http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 8 (7): 1039-1043, 2009
ISSN 1812-5638

Inducing Positive and Negative Rules Based on Rough Set

Tinghuai Ma, Jiazhao Leng, Mengmeng Cui and Wei Tian
School of Computer and Software, Nanjing University of Information Science and Technology, China

Abstract: Traditional classification rules take the positive form as C→D. A new method of retrieving the negative ¬C→¬D form is introduced in this paper. Negative rules can improve the classification quality in some case. We propose a classification algorithm named Rule Generation based on Classification Attribute (RGCA) to deduct negative and positive rules. The RGCA algorithm won't need processing records item by item. The real dataset are used to verify the presented algorithm. The result shows the negative rules is more than positive rules based on RGCA algorithm, the classification accuracy of RGCA algorithm is better than traditional positive based algorithm.

Key words: Rough set, algorithm, classification, positive and negative rules

INTRODUCTION

Classification is one of the most important techniques for data mining and machine learning. There are two factors, accuracy and efficiency, needing to be carefully considered. For example, it's very difficult to improve calculation efficiency due to high data dimension. The core of classification algorithm is to select the right attribute, which is used to find the least subset of condition attribute without violating the accuracy requirement. The rule with least condition attribute pairs, it only needs to match less condition attribute and thus improve the match efficiency, while a new case is classified.

Increasing classification's precision is the target of algorithm. Effective and efficient are very important for data mining. Classification was formerly improved by designing different algorithms to obtain rules in the form of C→D. But this form as rules couldn't improve classification accuracy sharply because it is not easy to improve the matured algorithms. New rules in the form of ¬C→¬D are introduced into data mining. Wu *et al.* (2004) discussed how to use negative association rule and designed constraints to reduce the search space. Ji and Tan (2004) studied how to inducing negative and positive rules for gene expression, which is also based on association rules. Tsumoto (2001) used positive and negative rules to predict clinical case, which is based on rough set.

As a new technique of data mining, the rough set has been thought as a suitable method for attribute reduction (Pawlak and Skowron, 2007). It is a new mathematical method to deal with vague and uncertain knowledge. It provides a satisfied way to deal with classification based on mathematics (Pawlak and Skowron, 2007; Hu *et al.*, 2006). Nowadays, there are many rough set applications, such as process control, KDD (data mining), pattern recognition, classification, medical reasoning and fault diagnosis (Hu *et al.*, 2006). With rough set, rules can be described as either positive or negative Tsumoto (2001). It can remedy the low accuracy caused by the singleness of positive rules reasoning.

To obtain rules based on rough set, records are processed item-by-item. For the rule reduction that based on classification attribute, it does not need to deal with records one by one so that to improve calculation efficiency. Negative rules can be generated along with positive rules. It can improve the classification accuracy. Sometimes, reasoning can be done based on negative rules and it reduces the classification time.

BASIC CONCEPTS

The classic knowledge system can be represented as: S = <U, C, D, V, f>. The complete domain U is a finite set of objects and the elements in U are objects/instances. C is a condition attribute set D is a decision attribute set. A is set of attribute and equals to C∪D. V is the set of attribute values and V_a is a domain of attribute a. f: U×A→V is a function such that f (x_i, a) ∈V_a for every a∈A, x_i∈U.

In Table 1, C = {a, b, c}, D = {d,e), V_b = {0, 1, 2}, V_c = {0, 2}, $V_{d,e}$ {0, 1, 2}.

Corresponding Author: Tinghuai Ma, School of Computer and Software,
Nanjing University of Information Science and Technology, Ningliu Road 219,
Pukou, Nanjing, Jiangsu, 210044, China Tel: 86-25-5873-1400

Table 1: Decision Table

U	a	b	c	d	e
x_1	1	0	2	1	1
x_2	2	1	0	1	0
x_3	2	1	2	0	2
x_4	1	2	2	1	1
x_5	1	2	0	0	2

The binary relation of indiscernibility is $\text{ind}(C) = \bigcap_{a \in C} \text{ind}(a)$. The attribute value of a for object x is a(x). ind (a (x)) is an equivalence class that including x and marked as $[V_a = a(x)]_a$ For example, if a (x) = 1, the equivalence class is $[V_a = 1]_a = \{x_1, x_4, x_5\}$.

