# ArmSquare: An Association Rule Miner Based on Multidimensional Numbered Information Spaces

Iliya Mitov, Krassimira Ivanova
Institute of Mathematics and Informatics, BAS
Sofia, Bulgaria
mitov@mail.bg, kivanova@math.bas.bg

Benoit Depaire, Koen Vanhoof
Hasselt University
Hasselt, Belgium
benoit.depaire@uhasselt.be, koen.vanhoof@uhasselt.be

***Abstract*** **– In this article, we propose a simple approach for association rule mining, which uses the possibilities of the multidimensional numbered information spaces as a storage structures. The main focus in the realization of ArmSquare is using the advantages of such spaces, i.e., the possibility to build growing space hierarchies of information elements, the great power for building interconnections between information elements stored in the information base, and the possibility to change searching with direct addressing in well structured tasks. The tested types of implementations of realized tool show the vividness of proposed approach.**

***Keywords – Association Rule Mining; Market Basket Analysis; Multidimensional Numbered Information Spaces.***

## I. INTRODUCTION

Data mining stands at the crossroad of databases, artificial intelligence, and machine learning. Association rule mining (ARM) is a popular and well researched method for discovering interesting rules from large collections of data. Association rule mining has a wide range of applicability, such as market basket analysis, gene-expression data analysis, building statistical thesaurus from the text databases, finding web access patterns from web log files, discovering associated images from huge sized image databases, etc.

The contemporary databases are very large, reaching gigabytes and terabytes, and the trend shows further increase. Therefore, for finding association rules one requires efficient scalable algorithms that solve the problem in a reasonable time. The efficiency of frequent itemset mining algorithms is determined mainly by three factors: (1) the way candidates are generated; (2) the data structure that is used; and (3) the implementation details. Most papers focus on the first factor, some describe the underlying data structures, and implementation details are almost always neglected [1].

### A. Problem description

The description of the problem of association rule mining is firstly presented in [2]. Below, the description of the problem follows one given in [3].

Let $\Im$ be a set of items. A set $X = \{i_1, ..., i_k\} \subseteq \Im$ is called an itemset or a k-itemset. A transaction over $\Im$ is a couple $T = (tid, I)$ where $tid$ is the transaction identifier and $I$ is an itemset. A transaction $T = (tid, I)$ is said to support an itemset $X \subseteq \Im$ if $X \subseteq I$. A transaction database $D$ over $\Im$ is a set of transactions over $\Im$. The cover of an itemset $X$ in $D$ consists of the set of transaction identifiers of transactions in $D$ that support $X$. The support of an itemset $X$ in $D$ is the number of transactions in the cover of $X$ in $D$: $support(X, D) := |cover(X, D)|$. An itemset is called frequent if its support is no less than a given absolute minimal support threshold $\sigma$. The collection of frequent itemsets in $D$ with respect to $\sigma$ is denoted by $F(D, \sigma) := \{X \subseteq \Im \mid support(X, D) \geq \sigma\}$.

*Problem 1.* (*Itemset Mining*) Given a set of items $\Im$, a transaction database $D$ over $\Im$, and minimal support threshold $\sigma$, find $F(D, \sigma)$.

An association rule is an expression of the form $X \Rightarrow Y$, where $X$ and $Y$ are itemsets, and $X \cap Y = \{\}$. $X$ is called the body or antecedent, and $Y$ is called the head or consequent of the rule. The support of an association rule $X \Rightarrow Y$ in $D$, is the support of $X \cup Y$ in $D$. An association rule is called frequent if its support exceeds a given minimal support threshold $\sigma$. The confidence of an association rule $X \Rightarrow Y$ in $D$ is the conditional probability $P(Y \mid X)$ : $confidence(X \Rightarrow Y, D) := \frac{support(X \cup Y, D)}{support(X, D)}$. The rule is called confident if $P(Y \mid X)$ exceeds a given minimal confidence threshold $\gamma$. The collection of frequent and confident association rules with respect to $\sigma$ and $\gamma$ is

$$R(D, \sigma, \gamma) := \{X \Rightarrow Y \mid X, Y \subseteq \Im, X \cap Y = \{\}, X \cup Y \in F(D, \sigma), confidence(X \Rightarrow Y, D) \geq \gamma\}.$$

*Problem 2.* (*Association Rule Mining*) Given a set of items $\Im$, a transaction database $D$ over $\Im$, and minimal support and confidence thresholds $\sigma$ and $\gamma$, find $R(D, \sigma, \gamma)$.

