J. Appl. Environ. Biol. Sci., 4(8S)275-280, 2014

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

Record-Couple Based Production Rule Mining Algorithm: Tested in Medical Billing Domain

Umair Abdullah[1], Aftab Ahmed[2], Sohail Asghar[3], Kashif Zafar[4]

[1]Assistant Professor, Department of Software Engineering, Foundation University Islamabd, Pakistan
[2]Professor, Department of Software Engineering, Foundation University Islamabd, Pakistan
[3]Professor, Comsats Institute of Information Technology, Islamabad, Pakistan
[4]Associate Professor, National University of Computer and Emerging Sciences, Lahore, Pakistan

Received: September 1, 2014
Accepted: November 13, 2014

ABSTRACT

Most research in the area of rule mining tends to focus mining of either 'classification rules' or 'association rules', while ignoring 'production rules'. Production rules are building blocks of rule based expert systems. This paper presents an algorithm for production rule mining. Algorithm matches two states of an object; current state and previous state - collectively termed as 'record-couple'. The algorithm tries to find differences within a record-couple, and generates production rules from these differences. The proposed algorithm has been tested on synthetic data and then applied on actual data. The Algorithm is being used for supplying production rules to a rule based expert system which is applied in medical billing domain. The system is being used for finding knowledge oriented data inconsistencies in medical claims. Implementation of the algorithm is a demonstration of how data mining technology can be used for providing continuous supply of knowledge to rule-based expert systems.

KEYWORDS: Data Mining, Production rule mining; rule based expert systems; knowledge engineering; medical billing.

INTRODUCTION

Rule mining is one of the important areas of research in the field of data mining [1]. From computer science perspective term 'rule' is used for 'if ➔ then' structure (i.e. 'If <condition> then <result>'). Logically, we can further split rules in three categories; association rules, classification rules and production rules. We can write the three types of rules in the form of implications;

Association Rule: *item(s)* ➔ *item(s)*
Classification Rule: *conditions* ➔ *class*
Production Rule: *conditions* ➔ *action*

In association rules, condition is a check on presence of objects in itemsets, and the result part indicates presence of objects in the corresponding itemsets. In classification rule, the result part is name of a class, and condition part contains some checks. In a production rule, condtion part is a collection of checks while result part is some 'action'or command.

Concept of empty rule is valid for association rules i.e. it is logically possible to have an association rule with no (empty) result part. To some extent it is also logical in classification rules to have empty/null class in result part. In such cases, 'null' can be a default class containing all those objects which do not belong to any class. However, it seems to be illogical for a production rule to be defined with empty/null result part (i.e. there is nothing to do), if no piece of code is to be executed then why bother check conditions of the rule.

Classification rules and association rules are used for decision support such as suggesting similar books when a user is purchasing books from a web store. Production rules are used by rule based expert systems for automating tasks. For example, AthenaHealth [2] developed a rule based expert system for scrubbing medical claims (i.e. identifying errors and performing small, legitimate, corrective actions on the claims by the system itself)

Although we can link-up action with association rules or classification rules; within association rules, single action is defined for all associations (like 'displaying' related books together). Similarly, in classification rules, action can be defined at class level. However, in production rules, action is defined at rule level.

In fact, Artificial Intelligence (AI) community is familiar with concept of 'production rules' and rule based systems, however data mining community only focuses on association rules and its sub types such as 'negative association rules' etc.

This research tries to fill this gap by proposing a production rule mining algorithm. Extracted rules are then supplied to a rule based system; thus speeding up the knowledge acquisition process of the system and maximizing the potential of data mining technology -by making extracted rules available for automated use. The algorithm has been tested in medical claim processing domain. A rule based system was already in operation. Implementation of the algorithm has enhanced the performance of the system by providing extracted rules to it.

Corresponding Author: Dr. Umair Abdullah, Assistant Professor, Department of Software Enigneering, Foudnation University Islamabad, Pakistan. Tel.: +923005341053; Fax: +92515151434

Section 2 gives an overview of research carried out in rule mining domain. Section 3 gives definition of some the main term i.e. 'record couple' used in the algorithm and presents the proposed algorithm Section 4 describes results of applying the proposed algorithm on a test data set. Section 5 discusses utilization of the proposed algorithm in real life operational environment. In the last section, the conclusion has been presented.

