
Improving the Quality of Association Rule Mining by Means of Rough Sets

Daniel Delic, Hans-J. Lenz, and Mattis Neiling

Free University of Berlin, Institute of Applied Computer Science,
Garystr. 21, D-14195 Berlin, Germany

Summary. We evaluate the rough set and the association rule method with respect to their performance and the quality of the produced rules. It is shown that despite their different approaches, both methods are based on the same principle and, consequently, must generate identical rules. However, they differ strongly with respect to performance. Subsequently an optimized association rule procedure is presented which unifies the advantages of both methods.

1 Introduction

Data mining methods are part of knowledge discovery from databases. Its objective is to extract nontrivial, beforehand unknown and potentially useful information. There exist several data mining procedures which differ in their methodology as well as in their data types.

In our work we focus on a comparison of the *association rules* with the *rough sets*. The association rule method was developed particularly for the analysis of large databases, whose attributes possess only Boolean values. The rough set method, however, investigates databases. Its attributes can possess several values with the constraint, that a predefined set of attributes must exist on which the generated rules are based on. In order to be able to make a fair comparison, both procedures should operate on the same data type. Our aim is to find out, which method is more efficient, and whether a combination of both methods improves the overall quality of unscrambled rules.

2 Association Rules

The association rule method was developed originally for the analysis of large databases of customer transactions. Each transaction consists of items purchased in a supermarket. In order to apply this method to data multi-value attributes, we defined the term *item* in a different way.

2.1 Transforming Attributes into Items

An item combines the name (label) of an attribute with one of its possible values. Each attribute has a finite set of values which is called domain.

Within a database each combination of attribute name and attribute value is distinguishable. Consequently, there exists a set $A = \{a_1, a_2, a_3, \dots, a_p\}$ with $p \in N$ items in a database.

Example 1. To represent an attribute with binary domain we only need one item $a_i \in \{0, 1\}$. For example, for attribute ‘spaghetti’ with $\text{domain}(\text{spaghetti}) = \{\text{bought}, \text{notbought}\}$ ($i = 1$) we set $a_1 = 1$ if spaghetti are bought (see Fig. 1).

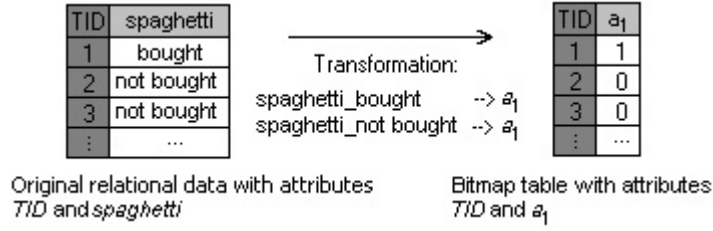


Fig. 1. Transformation of relational data into an efficient bitmap representation for attributes with binary domains.

Example 2. For an attribute with no-binary domain, each attribute value corresponds to one item. For example, for attribute ‘blood_pressure’ with $\text{domain}(\text{blood_pressure}) = \{\text{high}, \text{low}, \text{normal}\}$ ($i = \{1, 2, 3\}$) the following items result: $a_1 = \text{“blood_pressure_high”}$, $a_2 = \text{“blood_pressure_low”}$ and $a_3 = \text{“blood_pressure_normal”}$ (see Fig. 2).



Fig. 2. Transformation of relational data into an efficient bitmap representation for attributes with no-binary domains.

Finally an association rule can be defined as follows: Given the subsets $X \subset A$ and $Y \subset A$ with $X \cap Y = \emptyset$, an association rule is an implication in the form $X \rightarrow Y$, with a confidence c ($0 \leq c \leq 1$) and a support s ($0 \leq s \leq 1$).

The confidence is a measure of how many tuples that contain X also contain Y . The support is the relative frequency of the set $X \cup Y$.

2.2 Data Structure ‘Bitmap’

It is known from Database Theory that bitmaps are efficient data structures (Jürgens/Lenz, 2001). Therefore the original relational schema has to be transformed into a bitmap table. The first column holds the primarykey attribute TID , each of the remaining columns contain a binary bitmap-attribute i.e. an item from the original data set (see Fig. 1 and 2). This structure is efficient, because the column-address of a bitmap-attribute doesn’t change. Therefore it is possible to check very fast each tuple, by direct access for the appropriate column, whether the searched item is or is not available in the corresponding tuple of the original data set. If its present there is a ‘1’, otherwise a ‘0’ in the bitmap.