### B. Previous works

The main pillar of ARM-algorithms is Apriori [4]. It is the best-known algorithm to mine association rules, which uses a breadth-first search strategy to count the support of itemsets and uses a candidate generation function which exploits the downward closure property of support. Over the years, a lot of improvements of Apriori, supported with

 ISBN: 978-1-61208-162-5

different types of memory structures, are proposed.

Recent ARM-algorithms, based on graph mining can be roughly classified into two categories. The first category of algorithms employs a breadth-first strategy. Representative algorithms in this category include AGM [5] and FSG [6]. AGM finds all frequent induced sub-graphs with a vertex-growth strategy. FSG, on the other hand, finds all frequent connected sub-graphs based on an edge-growth strategy. Algorithms in the second category use a depth-first search for finding candidate frequent sub-graphs. A typical algorithm in this category is gSpan [7], which was reported to outperform both AGM and FSG in terms of computation time.

A different approach for association rule searching is used in ECLAT [8]. It is the first algorithm that uses a vertical data (inverted) layout. The frequent itemsets are determined using sets of intersections in a depth-first graph.

In graph ARM approaches the bottleneck is the necessity of performing many graph isomorphism tests. To overcome this problem, alternative approaches use hash-based techniques for candidate generation. The representatives in this direction are DHP [9] based on direct hashing and pruning, [10] which proposed the use of perfect hashing, and IHP [11] that uses inverted hashing and pruning.

FP-Tree [12], Frequent Pattern Mining is another milestone in the development of association rule mining, which breaks the main bottlenecks of the Apriori. FP-tree is an extended prefix-tree structure storing quantitative information about frequent patterns. The tree nodes are arranged in such a way that more frequently occurring nodes will have better chances of sharing nodes than less frequently occurring ones. The efficiency of FP-Tree algorithm has three reasons: (1) FP-Tree is a compressed representation of the original database; (2) it only scans the database twice; (3) it uses a divide and conquer method that considerably reduces the size of the subsequent conditional FP-Tree. The limitation of FP-Tree is its difficultness to be used in an interactive mining system, when a user wants to expand the dataset or change the threshold of support. Such changes lead to repetition of the whole mining process.

The Hmine algorithm [13] introduces the concept of hyperlinked data structure "Hyper structure" and uses it to dynamically adjust links in the mining process. Hyper structure is an array-based structure. Each node in a Hyper structure stores three pieces of information: an item, a pointer pointing to the next item in the same transaction and a pointer pointing to the same item in another transaction.

The innovation brought by TreeProjection [14] is the use of a lexicographical tree which requires substantially less memory than a hash tree. The number of nodes in its lexicographic tree is exactly that of the frequent itemsets. The support of the frequent itemsets is counted by projecting the transactions onto the nodes of this tree. This improves the performance of counting the number of transactions that have frequent itemsets. The lexicographical tree is traversed in a top-down fashion. The efficiency of TreeProjection can be explained by two main factors: (1) the transaction projection limits the support counting in a relatively small space; and (2) the lexicographical tree facilitates the management and counting of candidates and provides the flexibility of picking efficient strategy during the tree generation and transaction projection phrases.

Another data structure that is commonly used is a "trie" (or prefix-tree). Concerning speed, memory need and sensitivity of parameters, tries were proven to outperform hash-trees [15]. In a trie, every node stores the last item in the itemset it represents its support and its branches. The branches of a node can be implemented using several data structures such as hash table, binary search tree or vector.