2. RELATED WORK

Agrawal et al. [1] proposed 'association rule mining' in 1993 and till now hundreds of association rule mining algorithms have been proposed [3],[4]. Association rule mining is an important research topic in the field of data mining [5]. There are hundreds algorithms for mining of association rules, for example RULES series [6], frequent pattern (FP) growth [7], etc.

Classification rule mining is relatively older area of research with many famous algorithms, such as ID3 [8], ILA [9], CMAR [10]. Taniar et al., [11] implemented mining of classification rules with the help of structured query language of Oracle. Production rule mining algorithm proposed in this paper has also been practically implemented using structured query language. Many researchers have differentiated classification rules from association rules [10], [12], however, no one before this paper differentiated them from production rules.

Moreover, data mining researchers focus only on mining of classification rules and association rules (ignoring the production rules), and present the extracted rules to human decision makers. It is upon human decision makers either to use those rules or ignore them; which limit the potential of data mining technology.

Proposed algorithm is unique from previous work as it try to mine production rules which are then directly useable by a rule based system. The algorithm speeds up the knowledge acquisition process for rule based expert system; knowledge engineers just need to validate the extracted rules [13]. As of Tjioe & Taniar., [5], our proposed algorithm also works on a data warehouse instead of multi-dimensional transactional database. Similarly, as of Taniar et al., [11] algorithm has been practically implemented using structure query language.

3. Production rule mining algorithm

Proposed algorithm is for application domains in which same object has accepted state, which was previously in rejected state. Such as, a student passing a previously failed course, a medical claim accepted by insurance which got rejected initially due to some missing information. Algorithm tries to find difference between accepted and rejected states of same object, and check all other objects of same rejection type. Daily millions of records are processed by thousands of systems working worldwide. Processing systems can reject inconsistent records and accept corrected ones.

Proposed algorithm tries to find what corrective action is done by the user on a rejected record so that it got corrected on the basis of counts.

Definition: A 'record-couple' (RC) is collection of two states of an object i.e. current accepted one and previous rejected one stored in the dataset in the form of two tuples/records of a Table

$$\text{Record Couple} = \hat{C} = \{R, C\}$$

$$R = \langle V_{eid}, R_1, R_2, R_3, \ldots\ldots, R_m \rangle$$
$$C = \langle V_{eid}, C_1, C_2, C_3, \ldots\ldots, C_m \rangle$$

$$V_{eid} = V_{eid} \quad \text{and} \quad \exists\ i : R_i \neq C_i$$

Where 'R' is represents rejected record, and 'C' is the corrected record. 'Ri' is value of ith attribute in rejected record and 'Ci' is value of ith attribute in accepted record. 'Veid' is value of primary key, having same value (i.e. 'Veid') in 'R' and 'C' implies that both records belongs to the same object.

Difference between corrected and faulty records (within a record-couple) is actually action done by the user (in order to remove the fault). 100 % confidence for an action means same action is performed in all record- couples. Same attribute values (with same action) in all records of a dataset helps to determine 'when' that action should be performed (condition part of the rule).

Input a data set 'D' of n record-couples (each record-couple has a rejected and a corrected record), along with a threshold confidence ĉ, algorithm produces list L of production rules.

$$D = \{ \hat{C}_1, \hat{C}_2, \hat{C}_3, \ldots, \hat{C}n \}$$

Algorithm Steps:-

1. From all record-couples of the dataset D to identify 'focussed-attributes' (i.e. columns which has different values in rejected tuple and in corrected tuples).
2. Generate 'R-Table' and 'C-Table' from dataset D containing only rejected-values and accepted values respectively.
3. From 'R-Table' and 'C-Table' obtain additional information about 'Focussed Attributes' like their minimum and maximum lengths, list of wrong values (from R-table) and list of correct values (from C-Table), any fixed symbols compulsory (present in all correct values) etc.
4. For i = 1 to n, (where $\hat{C}_i$ is i^{th} record-couple of dataset D).