2.3 Rules Derivation

The procedure to generate association rules is based on the multi-pass-algorithm *Apriori*¹. The principle idea of this algorithm is to scan several times a database while searching for sets of items that occur in a sufficient large number. After each pass the number of items in those “large sets” is increased by one until all existing sets in the database are found. Subsequently the association rules are derived from these item sets.

3 Rough Sets

The rough set theory was introduced by Zdislaw Pawlak². It is a method for uncovering dependencies in data, which are recorded by relations. A detailed introduction to rough set theory can be found in *Munakata* (1998).

3.1 Model

The rough set method operates on data matrices, so called “information tables” (see Table 1). It contains data about the *universe* U of interest, *condition attributes* and *decision attributes*. The goal is to derive rules that give information how the decision attributes depend on the condition attributes.

A prerequisite for rule generation is a partitioning of U in a finite number of blocks, so called equivalence classes, of same attribute values by applying an equivalence relation. For example, the equivalence relation $R_1 = \{(u, v) | u(\text{blood pressure}, \text{temperature}) = v(\text{blood pressure}, \text{temperature})\}$

¹ See Agrawal/Srikant (1994).

² See Pawlak (1982).

Table 1. Information table.

universe <i>person</i>	condition attributes <i>temperature</i> <i>blood pressure</i>		decision attribute <i>heart problem</i>
Adams	normal	low	no
Brown	normal	low	no
Carter	normal	medium	jes
Ford	high	high	jes
Gill	high	high	no
Bellows	medium	medium	jes

leads to a partition of U into three equivalence classes $U_1 = \{Adams, Brown\}$, $U_2 = \{Carter\}$ and $U_3 = \{Ford, Gill\}$ (see Table 1). Given these classes, rules like e.g. “If temperature *normal* and blood pressure *low* then heart problem *no*.” can be derived. Generally, no unique rule derivation is possible. For example, *Ford* and *Gill* have identical values of the condition attributes, but differ in their values of the decision attribute. In order to analyze such data, the concept of approximation spaces is used to determine the confidence of the derived rules.

The quality of the extracted rules depends also strongly on the possibility of attribute reduction. Reducing the number of attributes in a dataset by removing the redundant ones is one of the main objectives of rough set theory and at the same time one of the main problems. Finding a minimal reduct is a NP-hard³ problem. Finding all reducts has exponential complexity⁴.

In our approach we try to reduce the computing time by applying the concept of reduct extraction directly to the produced rules, not to attributes. First, in order to generate strong rules, all rules which support and confidence values don’t reach the given minimum threshold, are deleted. Second, the reducts are extracted. Suppose, there are two rules with same decision item and same confidence value, and their only difference is the set of condition items. The condition itemset of rule no 1 is a subset of the condition itemset of rule no 2. In this special case rule no 2 is redundant and can be deleted without having loss of information.

3.2 Rough Set Based Rule-Generation with *RS-Rules+*

Besides the “classic” version of rule generation with a fixed decision attribute (cf. Table 1), our algorithm *RS-Rules+* offers the possibility of varying the selected decision attributes of a table. Each attribute of the table will be included either as a decision or condition attribute. *RS-Rules+* is explained on the basis of Table 2: The attributes of the original data set are $\{A\}$, $\{B\}$ and

³ See Rauszer (1991).

⁴ See Skowron/Rauszer (1992).

$\{C\}$. Each attribute has two non *null* values $A = \{a_1, a_2\}$, $B = \{a_3, a_4\}$ and $C = \{a_5, a_6\}$, so that there are six items (bitmap-attributes) for the resulting bitmap-table: $\{a_1\}, \{a_2\}, \{a_3\}, \{a_4\}, \{a_5\}, \{a_6\}$. In the first step all possible rules are constructed from all bitmap-attributes of the table. For example, for $\{A\} \rightarrow \{B\}$ four item-related rules can be set up: $\{a_1\} \rightarrow \{a_3\}, \{a_1\} \rightarrow \{a_4\}, \{a_2\} \rightarrow \{a_3\}$ and $\{a_2\} \rightarrow \{a_4\}$. All rules not fulfilling the minimum support and minimum confidence are deleted. The condition items from the remaining rules ($\{a_1\}$ and $\{a_4\}$), build pairwise the condition itemsets for the next run⁵ (i. e. $\{a_1, a_4\}$). Again all possible rules are produced, their support and confidence is measured and the ‘weak’ ones are deleted. The set of the remaining rules ($(\{a_1, a_4\} \rightarrow \{a_5\})$) is compared with the set of produced rules in the previous run in order to remove the unnecessary ones.

