AN EFFICIENT METHOD FOR POST ANALYSIS OF PATTERNS

Rein Kuusik, Innar Liiv and Grete Lind
Tallinn University of Technology, Department of Informatics
Raja 15, 12618 Tallinn
Estonia
kuusik@cc.ttu.ee, innar.liiv@ttu.ee, grete@cc.ttu.ee

ABSTRACT
Finding and extracting frequent patterns is one of the most important tasks in data mining, therefore various algorithms have been introduced over time. Unfortunately, when the sizes of datasets increase, completely different and new problems arise. Even if we are able to extract the IF…THEN rules in a reasonable time, it is possible that the algorithms will find millions of patterns. Interpretation of all of them would be a grievous baffling problem for even a team of analysts. In this paper we describe a method we have used for post-analysis of patterns. The basic idea is presented with an example and the rules for result transformation are given, making it possible to apply standard querying tools. Although we have implemented it as an extension to generator of hypotheses, it would also give reasonable results with other rule extracting methods.

KEY WORDS
Pattern post analysis, data mining, generator of hypotheses, and monotone system theory

1. Introduction

According to [1], data mining (DM) is a part of the process called knowledge discovery in databases (KDD), which consists of particular data mining algorithms and produces a particular enumeration of patterns.

Pattern is an expression (in a certain language) describing facts in a subset of facts.

Data mining has two high-level primary goals:

- **Prediction** involves using some variables or fields in the database to predict unknown or future values of other variables of interest.
- **Description** focuses on finding human-interpretable patterns describing the data.

In the context of KDD, description tends to be more important than prediction. These goals are achieved by using primary DM tasks as classification, regression, clustering, summarization, dependency modeling, change and deviation detection.

On the basis of these DM tasks several DM solutions have been developed [2][3]. In this paper we present a method for post analysis of patterns and implement it as an extension of a DM method called Generator of Hypotheses, which is based on monotone systems. Therefore, after describing the method in the second section, we will also describe the algorithm MONSA – what is a monotone system, how to create and use monotone systems and an application called Generator of Hypotheses (GH). In the fifth section we describe the results of the method implementation.

2. Description of the method

Instead of printing the result as a tree or list of rules, we make another table, where the first column is rule number (*rule_id*), second column is vertical coverage of the rule (how many rows does the rule cover; *ch*) and the third column is horizontal coverage of the rule (how may attributes are used to describe the rule; *cw*) and then all the attributes.

The first column is filled incrementally, second one is given by rule (for this method it would be better, if the algorithm outputs not percentage, but the real number of rows the rule covers) and the third one is simply by counting the attributes that are used to describe the rule. Attribute columns are filled with NULL value, if the attribute does not exist in the rule and with the value of attribute if it exists in the rule.

It gives possibility to an analyst to see rules distribution by ch (coverage) or by cw (number of attributes in rule) and to choose the roles by ch or/and cw for analysis. It is a process built on MONSA for administrating roles choosing process. GH is used to build a tree for chosen set of rules.

After describing the algorithm MONSA for context understanding, we will demonstrate in the fifth section the results of our method's implementation on analysing the extracted patterns from our generator of hypotheses.

3. Algorithm MONSA

In this section we describe a pattern mining algorithm, named MONSA (MONotone System Algorithm) we have used for finding rules and the concept of monotone systems [4].

3.1 What is a monotone system?

Definition 1:
Let a finite discrete *set* X and function π_x on it which maps to each *element* $\alpha \in X$ a certain nonnegative number $\pi_x(\alpha)$, be given.
The function π_x is called a *weight function* if it is defined on any subset $X' \subseteq X$; the number $\pi_x(\alpha)$ is called a *weight* of element α on X'.
Definition 2:
A set X with a weight function π_x is called a *system* and is denoted by $S=(X, \pi_x)$.
Definition 3:
The system $S'=(X', \pi_{x'})$ where $X' \subseteq X$ is called a *subsystem* of the system $S=(X, \pi_x)$.
Definition 4:
The system $S=(X, \pi_x)$ is called *monotone* if in the case of any $\alpha \in X' \setminus \{b\}$, $b \in X$:
$\pi_{x'\setminus\{b\}}(\alpha) \le \pi_{x'}(\alpha)$ where X' is any subset of X.