Another algorithm for efficiently generating large frequent candidate sets, which use different data structures, is Matrix Algorithm [16]. The algorithm generates a matrix with entries 1 or 0 by passing over the cruel database only once, and then the frequent candidate sets are obtained from the resulting matrix. Finally association rules are mined from the frequent candidate sets. Experiment results confirm that the proposed algorithm is more effective than the Apriori.

This short overview of available algorithms and used structures shows the variety of decisions in association rule mining. As we can see graph structures, hash tables, different kind of trees, bit matrices, arrays, etc., are used for storing and retrieving the information.

Each kind of data structure brings some benefits and bad features. Such questions are discussed for instance in [17] where the comparison between tree-structures and arrays is made. Tree-based structures are capable of reducing traversal cost because duplicated transactions can be merged and different transactions can share the storage of their prefixes. But they incur high construction cost, especially when the dataset is sparse and large. Array-based structures incur little construction cost but they need much more traversal cost because the traversal cost of different transactions cannot be shared.

### *C. ArmSquare*

Here, we offer one approach, which is focused on proposing appropriate coding of the items in database in order to use the possibilities of direct access to the information via coordinate vectors into multidimensional numbered information spaces. This structure combines the convenience of the work with array structures with economy and performance of tree structures, which lies in the ground of realized access method. The algorithm of obtaining association rules is very simple; we focus our attention over the possibilities of using such structures for storing information in data mining systems. In future more smart algorithms can be realized using multidimensional numbered information spaces as storage data structures.

The proposed approach is realized in the module ArmSquare as a part of Data Mining Environment PaGaNe [18]. ArmSquare is aimed to make analysis and monitoring over the produced association rules from frequent datasets. The main data structures in PaGaNe use the advantages of specific model for organization of the storage of information, called Multidimensional Numbered Information Spaces [19]. The model is realized in the access method ArM 32. Let us mention the existing confusion of abbreviation ARM used in literature for short denotation of "association rule miner" and

 ISBN: 978-1-61208-162-5 

ArM, which means "Archive Manager". The name ArM was born in 1991 year (see [20]), two years before the defining of the association rule mining in [2]. The duplicating was used in the name of the realization: ArmSquare.

The rest of paper is organized in the following way. In Section 2, a brief description of Multidimensional Numbered Information Spaces is given. Sections 3 and 4 present our approach and program realization, which are based on the given possibilities for direct access to the points of multidimensional numbered information spaces. The differences between proposed algorithm and existing ones are discussed in Section 5. A short explanation of used databases from different fields is given in Section 6. Finally, some conclusions are highlighted.

## II. MULTIDIMENSIONAL NUMBERED INFORMATION SPACES

Following the Multi-Domain Information Model (MDIM), presented in [19] and realized by ArM 32, the elements are organized in a hierarchy of numbered information spaces with variable ranges, called ArM-spaces.

### A. Constructs

There exist two main constructs in MDIM – *basic information elements* and *numbered information spaces*. Basic information element is an arbitrary long sequence of machine codes (bytes). Basic information elements are united in numbered sets, called numbered information spaces of range 1. The numbered information space of range $n$ is a set, which elements are numerically ordered information spaces of range $n-1$. ArM32 allows using of information spaces with different ranges in the same file.

Every element may be accessed by correspond multidimensional space address (ArM-address) given by a coordinate array. The coordinate array is represented as numerical vector $A=(n,p_1,...,p_n)$, in which starting position shows the dimension of the space and next positions contains the coordinates of the points, thorough which the information can be reached. Sometimes, accounting the difference between the meaning of starting coordinate and next coordinates, the vector is written as $(n:p_1,...,p_n)$.

Another constructs, connected with MDIM are *indexes* and *metaindexes*. Every sequence of space addresses $A_1,A_2,...,A_k$, where $k$ is an arbitrary natural number, is said to be an *index*. Every index may be considered as basic information element and may be stored in a point of any information domain. In such a case, it will have a space address which may be pointed again. Every index which point only to indexes is said to be a *meta-index*.

