{i} = 2^{tmax} - 1$ steps. The estimate of the complexity of one iteration of the transaction search is $O(2^{tmax})$. Also, when calculating the support for a set of length k, it is necessary to find the number of matching elements in lists of length n by $k - 1$ times.

6.11. Algorithm Using the Transposition of Transaction Table

The algorithm using the transposition of transaction table (Gunaseelan & Uma, 2012) is also based on the Apriori algorithm. The transposed transactional table consists of strings including the element number of the set E and the list of transaction numbers that include this element.

With a single enumeration at each iteration for each item in the transaction list, the line number corresponding to this element is added by the number of the current transaction. The estimate of the time complexity of one iteration of the transaction processing by the algorithm is $O(tmax)$.

As in the algorithm using matrix quadrants, to calculate the support, it is necessary to find the number of matching elements in the transaction lists of length n in the worst case.

The transposed transaction table has dimensions of $m \times n$. The estimate of the space complexity of the data structure used by this algorithm is $O(n)$.

6.12. Probabilistic Algorithms

All the above algorithms are exact. The association rule mining ceased to be only a practical problem and became mathematical. Therefore, probabilistic algorithms for association rule mining were proposed.

Due to the high complexity of algorithms for association rule mining related to transaction passes, algorithms have been proposed (for example (Toivonen, 1996)) that discover rules not for the entire set D, but for some sample of this set. To select the sample size, special algorithms have been developed (for example (Chuang, Chen, & Yang, 2005)).

Each rule has a cost, since it can increase profits. If an expensive rule is not discovered, this will lead to a loss of potential profit. The effectiveness of probabilistic algorithms raises questions. Therefore, probabilistic algorithms were not covered in this comparison.

6.13. Distributed Mining Capability

Let the set D be divided into parts: $D = \bigcup_{i=1}^{q} D_i$. We will denote $S_a(D)$ the data structure used by the algorithm a and containing the values obtained from the set D in a certain way. We denote by $\bigcup_{i=1}^{q} S_a(D_i)$ the data structure obtained by combining the data structures $S_a(D_i)$ in a certain way.

Algorithm a has the capability of distributed mining, if:

$S_a(D) = \bigcup_{i=1}^{q} S_a(D_i)$.

Let's consider the possibility of distributed mining by compared algorithms in this section.

For the reviewed algorithms we conclude that if the data structure used by the algorithm can restore the set D and the elements in the structure are not ordered, then the algorithm has the capability of distributed mining. Therefore, the matrix algorithm, the SimpleARM algorithm, algorithm using the matrix quadrant, and algorithm using the the transposition of transaction table have this capability.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

From the tree used by the FP-Growth algorithm, you can also restore the set D. However, before the tree is constructed, the elements are sorted (ordered). For different D_i, the order of the elements will be different, therefore, trees of different D_i can not be combined without additional transformations.

The Apriori algorithm at each iteration of the transaction search eliminates sets that support less than $minsupp$. For different D_i, the excluded sets of elements will be different. Therefore, this algorithm does not have the capability of distributed mining.

There are specially developed algorithms for distributed and parallel mining for association rules, including those based on the Apriori algorithm. The algorithms are discussed in (Shi, 2011). In the comparison conducted in the paper, the question of the possibility of distributed mining with the data structure used by the algorithm was considered.

6.14. Analysis of Complexity Estimates of Algorithms for Association Rule Mining

We reduce the complexity estimates of the comparable association rule mining algorithms to a table.

Table 01. Comparison of complexity estimates of algorithms for association rule mining

	Apriori	**FP-Growth**	**Matrix**	**SimpleARM**	**Using Matrix Quadrants**	**Using the transposition of transaction table**
Used data structure	List	Tree	Binary Matrix	6 Matrices and Arrays	Matrix and List	Matrix and List
The time complexity of one iteration of the transaction pass	$O(C_{tmax}^{kmax})$	$O(m)$	$O(m)$	$O(tmax)$	$O(2^{tmax})$	$O(tmax)$
The space complexity	$O(C_m^{kmax})$	$O(C_m^{kmax})$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Count of passes	$kmax$	2	1 $(kmax+1)$	1 $(kmax+1$ if $kmax>3)$	> 1	> 1
Distributed mining capability	No	No	Yes	Yes	Yes	Yes

Conclusions from the comparison of algorithms for association rule mining:

- The Apriori algorithm is the only algorithm among those considered in this paper, output of which is the final answer after filling out the data structure with values obtained from the set D. The remaining algorithms have two main phases:

1) filling the data structure with values obtained from the set D;

2) mining of association rule in this structure.

- Most of the compared algorithms transform the set D into data structures commensurate in size with the set D. As a result, the association rule mining in such structures requires considerable amount of time. We find the tree using the FP-Growth algorithm is the best data structure since the tree is a compact representation of the set D.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

- At the same time, if the data structure used by the analyzed in this paper algorithm can reconstruct the set D and the elements in the structure are not ordered, then the algorithm has the possibility of distributed mining, which is actual with the constant growth of the processed data.
- Among the compared algorithms there is no algorithm that surpasses the others in terms of time and space complexities.
-

6.15. Problems Related to Association Rule Mining

There are problems somehow related to the association rule mining.

6.16. Sequential Pattern Mining

Let each transaction be associated with the customer ID number, date and time. It is necessary to find among them the most frequently occurring sequences of elements of the set E, sorted by time. This problem is a Sequential Pattern Mining. In more detail this problem and algorithms for its solution are considered in (Agrawal & Srikant, 1995).