Match all attributes of R (rejected tuple of $\hat{C}_i$) and C (corrected tuple of $\hat{C}_i$) to update 'What-count' table with what has been done within a record couple (i.e. which column inserted/update/deleted).

5. From all records of dataset D (includind rejected and corrected versions) identify which columns have same values and prepare a 'when-count' table.
6. Create a merged 'What-When-count' table by combining 'what-count' and 'when-count' tables on the basis of primary key attributes.
7. Apply following subalgorithm to generate rule list L.

Rule List: L = {}
Action: nil
Condition: nil
Rule: nil

1. For each i^{th} row of 'what-when-table' do;
1.1 Suppose WWij is count value of i^{th} row ($what_i$) and j^{th} column ($When_j$)
1.2 $\forall$ WWij > $\hat{c}$: Condition = $When_j$ ^ Condition
1.3 Action = *action* ($What_i$)
1.4 Rule = Condition ➔ Action
1.5 L = Rule $\cup$ L
2. Return rule list L

'Action ($What_i$)' returns call of the functionaction which is stored as rule-actionin 'user-action rule-action mapping' table. Confidence of a production rule can be calculated by equation (1) as follow;

$$Confidence\ (\text{Rule}) = \frac{B}{|D|} \quad \text{-------- (1)}$$

Where 'B' is count of records in which both 'What' (Action) and 'When' (condition) exist, and | D | represents total number of records in a dataset. Threshold confidence is the minimum confidence value of a rule which allows generation of the rule. Its value is adjusted by the user at runtime.

4. Applying the proposed rule mining algorithm on medical billing dataset

Proposed algorithm is tested on a medical claim dataset. Table 1 – to be used by the Alogorithm – given below, shows mapping of rule-action and user-action in medical billing domin.

Table 1: Sample records of 'User-Action-Rule-Action' mapping table

SR	Reason/ What	Action of production rule
1	Datetime is less than other datetime column	Store second column valude as datetime of the first column
2	Datetime is greater than other datetime column	Store second column value as datetime of the first column
3	Predefined suffix missing	Append the suffix
4	Predefined prefix missing	Attach the prefix
5	Required character missing inside string value	Insert the required character at specified position
6	Negative value stored while postive value required	Change the sign to + ive
7	Value from invalid list inserted	Block the claim
8	Value not from predefine list of values	Block the claim
9	*Default action*	Block the claim

Dataset: A test dataset of 500 claims is given as input to the proposed algorithm with fault of 'patient marital status invalid or missing'. Three record couples of dataset have been shown in Table 2 given below; shaded records (with serial number 1, 3, 5) are rejected ones, while records with serial number 2, 4, 6 are corrected ones.

Table 2: A portion of dataset showing three record couples (having SR 1-2, 3-4, 5-6)

SR	Claim No	Payer Code	Patient Account	Marital status	……
1	C101	I011	P10003		……
2	C101	I011	P10003	Married	……
3	C102	I012	P10006	Unknown	……..
4	C102	I012	P10006	Single	……
5	C103	I011	P10024	Undefined	……
6	C103	I011	P10024	Un-married	……

In this example, correct values of focused column 'marital status' taken from corrected records of all record-couples of the rejection, are 'Married' and 'Single' listed in 'C-Table'. Rejected records have values '', 'Unknown', Undefined'

which are listed in 'R-Table' (i.e. Table 3 and Table 4 respectively). These tables are used to generate list of valid and invalid values of the attribute 'Marital status'.

Table 3: Few records of R-Table (containing rejected records)

SR	Claim_no	Marital status
1	C101	' '
3	C102	Unknown
5	C103	Undefined

Table 4: Few records of C-Table of rejection data set 3

SR	Claim_no	Marital status
2	C101	Married
4	C102	Single
6	C103	Un-married

Table 5 shows a portion of 'what-table' generated by step 4. 'When-Table' generated at step 5 shown as Table 6 given below;

Table 5: Two sample records What-count table

SR	Action	Couple Count
1	Marital status inserted	265
2	Marital status modified	142

Table 6: Few records of When-table

SR	Column	Value	Couple Count
1	Marital status	*blank*	**265**
2	Marital status	Unknown	93
3	Marital status	Undefined	42

'what-when table' generated during the step 6 of production rule mining algorithm, is shown as Table 7, given below. Counts greater than threshold value (50 in this example) have been encircled.