Special kind of space index became the *projection*, which is analytical given index. There are two types of projections: (1) *Hierarchical projection* – in which the top part of coordinates is fixed and low part vary for all possible values of coordinates, where non-empty elements exist; and (2) *Arbitrary projection* – in this case is possible to fix coordinates in arbitrary positions and the rest coordinates vary for all possible values of coordinates, where non-empty elements exist.

### B. Operations

It is clear that the operations are closely connected to the defined structures. So, we have operations with:

- *basic information elements*: Because of the rule for existing of the all structures given above we have need of only two operations: updating and getting the value and two service operations: getting length and positioning in the element;
- *spaces*: With two spaces we may provide two operations: copying the first space in the second and moving the first space in the second with modifications specifying clearing or remaining the second space before operation;
- *indexes and meta-indexes*: Using the hierarchical projection we may crawl the defined area and extract next or previous empty or non-empty elements as well as to receive the whole index or its length, of the non-empty elements, which addresses fall into defined projection. The same operations (but only for non-empty elements) can be made for arbitrary projection. The operations between indexes are based on usual logical operations between sets. The difference from usual sets is that the information spaces are built by interconnection between two main sets: set of co-ordinates and set of information elements.

## III. ALGORITHM DESCRIPTION

We propose to use the abilities of multidimensional numbered information spaces for storage the information about interconnections between items and their combinations for facilitating association rule mining.

The proposed algorithm makes special analysis for each transaction and stores the frequency information in ArM-space, using the possibility of these spaces for accessing to the data via coordinate arrays. The algorithm consists of three phases: (1) pretreatment; (2) data processing and (3) analysis and monitoring.

### A. Pretreatment

In the pretreatment phase, the following steps are made:

- The transactions of the incoming dataset $D$ may be split into subsets, $D_b, b=1,...,d \quad \bigcup_{b=1}^{d} D_b = D$ by some condition (periods, regions, etc.). The mapping between the names of these groups and natural numbers $b=1,...,d$ is made;
- Creating a mapping between incoming items $i_j \in \Im$ and natural numbers $c_j \in \mathbb{N}$ by order of first occurrence of the item. This way if $\Im$ is a set of items, then $\overline{\Im}$ is also a set of items that contains the numbers from 1 to n ($n=|\Im|$), which code the items of $\Im$. Each incoming transaction $T=(tid,I), \ I=\{i_1,...,i_k\} \subseteq \Im$ is transformed into $\overline{T}=(tid,C), \quad C=\{c_1,...,c_k\} \subseteq \overline{\Im}$. The transaction database $D$ over $\Im$ is transformed to $\overline{D}$ over $\overline{\Im}$;
- The items in each transaction $\overline{T}$ are sorted in increasing order. The received transaction is denoted by $\vec{T}$. Ordering the items in the transaction has a great importance for the consequent steps.

*B. Data processing*

The intermediate phase is data processing. The data processing is closely depended on the length of the derived association rules. The greater the length, the more resources are needed. Because of this usually a special parameter $MaxK$ is used for limiting the maximum length of examined association rules. The algorithm traverse all combinations from 1 to $MaxK$. Let $k$ be the examined number of items $1 \le k \le MaxK$. For each transaction $\vec{T}$, $n = |\vec{T}| \ge k$ we make all possible combinations $Z^l = \{c_1^l, ..., c_k^l\}$, $Z \subseteq \vec{T}$, $l = 1, ..., \frac{n!}{k!(n-k)!}$. The element of Arm-spaces with co-ordinates $(c_1^l, ..., c_k^l)$ accumulates the number of occurrence of corresponded itemset $Z^l = \{c_1^l, ..., c_k^l\}$ (Fig. 1).