6.17. Association Rule Hiding

As mentioned above, the association rules have a price, like the set D. The aggregate of transactions became a commodity. However, in some cases it is necessary that sensitive association rules have support or confidence lower than they have in the set D. The problem arises of hiding association rules by transforming transactions of the original set D and obtaining a new sanitized set D'.

The transformation of the set D introduces three rules:

1) all sensitive association rules must have support or confidence in the set D' below the given one;

2) all nonsensitive association rules should have in the set D' support or confidence the same or greater than they had in the set D;

3) there should not be any mined association rules in the set D', which are not mined in the set D.

The problem of association rule hiding and its solutions are discussed in detail in (Gkoulalas-Divanis & Verykos, 2010).

7. Conclusion

1. We demonstrate that the problem of association rule mining is widely used in various subject areas.

2. We propose the method for estimating the time and space complexities of the algorithms for association rule mining. The time complexity of the algorithm is estimated by the number of operations at each iteration of the transaction pass in the set D, since the set D size exceeds the sizes of the other used sets. The space complexity of the algorithm is estimated from the used data structure.

3. We estimate the time and space complexities of the algorithms for association rule mining using various data structures. Comparison of these parameters of algorithms is made and conclusions are formulated. The main conclusion from the comparison is that none of the algorithms exceed the other algorithms for time and space complexities.

https://dx.doi.org/10.15405/epsbs.2018.11.02.62
Corresponding Author: Alexander Prutzkow
Selection and peer-review under responsibility of the Organizing Committee of the conference
eISSN: 2357-1330

4. The problems accompanying the problem of association rule mining are reviewed: the sequential pattern mining and the association rule hiding in the set D.

There is a remark. The association rule mining is often referred to the problem of artificial intelligence. We guess artificial intelligence solves intellectual problems. The intellectual problems are problems for which there are no conventional, classical solutions (superior to other solutions or optimal in certain parameters) and for which often it is necessary to develop a solution algorithm (unlike most mathematical problems). The association rule mining, as demonstrated in this paper, does not have a generally accepted solution, but at the same time it has a large number of algorithms with different complexity. Consequently, this association rule mining is the problem of artificial intelligence.

We believe our estimates help to other researchers to develop a data structure for the effective association rule mining.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *VLDB'94: Proc. of the 20th Intl. Conference on Very Large Data Bases, San Francisco, USA* (pp. 487-499).

Agrawal, R., & Srikant, R. (1995). Mining Sequential Patterns. In *Proc. of the 11th Intl. Conference on Data Engineering* (pp. 3-14).

Bremer, M. (2016). *Principle of Data Mining*. 3rd ed. Springer.

Chuang, K.-T., Chen, M.-S., & Yang, W.-C. (2005). Progressive Sampling for Association Rules Based on Sampling Error Estimation. In T.B. Ho, D. Cheung, and H. Liu (Eds.), *PAKDD 2005, LNAI 3518* (pp. 505-515).

Fajriya Hakim, R. B. (2013) New Method to Mining Association Rules Using Multi-Layer Matrix Quadrant. *Media Statistika*, *6*(2), 61-70.

Gkoulalas-Divanis, A., & Verykos, V. S. (2010). *Association Rules Hiding for Data Mining*. Springer.

Gunaseelan, D., & Uma, P. (2012). An Efficient Algorithm for Mining Association Rules in Massive Datasets. *Global Journal of Computer Science and Technology Software & Data Engineering, 12*, (13).

Han, J., Pei, J., & Yin, Y. (2000). Mining Frequent Patterns without Candidate Generation. In *Proc. ACM SIGMOD Intl. Conference on Management of Data*.

Hipp, J., Gontzer, U., & Nakhaeizadeh, G. (2000). Algorithms for Association Rule Mining – A General Survey and Comparison. *SIGKDD Explorations*, *2*(1), 58-64.

Knuth, D. E. (2005). *The Art of Computer Programming. Volume 4, Fascicle 3: Generating All Combinations and Partitions* (pp. 5-6). Addison-Wesley.

Kotsiantis, S. & Kanellopoulos, D. (2006). Association Rules Mining: A Recent Overview. In *GESTS International Transactions on Computer Science and Engineering*, *32*(1), 71-82.

Loraine Charlet Annie, M. C., & Ashok Kumar, D. (2012). Market Basket Analysis for a Supermarket based on Frequent Itemset Mining. *International Journal of Computer Science Issues, 9*(5), 3.

Mehta, R. (2017). *Big Data Analytics with Java*. Packt Publishing.

Park, J. S., Chen, M.-Y., & Yu, P. S. (1995). An Effective Hash-Based Algorithm for Mining Association Rules. In *Proc. 1995 ACM SIGMOD Intl. Conference on Management of Data* (pp. 175-186).

Saraee, M., & Al-Mejrab, M. (2004). One Scan is Enough: Optimising Association Rules Mining. In *Proceedings of the International Conference on Information and Knowledge Engineering*, (pp. 491-494). CSREA Press.

Shi, Z. (2011). *Advanced Artificial Intellegence*. World Scientific Publishing.

Toivonen, H. (1996). Sampling Large Databases for Association Rules. In *Proc. of the 22nd VLDB Conference Mumbai (Bombay), India* (pp. 134-145).

Yuan, Y., & Huang, T. (2005). A Matrix Algorithm for Mining Association Rules. In D.S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), *ICIC 2005, Part I, LNCS 3644* (pp. 370-379).