3.2 How to create and use a monotone system

Let be given data set X(N,M), where N is the number of objects (i=1,…,N) and M is the number of attributes (j=1,…,M). Element α can be X_{ij}, row i, column j or any sub table of X.
To use the method of monotone systems we have to fulfil two conditions:

1. There has to be a weight function $\pi_x(\alpha)$ which will give a measure of influence for every element α of the monotone system on X;
2. Certain activities (adding or removing) can be applied to the elements. There have to be rules f to recompute the weight of the system elements after used activities. Weights can be changed only to one direction (increasing or decreasing).

These conditions give a lot of freedom (to user) to choose the weight functions and rules of weight change in the system. The only constraint we have to keep in mind is that after eliminating all elements α from the system X the final weights of $\alpha \in X$ must be equal to zero. In our case:

1. A suitable weight function is object's frequency in a (concerned) system. In case of tables of object-attribute type we weigh the attribute's value and the weight is a number of objects having that certain value.
2. Rules for recomputing the weights:

- Choose the element(s) of interest.
- Extract the objects having the(se) element(s) from the concerned set. So the set of objects under consideration can only decrease.
- For the rest of objects calculate new weights using the same weight function. If there are no objects with given elements then the weight is zero.

3.3 Description of MONSA

MONSA finds patterns in given set X(N,M), where N is the number of objects (for example transactions), M is the number of attributes and each attribute j has an integer value h_j=0,1,2,…,K-1. Pair of attribute and its certain value is called element.
By essence MONSA is a recursive algorithm. Here its backtracking version is presented.
The denotations used in this algorithm are described on the top of the Figure 1.

t	the number of the step (or level) of the recursion
FT_{t+1}	frequency table for a set $X_{t+1} \subset X_t$
$Pattern_t$	vector of elements 'attribute.value' over set X_t (for example, A1.1 (A1 value equals 1))
Init	activity for initial evaluation

```
Init
t=0, $Pattern_0=\{\}$
To find a table of frequencies $FT_0$ for all attributes in $X_0$
DO WHILE there exists $FT_s \# \emptyset$ in $\{FT_s\}$, $s \le t$
   FOR an element $h_f \in FT_t$, $1 \le f \le M*K$ with frequency
                                          $V=\max FT_t(h_f) \# 0$ DO
       To separate submatrix $X_{t+1} \subset X_t$ such that
             $X_{t+1}=\{X_i \in X_t;\ i=1,\dots,N_t \mid X(i,f)=h_f\}$
       To find a table of frequencies on $X_{t+1}$
       Attributes j values $h_j$, j=1,…,M with $FT_{t+1}(h)=V$
                                                   form $Pattern_{t+1}$
       FOR j=1,…,M, $h_j$=0,…,K-1 DO
          IF $FT_t(h_j,j)=0$, THEN
             $FT_{t+1}(h_j,j)=0$
          ENDIF
          IF $FT_{t+1}(h_j,j)=V$ THEN
             $FT_t(h_j,j)=0$
             $FT_{t+1}(h_j,j)=0$
          ENDIF
          IF $FT_{t+1}(h_j,j)=FT_t(h_j,j)$ THEN
             $FT_t(h_j,j)=0$
          ENDIF
       ENDFOR
       IF there exist attributes to analyse THEN t=t+1
       Output of $Pattern_t$
   ENDFOR
   t=t-1
ENDDO
All patterns are found
END
```

Figure 1. Algorithm MONSA

The main idea of the work of MONSA is simple:

- subset $X_{t+1} \subset X_t$ of objects with certain properties is being separated;
- then pattern over this subset X_{t+1} ($IntSec_{X_{t+1}}$) is being found.