Table 7: Two tuples of What-when (action –condition) Count Table

What-When	Condition 1 *Marital status* = '' (blank)	Condition 2 Marital status = 'unknown'	Condition 3 Marital status = 'Undefined'
1 Marital status inserted	**265**	20	5
2 Marital status modified	10	**93**	42

From the test dataset our porposed algorithm has generated following production rules;

(<Patient Marital status > = '') ➔ block the claim
(<Patient Marital status > = 'unknown') ➔ block the claim
(<Patient Marital status > = 'undefined') ➔ block the claim

Action, 'block the claim' has been inserted from 'User action-Rule action' mapping table, by the rule generation step 4.

5. Utilization of the algorithm

Proposed production rule mining algorithm is being used to supply knowledge (in the form of production rules) to a rule based expert system implemented using structured query language, applied in the domain of medical billing [14].

As shown in Figure 1, during first four months period (of last year) a total of 208 rules have been created, while during second four month period a total of 366 rules created. Thus during first eight month period a total of 574 rules created with monthly average of 71.75 rules per month. Number of rules created during last four months (of the last year) are 895, with monthly average of 223 rules. Speed of rules creation increased by 3.11 times as compared to previous months of the year, after the implementation of proposed algorithm. The proposed production rule mining algorithm works at the background, on a data warehouse. Hybrid design of the data warehouse is presented by Aftab [15]. Therefore it does not have any negative effect on the performance of operational environment. Domain experts validated the extracted rules with the help of a GUI based knowledge editor [3]. It is important to note that 'identification' of the error is actual achievement of this algorithm. Its correction is still implemented by the human experts, may be in the form

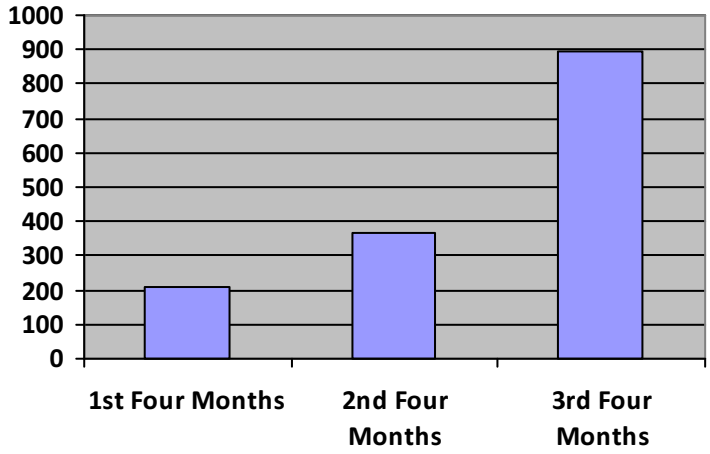

Fig. 1. Graph showing number of rules developed during three terms (four month each) of one year.

of a stored procedure or in the form of a check on front end (user interface).

6. CONCLUSION

This paper proposed to use data mining technology for knowledge acquisition of rule based expert systems. Presented algorithm is unique and different from other rule mining algorithms as it focuses mining of the rules immediatley useable by a rule based system. Main entity used in the proposed algorithm is record-couple. A 'recordcouple' is collection of two states of same object, last sate (i.e. faulty one) rejected by some processing system and current state (i.e. corrected one), which is accepted by some processing system. Algorithm counts the differences within record-couples of same rejection type to identify what domain users have done to correct the rejected states. User actions have been mapped with actions to be performed by the system, combined with conditions to generate production rules. Automated identification of errors is the main achievement of the proposed algorithm, how those errors are automatically rectified (at runtime) can be future future extension of the algorithm. The proposed algorithm can be used in any application domain which involves concept of 'record-couples' i.e. rejected and corrected states of entities. It may be extened in order to address domains with fuzzy rules [16] and for the recent domain of cloud computing [17].