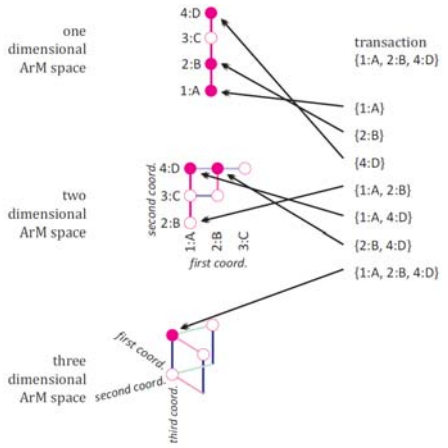


Figure 1. Accumulating in ArM spaces of the number of occurrence of produced itemsets from one transaction

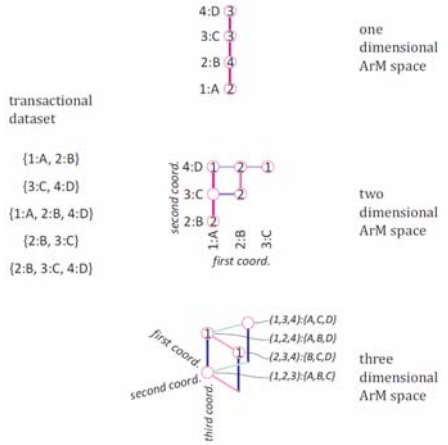


Fig. 2. Result of data processing of the database

In the case when $D$ is split in subsets $D_b, b = 1, ..., d$ additional coordinate in the space address is placed for marking the number of the group $b$ where transaction belongs to and the space address became following form $(b, c_1^l, ..., c_k^l)$.

As far as the processing over combinations with different lengths as well as subsets of database $D$ are independent, operations may be provided in parallel.

Finally, the support of the itemsets, which are driven from the transactions, is accumulated in the corresponded points in ArM-spaces (Fig. 2). Note that because of the ordering, not all coordinates in corresponded space are used. ArM 32 does not waste memory for empty points.

*C. Analysis and monitoring*

The analysis is made over the itemsets with particular length $k$, $1 \le k \le MaxK$ and using:

- A minimal support $\sigma$, which the itemsets to be included in the resulting list must have;
- A minimal confidence $\gamma$, which the association rules, created on the basis of the already selected itemsets, must have.

For obtaining all existing k-itemsets, whose support are at least $\sigma$, a traversal of all non-empty elements in a k-dimensional ArM-space is made. The coordinates of each element $(c_1^l, ..., c_k^l)$, which contains value, no less than $\sigma$, corresponds to itemset $Z^l = \{c_1^l, ..., c_k^l\}$, which is included in the resulting list $F(D, \sigma)$.

In the case where a database is split into groups, the traversal is made for each group taking into account that the first coordinate indicates the number of the group. The support of itemset $Z^l = \{c_1^l, ..., c_k^l\}$ for the whole database is received as a sum of values, contained in the points with corresponding coordinates in each group $(b, c_1^l, ..., c_k^l), b = 1, ..., d$.

One itemset $Z^l = \{c_1^l, ..., c_k^l\}$ is a source of producing several association rules.

Let $Z = \{c_1, ..., c_k\}$ be a k-itemset, $Z \in F(D, \sigma)$. The collection of association rules

$$R = \{X \Rightarrow Y \mid X, Y \subseteq \overline{\Im}, X \cap Y = \{\},$$
$$X \cup Y = Z, confidence(X \Rightarrow Y, D) \ge \gamma\}$$

is obtained by examining all possible combinations with length from 1 to k-1, which is given as a body of the rule $X^j = \{c_1^j, ..., c_p^j\}$, $p = 1, ..., k-1, X^j \subset Z$. The rest of the items forms the head of the rule $Y^j = Z \setminus X^j$. For an association rule $X^j \Rightarrow Y^j$:

- from the point with the space address $(c_1^j, ..., c_p^j)$, which correspond to $X^j$, the $support(X^j)$ is received. Taking into account that the body is part of an already existing itemset, this value is more than zero. The confidence of this association rule is calculated as $confidence(X^j \Rightarrow Y^j) = \frac{support(Z)}{support(X^j)}$;
- if $confidence(X^j \Rightarrow Y^j) \ge \gamma$, then $X^j \Rightarrow Y^j$ is included in the list of resulting association rules $R(D, \sigma, \gamma)$.