This is a depth-first search algorithm, it uses frequency tables and special techniques to prevent repetitions. Also, it works with any set of discrete values (not only 0 and 1).

For finding patterns the frequency table is used. Element's frequency is the number of times the element (i.e. attribute with its certain value) occurs in given data. The frequency table consists of counts of occurrences of elements in the set or subset of data.

In examples the following table of object-attribute type (Table 1) will be used. It contains a subset of data given in [5], only the objects with A4.1 (A4 has value 1) are chosen, attribute A4 is not shown in the table. The corresponding frequency table is given in Table 2.

Table 1. Initial table

Object \ Attribute	*A1*	*A2*	*A3*
O1	2	1	1
O2	1	1	1
O3	2	3	2
O4	1	3	2
O5	2	1	2

Table 2. the corresponding frequency table

Value \ Attribute	*A1*	*A2*	*A3*
1	2	3	2
2	3	0	3
3	0	2	0

For example, from value/attribute table we can see that A1.1 appears 2 times in the initial table, A1.2 - 3 times, A1.3 doesn't appear, A2.1 – 3 times etc.

To find a pattern on the set X_{t+1}, the frequency table FT_t of the set X_t can be used effectively. Maximal frequency $MAX=|X_{t+1}|$ of a certain element $Xij \in X_{t+1}$ in frequency table FT_{t+1} is defined by the frequency in FT_t of pattern which was the base to separate X_{t+1}. Consequently, all elements $Xij(=h_j) \in X_{t+1}$ the frequencies of which equal MAX, appear simultaneously in all objects of X_{t+1} and define a pattern over the set X_{t+1}.

To prevent repetitions we use nullifying techniques. Zero in FT means that this value is not in analyze. Bringing zeroes down (from FT_t to FT_{t+1}) prohibits arbitrary output repetition of already separated pattern on level (t+1). Bringing zeroes up (from FT_{t+1} to FT_t) does not allow the output of the separated pattern on the same (current) level t+1 and on steps t, t-1, ..., 0.

3.4 Results of MONSA

(With minimal frequency allowed = 2) MONSA finds from our data (see Table 1) eight patterns (see Figure 2). All of them are different, and they are not empty. And there was no special effort to check their uniqueness during the extraction process. Repetitions were prevented just by using nullifying techniques.

	Set	Pattern
1.	O1, O3, O5	A1.2 =3
2.	O1, O5	A1.2&A2.1 =2
3.	O3, O5	A1.2&A3.2 =2
4.	O1, O2, O5	A2.1 =3
5.	O1,O2	A2.1&A3.1 =2
6.	O3, O4, O5	A3.2 =3
7.	O3, O4	A3.2&A2.3 =2
8.	O2, O4	A1.1 =2

Figure 2. Patterns found by MONSA

4. Application of MONSA – Generator of Hypotheses, a Method for Knowledge Discovery

Generator of Hypotheses (GH) is a method for data mining which main aim is mining for patterns and association rules [6] [7] [8]. The goal is to describe the source data. Used evaluation criteria are deterministic (not probabilistic). The association rules it produces are represented as trees, which are easy to comprehend and interpret.

4.1 Output of Generator of Hypotheses

By depth-first search (from root to leaves) GH forms a hierarchical grouping tree. Such tree got with our example (see Table 1) is given on Figure 3.