The first record in Table 1 can be written as:

$$[a = 1] \wedge [b = 0] \wedge [c = 2] \rightarrow [d = 1, e = 1] \quad (1)$$

In order to describe the rule's accuracy degree, the trust degree is adopted to estimate the quality of classification. In this study, we use classification accuracy ratio and coverage ratio to describe rule.

Definition 1: Let R denote the equivalent set of condition, D denote the equivalent set of conclusion. For rule r→d, two variables are defined as follow:

Accuracy ratio: $a_R(D) = |R \cap D|/|R|$, which means the probability of D occurrence can be denoted as P (D|R) for a rule that satisfies R.

Coverage ratio: $K_R(D) = |R \cap D|/|D|$, which means the probability of R occurrence can be denoted as P (R|D) for a rule that satisfiesD.

As shown in Table 1, let R equal [a = 1], D equal [d = 0, e = 2], then $[V_a = 1]_a = \{x_1, x_4, x_5\}$, $[d = 0, e = 2]_{d,e} = \{x_3, x_5\}$, $a_R(D) = 1/3$, $K_R(D) = 1/2$.

Thus, the rule r→d can be rewritten in the following form:

$$\wedge_j [a_j = v_k] \rightarrow d(\alpha_R(D), \kappa_R(D)) \quad (2)$$

$\alpha_R(D)$, $\kappa_R(D)$ denote the rule's accuracy ratio and coverage ratio respectively. The former rule can be expressed as:

$$[a = 1] \rightarrow [d = 0, e = 2](0.33, 05) \quad (3)$$

Definition 2 (positive rule) R→d is positive rule if $\alpha_R(D) = 1.0$ and $R = \wedge_j [a_j = v_k]$.

While $\alpha_R(D) = 1.0$, the equivalence class of D is a subset of equivalence class of D, it means $[R] \subseteq [D]$. As we know, the positive domain of D, $\text{Pos}_R(D)$ equals to $\cup \{Y = U/R, Y \subseteq D\}$. $|Y \cap D|/|Y|$, if $\alpha_R(D) = 1.0$. Thus the positive rules are those coming from $\text{Pos}_R(D)$.

In the above approximation of D relative to R, $R^-(D) = \cup \{Y \in U/R; Y \cap D \neq \Phi\}$, so $|R^-(D) \cap D||D| = 1.0$ equals 1.0. This can be explained as $R^-(D)$ is a particular case that satisfies $K_R(D)$ 1.0.

In Table 1, let R = a, D is [d = 0, e = 2]. U/IND (R) = $\{\{x_1, x_4, x_5\}, \{x_2, x_3\}\}$, $[d = 2, e = 2]_{d,e} = \{x_2, x_5\}$, Thus $R^-(D) = \{x_1, x_2, x_3, x_4, x_5\}$ and $|R^-(D) \cap D/|D| = 1.0$. To obtain [d = 0, e = 2], there are two choices [a = 1], [a = 2] for R = a. Thus,

$$[d = 2, e = 2][a = 1] \vee [a = 2] \quad (4)$$

Formalization: $d \rightarrow \vee_j [a_j = v_k]$

So, we get $\wedge_j \neg [a_j = a_k] \rightarrow \neg d$

Definition 3 (negative rule (Tsumoto, 2001)) $\wedge_j \neg [a_j = a_k] \rightarrow \neg d$ is negative rule, while $\vee_j [a_j = v_k]$ (D) $\kappa_{\vee_j [a_j = v_k]}(D) = 1.0$.

Consider Eq. 4, for the same attribute a, $[a = 1] \vee [a = 2] \rightarrow [d = 2, e = 2]$, $\kappa = 1.0$ and $[a = 2] \vee [b = 2] \rightarrow [d = 2, e = 2]$, $\kappa = 1.0$. Also, we can get negative rule: $\neg [a = 2] \wedge \neg [b = 2] \rightarrow \neg [d = 2, e = 2]$, where a, b are different attributes.