Suppose, $\{a_1, a_4\} \rightarrow \{a_5\}$ has the same confidence as $\{a_4\} \rightarrow \{a_5\}$. In this case $\{a_1, a_4\} \rightarrow \{a_5\}$ is dominant and will be deleted. Since there are no more rules left to extract from the itemsets of the next generation, the algorithm stops.

In case that $\{a_1, a_4\} \rightarrow \{a_5\}$ is necessary it is stored as a valid rule and its condition itemset is part of the basis for next generation condition itemsets. Since there is no other new rule left, no new condition itemset(s) can be setup and the algorithm stops.

Table 2. Rule-Generation with *RS-Rules+*.

	1st run	2nd run
New condition attribute-sets	A = {a ₁ , a ₂ } B = {a ₃ , a ₄ } C = {a ₅ , a ₆ }	{a ₁ , a ₄ }
Possible Rules	{a ₁ } → {a ₃ } {a ₁ } → {a ₄ } {a ₁ } → {a ₅ } {a ₁ } → {a ₆ } {a ₂ } → {a ₃ } {a ₂ } → {a ₄ } {a ₂ } → {a ₅ } {a ₂ } → {a ₆ } {a ₃ } → {a ₅ } {a ₃ } → {a ₆ } {a ₄ } → {a ₅ } {a ₄ } → {a ₆ }	{a ₁ , a ₄ } → {a ₂ } {a ₁ , a ₄ } → {a ₃ } {a ₁ , a ₄ } → {a ₅ } {a ₁ , a ₄ } → {a ₆ }
Rules with minsupport (valid rules)	{a ₁ } → {a ₃ } {a ₄ } → {a ₅ }	{a ₁ , a ₄ } → {a ₅ }
Condition attribute-sets with minsupport	{a ₁ }, {a ₄ }	{a ₁ , a ₄ }

⁵ The formation of condition itemsets is based on the same principle as the *k*-itemset construction in 2.3.

4 Comparison of the Procedures

The transformation of the underlying data set into a bitmap-representation and the modification of the rough set rules induction scheme allow an objective comparison of both methods. Three benchmark data sets have been selected, which differ in size and number of attributes⁶:

1. *Car Evaluation Database*: 1728 tuples, 25 Boolean attributes in the bitmap-table.
2. *Mushroom Database*: 8416 tuples, 12 original attributes selected, 68 boolean attributes in the bitmap-table.
3. *Adult*: 32561 tuples, 12 original attributes selected, 61 boolean attributes in the resulting bitmap-table.

The computing times of the rough set algorithm for the benchmark data were between 2 minutes and 4 hours, those of the association rule algorithm between 40 seconds and 45 minutes⁷ (c.f. Table 3). Both procedures supplied similar results for all examined tables, except for few cases due to reducts of the rough set algorithm⁸. However, not creating redundant rules does not lead to any principal difference of rule production. Hence we conclude that both procedures are equivalent with respect to the produced rules.

Theorem. Let $c, s \in (0, 1]$ values for minimum confidence and minimum support. Then the algorithms *Apriori* and *RS-Rules+* induce identical rules from given data D w.r.t. c and s if the extraction of reducts by *RS-Rules+* does not take place.

Proof. Let $X \rightarrow Y$ be any rule derived with *RS-Rules+* and S the approximation space for Y . S is spanned by all the items, which are involved in X . The restriction of S on the lefthand-side X of a rule selects from D exactly those records which inherit a condition itemset equal to X . For this set D_X of the extracted records $\frac{|D_X|}{|D|} \geq s$ holds. Since only those rules are derived by *RS-Rules+*, which fulfill the minconfidence c and the minsupport s , both criterions are satisfied for the set $D_{X \cup Y} \subset D_X$. Hence the *Apriori*-algorithm will induce this rule also.

Conversely, let $X \rightarrow Y$ be any rule derived with *Apriori*. Then $D_{X \cup Y}$ the set of all records inheriting the large itemset $X \cup Y$ fulfills the inequalities $\frac{|D_{X \cup Y}|}{|D|} \geq s$ and $\frac{|D_{X \cup Y}|}{|D_X|} \geq c$. Since *all* approximation spaces S are derived by *RS-Rules+*, there exist a S' , such that S' is an approximation space for Y , and hence the rule $X \rightarrow Y$ is induced, too. \square

⁶ The benchmark data can be found in *UCI Repository of Machine Learning Databases and Domain Theories* (URL: ftp.ics.uci.edu/pub/machine-learning-databases/Adult, /car, /mushroom).