Acknowledgments. Many thanks to Healthcare IT company MaxRemind (http://www.mremind.com/) for providing excellent research environment. This research is financially supported by Higher Education Commisson of Pakistan (HEC) through '5000-PhD indeginous fellowship scheme'.

REFERENCES

1. Agrawal, R., Imielinski, T., & Swami. A. N., Mining association rules between sets of items in large databases. In: Peter Buneman, Sushil Jajodia editors. Proceedings of the 1993 ACM SIGMOD Intl. Conference on Management of Data. ACM Press, 1993, (pp. 207–216). Washington, D.C. USA.
2. athenahealth, I. (2010). Patent No. US7,720,701B2. USA.
3. Zaki, M. J., Scalable Algorithms for Association Mining. IEEE Transactions on Knowledge and Data Engineering, 12(3), 2000, 372-390.
4. Tjioe, H. C., & Taniar, D., Mining Association Rules in Data Warehouses. International Journal of Data Warehousing and Mining (IJDWM), 1(3), 2005, 28-62.

5. Zhang, S., & Wu, X., Fundamentals of association rules in data mining and knowledge discovery. Wiley Interdisc. Rew., Data Mining and Knowledge Discovery, 1(2), 2011 97-116.

6. Pham, D. T., & Afify, A. A., RULES-6: A Simple Rule Induction Algorithm for Supporting Decision Making. In proceedings of 31st Annual Conference of IEEE Industrial Electronics Society, IECON 2005. Raleigh, IEEE Press: 6-10 Nov. 2005 (pp. 2184-2189). North Carolina, USA

7. Han, J,. Pei, J., Yin, Y., & Mao, R,. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery, 1(8), 2004, 53-87.

8. Quinlan. J. R., Induction of Decision Trees. Machine Learning, 1(1), 1986, 81-106.

9. Tolun, M. R., & Abu-Soud, S. M., ILA: An Inductive Learning Algorithm for Production Rule Discovery. Expert Systems with Applications, 1(14), 1998, pp. 361-370.

10. Li, W., Han, J., & Pei, J., CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. IEEE International Conference on Data Mining (ICDM), 2001, (pp. 369-376), San Jose, California, USA.

11. Taniar, D., D'Cruz, G., & Rahayu, J. W., Implementation of Classification Rules Using Oracle PL/SQL. *FSKD,* 2002, 509-513

12. Freitas, A., Understanding the crucial differences between classification and discovery of association rules - a position paper. ACM SIGKDD Explorations, 2(1), 2000, 65-69,

13. Abdullah, U., Ahmed, A., & Sawar, M. J., Knowledge Representation and Knowledge Editor of a Medical Claim Processing System. Journal of Basic and Applied Scientific Research, 2 (2), 2012, 1373-1384.

14. Abdullah, U., Sawar, M. J., & Ahmed, A., Design of a rule based system using Structured Query Language. In: Yang, B editor. Proceedings of 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC09), IEEE Press; 2009, (pp. 223-228). Chengdu, China.

15. Ahmed, A., Zafar, K., Siddiqui, A. B. , Abdullah, U., "Data Warehouse Design For Knowledge Discovery From Healthcare Data," The 2013 International Conference of Data Mining and Knowledge Engineering (ICDMKE), In: S. I. Ao and Len Gelman and David WL Hukins and Andrew Hunter and A. M. Korsunsky Editors. *Proceedings of The World Congress on Engineering 2013*, London, U.K., July 3 – 5, 2013. pp1589-1594, ISBN: 978-988-19252-9-9

16. NAZIR, S., & NAZIR, M. "Comparisons Of Membership Functions For Fuzzy Rules." VAWKUM Transaction on Computer Sciences, 3(1), 2014, 10-14.

17. Iqbal, W., & Yousaf, S. Formal Modeling of Agent Based Cloud Computing Services using Petri nets. VFAST Transactions on Software Engineering, 1(2), 2013, 1-6.