According to above analysis, we come to the conclusion that negative rule can be deducted from combination of different values for the same attribute, also can be deduced from combination of different attribute-value pair. And it will cost much time to generate all negative.

ALGORITHM OF RULE GENERATION

Positive and negative rules are the two forms for an equivalent class of D. The negative rule corresponds to the negative domain $\text{Neg}_R(D)$ while the positive rule corresponds to the positive domain $\text{Pos}_R(D)$. In rough set theory, each positive domain is with a negative domain. For a particular R and D, there are several rules in a positive domain but only one negative rule in a negative domain. The classic rule reduction is based on records in the decision table, where the records are serially processed. It is not suitable for negative rule generation. The rule generation based on classification attribute that we provide can generate positive and negative rules in parallel instead of serial process.

Positive and negative rules own the same feature as $[a_j = v_k]_a \cap [d]_D \neq \Phi$. Thus it's necessary to find out all the condition attribute-value pairs that satisfy $[a_j = v_k]_a \cap [d]_D \neq \Phi$ while D's value equals d.

For positive rules, we need to find out all attribute pairs' conjunctions. Those attribute pairs should satisfy $\wedge_j [a_j = v_k] \subseteq [d]_D$, where j is the maximum number of condition attribute. For negative rules, $\vee_j [a_j = v_k] \supseteq [d]_D$ needs to be satisfied.

```
Void PosNegAlgorithm(){
    L = all condition attribute-pairs( [ai=vi] )
      For( j=1; j<k; j++){ //k is the equivalent class number of //classification attribute
          Select a equivalence class of classification [d]aj;
          While ( L {}) do
          {
                Select one pair [aj=vk] from L
                If [ai=vi] A [d]aj then
                      Laj = Laj+[ai=vi];
          }// Laj is all attribute- pairs, which overlap with [d aj
    Laj=Laj; // The positive rule generation is begin
          for (i=1; i< n; i++) { //n is the number of condition attribute
              While (Laj !=( )) do
                  {
                   Select one pair  [aj=vk] from Laj;
                   Laj = Laj -{R};
                   If (aR(D)=1.0) then Pos=Pos+{R};
                   Else M=M+{R};
                   }
        Laj+1=all attribute- pairs which generated from M, conjunct every two attribute-pair in M
    }// The positive rule generation is over
   LLi=Li  //the negative rule generation is begin
   For( i=1; i< m; i++){  //m = k1 k2 ... kn          //ki is the equivalence class number of attribute I
     While (LLi ! ={}) do
      {
          Select one pair  [ai=vi] from LLi;
          LLi = LLi -{R};
          If ( ) then Neg=Neg+{R};
          Else N=N+{R};
       }
      LLi+1= all attribute- pairs which generated from N, connect every two attribute-pair in N
 }// the negative rule generation is over
 Generate positive and negative rules
 } // loop j is over
 }// Algorithm is over.
```

Fig. 1: Algorithm RGCA (Rule Generation based on Classification Attribute)

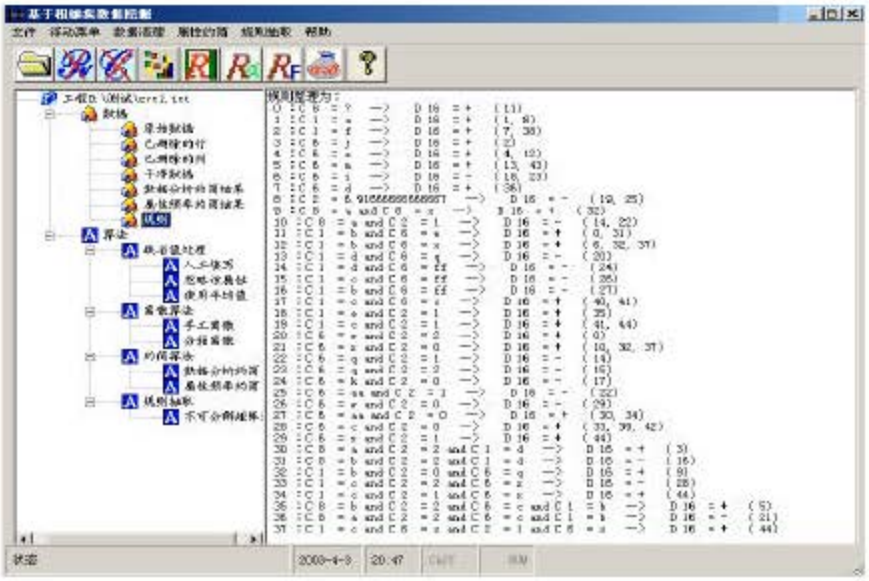

Fig. 2: Algorithms run window (RGCA)