The meanings of used values are given in Table 5.

```
(3)     0.667(2)  0.500(1)
Height.2=>Hair  .1->Eyes  .1
                  0.500(1)
                  ->Eyes  .2
        0.667(2)  0.500(1)
        =>Eyes  .2->Hair  .3

 (3)     0.667(2)  0.500(1)
Hair  .1=>Eyes  .1->Height.1
        0.333(1)
        =>Eyes  .2

 (3)     0.667(2)  0.500(1)
Eyes  .2=>Hair  .3->Height.1
```

Figure 3. Fragment of hierarchical grouping tree for data from table 1

Table 3. Meanings of used attributes' values

Attribute	Attribute's value	Value's meaning
Height (A1)	1	short
Height (A1)	2	tall

Hair (A2)	1	dark
Hair (A2)	3	blond
Eyes (A3)	1	blue
Eyes (A3)	2	brown

The numbers above node show node's absolute frequency (in parentheses) and node's relative (to previous level) frequency (before parentheses).

Absolute frequency of node t shows how many objects have certain attribute with certain value (among objects having properties (i.e. certain attributes with certain values) of all previous levels t-1,…,1). Relative frequency is a ratio A/B, where A is the absolute frequency of node t and B is the absolute frequency of node t-1. For the first level the relative frequency is not calculated.
For example we can translate the first tree (Height.2=>) of set of trees as "3 persons (objects/examples) are tall (Height.2), 67% of them have dark hair (Hair.1), and of those (with Height.2 and Hair.1) 50% have blue eyes (Eyes.1) and 50% have brown eyes (Eyes.2). Also, 67% of tall persons (Height.2) have brown eyes (Eyes.2) and 50% of those have blond hair (Hair.3)."

4.2 Properties of Generator of Hypotheses

GH has the following properties:

- GH guarantees immediate and simple output of rules in the form IF=>THEN
- GH enables larger set of discrete values (not only binary);
- GH enables to use several pruning techniques;
- The result is presented in form of trees;
- GH enables to treat large datasets;
- GH enables sampling.

5. Implementation of the post analysis method

For the implementation of the method (see section 2) we only needed to transform the results in the way described in the second section. In order to use standard tools, we actually transformed the results (the format is shown on Figure 3) into SQL INSERT queries, that would construct us the coverage and rule table described in the second section. Used dataset was more complex than in previous examples. The specific (although the method itself is not platform dependent) SQL server was MySQL 4.0.17 on FreeBSD 4.9.

Next four sample queries, that very often gave us different interesting results, will be introduced. The denotations used are as: ch – the number of rows covered, cw – the number of columns covered, A*n* – attribute *n*.

1. The distribution of the coverage ratios (see Figure 4)

```
SELECT   ch,cw,COUNT(*)   FROM   monsys   GROUP   BY
ch,cw;
+------+------+----------+
| ch   | cw   | COUNT(*) |
+------+------+----------+
...
|   10 |    4 |        4 |
|   10 |    5 |      319 |
|   10 |    6 |     3001 |
|   10 |    7 |    10726 |
|   10 |    8 |    15049 |
|   10 |    9 |    11264 |
|   10 |   10 |     4301 |
|   10 |   11 |      432 |
|   10 |   12 |       45 |
...
|   13 |   10 |        1 |
|   14 |    3 |       45 |
|   14 |    4 |      919 |
|   14 |    5 |     3038 |
|   14 |    6 |     2279 |
|   14 |    7 |      589 |
|   14 |    8 |       20 |
|   14 |    9 |        2 |
...
+------+------+----------+
```

Figure 4. Sample query #1

The second row of the table on Figure 4 shows that there are 319 rules with width 4 (i.e. consist of 4 attributes) giving a group of exactly 10 objects.
Grouping rules by ch (the number of rows/objects covered) shows how many attributes are needed to get the coverage of certain number of the objects. Also it shows that these numbers of attributes are different for different object coverage desired.
On the figure 4 there are results for groups with size of 10 and 14 objects. Both groups (by ch) have a normal distribution. We can see that groups of 10 objects are mostly described by 7 to 9 attributes; groups of 14 objects are described by 5 or 6 attributes. We can see the main trend: the larger groups are described by less number of attributes.
Grouping the results by cw (the number of columns/ attributes covered) we would know the size of object coverage given by specific length of rule.
Of cource these distributions are dataset-specific.