## IV. PROGRAM REALIZATION

The proposed algorithm was realized as analyzing tool in data mining environment PaGaNe.

### A. Input data

The system allows creation of a new database as well as adding new transactions to the same or different groups in an already existing database. During the input, the system accumulates the information for maximal length and average length of the transactions by each group separately. The repeated elements within the transaction are omitted.

Each transaction in the system is presented as numerical vector, with the length equal to the number of the elements, which participate in the transaction. The elements in the vectors are numbers, which correspond to the position of the element in dynamically expanded nomenclature, which contains the names of the items. Finally, these numbers are sorted. Sorting the elements in the vectors has a great importance for the consequent steps.

### B. Pretreatment in ArmSquare

Before starting the processing, one has to give a maximal number of combinations, which will be interesting – *MaxK*. Usually not all combinations between elements are interesting, but only a limited number of them – 2, 3, or 4, and no more than 10. The pretreatment performs a special analysis for each transaction in all groups and stores the frequency information in ArM-spaces, using the ability of the ArM-spaces for accessing the data via coordinate arrays. The user is interested in combinations from 1 to *MaxK*.

Let us trace the process for the transaction $\vec{T}$, which belongs to the group with number $b$. The length $n = |\vec{T}|$ varies depending on the numbers of the elements that were in the transaction. The system loops by $k$ from 1 to $\min(n, MaxK)$ in order to traverse all possible combinations. Each combination $\{c_1,...,c_k\}$ is used to form ArM-address for "k+1" dimensional space $(b, c_1,...,c_k)$ and at this point a support value is incremented by 1.

### C. Analysis and monitoring

The user can choose which length of itemsets he wants to observe. This length can vary between 2 and *MaxK*. Other parameters are minimal support and minimal confidence for a given database.

For obtaining all existing k-itemset with a support of at least $\sigma$, crawling over the k-dimensional ArM-space is done using the function *ArmNextProj*, starting with hierarchical projection $(-,-,...,-)$. In case of observing only a concrete group b, the crawling is made in k+1-dimensional ArM space with starting projection $(b,-,...,-)$. Using the function *ArmRead* for the current extracted non-empty element, the value is read and is compared with $\sigma$. If this value is no less than $\sigma$, the corresponded itemset is included into the resulting list of itemsets $F(D,\sigma)$. The resulting itemsets $F(D,\sigma)$ is sorted by decreasing support.

For receiving all association rules, created on the basis of itemsets from $F(D,\sigma)$, which have a confidence no less than $\gamma$, for each itemset $Z = \{c_1,...,c_k\}$ where $Z \in F(D,\sigma)$, every possible combination with a length from 1 to k-1, where $X^j = \{c_1^j,...,c_p^j\}$, $p = 1,...,k-1, X^j \subset Z$ is assumed as a body, while the rest of the items are taken as a head of the rule $Y^j = Z \setminus X^j$, is examined. For association rule $X^j \Rightarrow Y^j$:

- the value of $support(X^j)$ is received using function *ArmRead* from coordinate space address $(c_1^j,...,c_p^j)$, which correspond to the body $X^j$;
- the $confidence(X^j \Rightarrow Y^j) = \dfrac{support(Z)}{support(X^j)}$ is calculated;
- the association rule $X^j \Rightarrow Y^j$, whose confidence is no less than $\gamma$ is included in the list of resulting association rules $R(D,\sigma,\gamma)$.

Optionally, association rules can be sorted by decreasing confidence.

Using the ArM functions, the following additional operations, which can be used for analysis of the database, can be executed:

(1) Observation of all itemsets with given length k. For this purpose a function *ArmProjIndex* with hierarchical projection on the highest level in k-dimensional space is used. In the case of viewing the itemsets, belonging to a given group – the projection is one level lower in k+1 dimensional space with the highest coordinate equal to the number of the group fixed.