⁷ All tests were executed on a PC with an AMD K6-2/400 processor.

⁸ A more detailed description can be found in Delic (2001).

5 Hybrid Association Rule Procedure *Apriori+*

Since the generated sets of rules of both procedures are almost equivalent, the selection of the procedure depends on the role of reducts. The best solution would be an additional extension of the faster association rule method with functions of the rough set procedure, such as formation of reducts and the ability to refer to a fixed decision attribute if needed. This hybrid method was realized with our algorithm “*Apriori+*”: The derived rules are examined on reducts, unnecessary rules are deleted. If there is a given fixed decision attribute, all itemsets without this attribute can be ignored for rule generation.

A further benchmark test confirmed the advantages of *Apriori+*: With all examined data sets the same rules (including reducts) as with *RS-Rules+* have been extracted, whereby the necessary computing times corresponded to almost 100% to the *Apriori* association rule procedure (see Table 3).

Table 3. Complexity (CPU time) grouped by *database*, *parameters* and *the use of the decision attributes* for the algorithms *Apriori* (Apr), *Apriori+* (Apr+) and *RS-Rules+* (RS+).

Database	Car Evaluation					Mushroom					Adult				
Minsupport	10%					35%					17%				
Minconfidence	75%					90%					94%				
Fixed Decision Attribute	Yes		No			Yes		No			Yes		No		
Method	RS+	Apr+	Apr	RS+	Apr+	RS+	Apr+	Apr	RS+	Apr+	RS+	Apr+	Apr	RS+	Apr+
CPU Time [min]	1,15	1,12	1,10	3,15	1,12	3,32	2,02	2	15	2,02	64	44	44	233	44

From the examination of reducts on the rules and Theorem 1 follows directly:

Corollary. Let $c, s \in (0, 1]$ values for minimum confidence and minimum support. Then both algorithms *Apriori+* and *RS-Rules+* induce identical rules from given data D w.r.t. c and s , if the extraction of reducts does take place.

In this case, the same redundant rules are left out by both algorithms.

6 Summary

The association rule procedure, which was originally developed for processing attributes with Boolean domains, can be based upon bitmap tables in order to analyze attributes with multi-value domains. The Rough set procedure

originally was for generating rules with a fixed decision attribute. Evidently, the need of an a-priori-definition was removed.

On the basis of different data sets we compared the quality of produced rules and the necessary computing times of the algorithms. It turned out, that the rules of *RS-Rules+* and *Apriori+* do not differ. The computing times were in favor of the association rule procedure.

Even so, any final judgement should be stated carefully, since many factors have a great impact. The needed time of computation e.g. depends also strongly on the implemented algorithms. Improvements with the rough set algorithm could lead to reduction of computing time. Another factor is the methodology of the rule generation. It cannot be excluded that there exists another rough set based procedure which is more efficient for rule generation than the one introduced here. The investigation on these and other related questions will be subject of further work.

References

1. Agrawal, R., Imielinski, T., Swami, A. (1993). Mining Association Rules between Sets of Items in Large Databases. In: *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, Washington D.C., USA. 207-216. ACM Press.
2. Agrawal, R. and Srikant, S. (1994). Fast Algorithms for Mining Association Rules in Large Databases. In: *VLDB'94*, 487-499. Morgan Kaufmann.
3. Delic, D. (2001). Data Mining - Abhängigkeitsanalyse von Attributen mit Assoziationsregeln und Rough Sets. MS thesis. Free University of Berlin, Institute of Applied Computer Science, Berlin, Germany.
4. Jürgens, M. and Lenz, H.-J. (2001). Tree Based Indexes Versus Bitmap Indexes: A Performance Study. *International Journal of Cooperative Information Systems*, **10**, 355-376.
5. Munakata, T. (1998). Rough Sets. In: *Fundamentals of the New Artificial Intelligence*, 140-182. New York: Springer-Verlag.
6. Pawlak, Z. (1982). Rough Sets. *Int. J. Computer and Information Sci*, **11**, 341-356.
7. Skowron, A. and Rauszer, C. (1992). The discernibility matrices and functions in information systems. In R. Słowiński (ed.): *Intelligent Decision Support. Handbook of Applications and Advances of Rough Sets Theory*, 331-362, Dordrecht: Kluwer.
8. Rauszer, C. (1991). Reducts in information systems. *Fundamenta Informaticae*, **15**, 1-12.