2. K best rules by the coverage (ch*cw) , K is given by the analyst. Figure 5 shows 3 best rules by such coverage. It is also possible to give a certain treshold for (ch*cw).

```
SELECT * FROM monsys ORDER BY (ch*cw) DESC LIMIT
3;
+----------+------+------+-------+----+----+-...
| rule_id  | ch   | cw   | A1    | A2 | A3 | ...
+----------+------+------+-------+----+----+-...
|       10 |   13 |   10 |  NULL |  7 |  5 | ...
|     2127 |   10 |   13 |     3 |  0 |  1 | ...
|        9 |   14 |    9 |     1 |  9 |  4 | ...
+----------+------+------+-------+----+----+-...
```

Figure 5. Sample query #2

Next queries select rules for which ch and cw are equal, i.e the number of attributes describing the rule is the same as the number of objects covered by the rule. Those patterns give possibly interesting extra information for the analyst.

3. Total number of rules with equal ch and cw (see Figure 6) – this often gives us some kind of first glance of the dataset.

```
SELECT COUNT(*) FROM monsys WHERE ch=cw;
+----------+
| COUNT(*) |
+----------+
|    32871 |
+----------+
```

Figure 6. Sample query #3

4. The distribution of the patterns that have equal number of attributes describing it and number of objects covered (see Figure 7)

```
SELECT  ch,cw,COUNT(*)  FROM  monsys  WHERE  ch=cw
GROUP BY ch,cw ORDER BY ch;
+------+------+----------+
| ch   | cw   | COUNT(*) |
+------+------+----------+
|    6 |    6 |        9 |
|    7 |    7 |      884 |
|    8 |    8 |    10298 |
|    9 |    9 |    17331 |
|   10 |   10 |     4301 |
|   11 |   11 |       48 |
+------+------+----------+
```

Figure 7. Sample query #4

6. Conclusion

In this paper we presented a new method for post analysis of patterns. It has been implemented as an extension to generator of hypotheses, but it is possible to use it with other algorithms and applications as well. The main importance is that it will help us analyze the results more specifically, not just change the threshold and rerun the whole process. If we transform the extracted patterns in a way proposed in this paper, it will allow us to make ad hoc queries from the extracted patterns. Using "width" of a pattern in addition to "height" gives us extra information about the pattern and could help us to find the patterns we are interested in.

The future work includes making this process semi-automatic, instead of manual querying. Also many integrated visualization methods are to be developed, as it should have a substantial role in post analysis.

References:

[1] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From Data Mining to Knowledge Discovery: An Overview. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining* (AAAI Press/ The MIT Press, 1996), 1-36.

[2] M. H. Dunham, *Data Mining: Introductory and Advanced Topics* (Prentice Hall, 2002).

[3] T. Hastie, R. Tibshirani, J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer Series in Statistics (Springer Verlag, 2001).

[4] I. Mullat, Extremal monotone systems, *Automation and Remote Control (in Russian)*, 1976, 5, 130-139; 8, 169-178.

[5] J. R. Quinlan, Learning efficient classification procedures and their application to chess and games. In J. G. Carbonell, R. S. Michalski, T. M. Mitchell (Eds.), *Machine Learning. An Artificial Intelligence Approach* (Springer-Verlag, 1984).

[6] R. Kuusik, The Super-Fast Algorithm of Hierarchical Clustering and the Theory of Monotone Systems, *Transactions of Tallinn Technical University*, 734, 1993, 37-62.

[7] R. Kuusik, G. Lind, An Approach of data mining using monotone systems, *Proceedings of 5th International Conference on Enterprise Information Systems, Vol. 2*, Angers, 2003, 482-485.

[8] I. Liiv, Mining Association Rules Using the Theory of Monotone Systems, *M.Sc. Thesis at Tallinn University of Technology*, 2004.