(2) Looking for k-itemsets, containing concrete item with given number $c$. A loop from 1 to k allows crawling the arbitrary projection (-,…,-,c,-,…,-), where position of c varies accordingly to the loop phase. A function *ArmProjIndex* uses these projections and extracts all non-empty elements, which define the corresponding itemset. The union of all these elements is the result of the request.

### D. Advanced specifics of ArmSquare

In Apriori algorithm min-support is set globally for combinations with different lengths. In our algorithm, after building the spaces, statistics for min-support for each area can be derived separately (the amount of space is equal to the number of elements in combinations), which allows to give for further analysis different min-support for different numbers of elements in combinations.

In a higher value of min-support Apriori is highly convergent and reaches a relatively short itemsets, where a small amount of min-support is close to total exhaustion of short itemsets.

Structuring the support of the itemsets in ArM-space allows subsequent analysis to be made very quickly by setting a different min-support and profiles of different lengths of itemsets, while other ARM-approaches derive all successive combinations in ascending order and changing the min-support causes a repetition of the whole algorithm.

 ISBN: 978-1-61208-162-5 

The information for itemsets with particular length containing a specific element can be directly extracted.

The database can be interactively expanded as well as the processing of the transactions can be made in parallel.

## V. IMPLEMENTATIONS

The realized tool allows different types of useful implementations in a wide spectrum of applications.

In one experiment we have used a retail market basket data set supplied by anonymous Bulgarian retail supermarket store. The data was collected over one year period (2008 year) from purchasing in a middle supermarket in a town with about 30 000 citizens. The total number of transactions was 108 846. The number of items were 3 609. The maximum length of the transactions was 23. The average items into transactions were 2.76 items per transaction. The transactions were grouped in accordance of months, when corresponded purchase was made. Several experiments were conducted with this dataset. Using all transactions, the most frequent combinations with 2, 3 and 4 length were extracted. Also there was used the possibilities of the ArmSquare to analyze different bins one to others and to extract the deviations of purchasing during the months.

Other experiments were made over a dataset that included several types of color harmonies and contrast features, extracted by 600 paintings of 19 artists from different movements of West-European fine arts and Eastern Medieval Culture. The pictures were obtained from different web-museums sources using ArtCyclopedia as a gate to the museum-quality fine art on the Internet. Each row of formed dataset contained the name of the artists, followed with harmonies and contrast features, presented in the manner of transactional dataset: "feature name"="value". Using the possibility of binning the dataset by class label allowed to use ArmSquare as element in the generation rule phase of CAR-algorithm and extract typical combinations of features for examined artists [21]. The constructed classifier, based on ArmSquare, outperforms classifiers with similar classification models (OneR, J48, JRip), realized in Weka.

## VI. CONCLUSIONS AND FUTURE WORKS

The main focus in the realization of ArmSquare is to show the possibilities to use the advantages of multi-dimensional information spaces for memory structuring in the area of data mining and knowledge discovery. The variety of the tasks that can be made with proposed frequent association rule miner ArmSquare allows comprehensive and facile analysis of the situations and conducting forecasting in wide areas of applications. The next steps will be focused on improving the algorithm of extracting rules, especially realizing the ARUBAS algorithm [22] over the multi-dimensional information spaces.

## ACKNOWLEDGMENT

This work was supported in part by Hasselt University under the Project R-1876 and by Bulgarian NSF under the project D002-308.

## REFERENCES

[1] Bodon, F., "A fast APRIORI implementation". In IEEE ICDM Workshop on FIMI, Melbourne, Florida, USA, 2003.

[2] Agrawal, R., Imieliński, T., and Swami, A., "Mining association rules between sets of items in large databases". In Proc. of the ACM SIGMOD ICMD, Washington, DC, 1993, pp. 207-216.

[3] Goethals, B., Efficient Frequent Pattern Mining. PhD thesis in Transnationale Universiteit Limburg, 2002.

[4] Agrawal, R. and Srikant, R., "Fast algorithms for mining association rules", In Proc. of the 20th Int. Conf. on VLDB, 1994, pp. 487-499.

[5] Inokuchi, A., Washio, T., and Motoda, H., "Complete mining of frequent patterns from graphs: mining graph data". In Machine Learning, Vol.50, 2003, pp. 321-354.

[6] Kuramochi, M. and Karypis, G., "Frequent subgraph discovery". In Proc. of the 1st IEEE Int. Conf. on DM, 2001, pp. 313-320.

[7] Yan, X. and Han, J., "gSpan: Graph-based structure pattern mining". In Proc. of the 2nd IEEE Int. Conf. on DM, 2002, pp. 721-724.

[8] Zaki, M., Parthasarathy, S., Ogihara, M., and Li, W., "New algorithms for fast discovery of assoctiation rules". In Proc. of the 3rd Int. Conf. on KD and DM, 1997, pp. 283-286.

[9] Park, J., Chen, M., and Yu, P., "An effective hash based algorithm for mining association rules". In Proc. of ACM SIGMOD Int. Conf. on Management of Data, 24/2, 1995, pp. 175-186.

[10] Özel, S. and Güvenir, H., "An algorithm for mining association rules using perfect hashing and database pruning". In Proc. of the TAINN, 2001, pp. 257-264.

[11] Holt, J. and Chung, S., "Mining association rules using inverted hashing and pruning". Information Processing Letters Archive, 83/4, 2002, pp. 211-220.

[12] Han, J. and Pei, J., "Mining frequent patterns by pattern-growth: methodology and implications. In ACM SIGKDD Explorations Newsletter 2/2, 2000, pp. 14-20.

[13] Pei, J., Han, J., Lu, H., Nishio, S., Tang, S., and Yang, D., "Hmine: hyper-structure mining of frequent patterns in large databases". In Proc. of IEEE ICDM, 2001, pp. 441-448.

[14] Agarwal, R., Aggarwal, C., and Prasad V., "A tree projection algorithm for generation of frequent item-sets". In Journal of Parallel and Distributed Computing, 61/3, 2000, pp. 350-371.

[15] Bodon, F. and Ronyai, L., "Trie: an alternative data structure for data mining algorithms". In Mathematical and Computer Modelling, 38/7, 2003, pp. 739-751.

[16] Yuan, Y. and Huang, T., "A Matrix algorithm for mining association rules". In LNCS, Vol. 3644, 2005, pp. 370-379.

[17] Liu, G., Lu, H., Yu, J., Wang, W., and Xiao, X., "AFOPT: An efficient implementation of pattern growth approach". In Workshop on Frequent Itemset Mining Implementation. (FIMI 03), 2003.

[18] Mitov, I., Ivanova, K., Markov, K., Velychko, V., Vanhoof, K., and Stanchev, P., "PaGaNe – a classification machine learning system based on the multidimensional numbered information spaces". In WSPS on CEIS, No. 2, 2009, pp. 279-286.

[19] Markov, K., "Multi-domain information model". In Int. J. on Information Theories and Applications, 11/4, 2004, pp. 303-308.

[20] Markov K., Ivanova, K., Mitov, I., and Karastanev, S., "Advance of the access methods". Int. J. on Information Technologies and Knowledge, 2/2, 2008, pp. 123-135.

[21] Ivanova K., Stanchev P., and Vanhoof K., "Automatic tagging of art images with color harmonies and contrasts characteristics in art image collections". Int. J. on Advances in Software, 3/3&4, 2010, pp. 474-484.

[22] Depaire, B., Vanhoof, K., and Wets, G., "ARUBAS: an association rule based similarity framework for associative classifiers". In IEEE Int. Conf. on Data Mining Workshops, 2008, pp. 692-699.

 ISBN: 978-1-61208-162-5 