The algorithm is described as follow:

- **Step 1:** Select an equivalent class $[d]_D$ of classification attributes and find out all condition attribute-pair L_{ir}, which overlaps with set $[d]_D$
- **Step 2:** Check condition attribute-pair R in $L_1 = L_{ir}$ and add those satisfy $\alpha_R(D) = 1.0$ to Pos set list and delete them from L_1. Otherwise, these condition attribute-pairs will be kept in L_1. The left pairs in L_1 will be conjunct in the following phase
- **Step 3:** Connect every two pairs in L_{ir} , use LL_1 represent connection result. If $K_R(D) = 1.0$, add R to Neg set and delete it from LL_1. Otherwise, these condition attribute-pair will go through next phase, where these pairs will be connected
- **Step 4:** With Pos, Neg, d, the positive and negative rules are obtained

It costs much more time to get negative rules than positive rules. Thus, we focus on achieving positive rules and negative rules act as supplements. At the same time we must reduce negative rules deduction time. Since, the negative rules deduction time increase when the negative condition dimension of is high, we ignore the negative rule with high condition dimensions. The algorithm pseudo code is as follow Fig. 1 and the GUI of algorithm is as Fig. 2.

EXPERIMENT RESULTS

We use datasets from UCI repository of machine learning databases and domain theories to verify the presented algorithm. Considering that rough set theory algorithms' classification precision is affected by attribute's discreteness forms, we select those datasets without need discrete.

It's not necessary to obtain all the negative rules, since it takes time to deduce them and some of them may have only a little effect for classification or reasoning. A threshold is used to limit the time for negative rules.

As we know, negative rules as $\wedge_j \neg[a_j = v_k] \rightarrow \neg d$, where a_j may come from the same attributes or different attributes. Assume μ is the number of a_j which comes from different attributes, μ should satisfy the following equation, as our suggestion.

$$\mu \leq n/3 \quad (5)$$

where, n is the number of dataset's attribute. μ is a parameter decided by users and used to select relative simple formula for negative rules' computation.

Table 2 summarizes our analysis results, which shows the number of positive rules and negative rules obtained by RGCA. The number of negative rules is bigger than positive rules under the threshold μ. More redundant negative rules will be obtained by increasing μ.

Table 2: Result of the RGCA

Dataset	Samples	Attributes	Positive rules	Negative rules
Lens	12	6	8	14
Balloons	20	4	13	44
Iris	150	4	27	43
Wine	178	4	268	532

Table 3: The classification of positive rules and negative rules

Dataset	Positive rules (%)	Positive rules and negative rules (%)
Lens	63	82
Balloons	56	75
Iris	78	84
Wine	68	73

In order to test the negative rules' usage for classification and reasoning, each dataset was divided into two subsets equal in data number. One is for generating positive and negative rules (supervised learning) and the other is for reasoning. The samples were randomly selected for learning.

Based on RGCA, we can easily get the positive rules and perform classification experiment based only on positive rules. The classification precision is shown as follow.

Table 3 shows that the classification precision increases rapidly when samples used for learning are insufficiency. It means that negative rules have positive effect on classification and increase the reasoning precision. When samples are adequate, the classification mainly depends on positive rules while negative rules affect classification slightly.

It costs much more time to obtain negative rules than positive rules. The time will increase as the samples and attributes become larger. But for datasets with large samples and attributes, the positive rules are sufficient enough for classification and there is no need to deduce negative rules. The suggestion for such scenario is letting μ equal 1.

DISCUSSION

For the negative rules extraction, we don't deduce all negative rules for classification. Tsumoto (2001) gave an algorithm to deduce all negative rules based on rough set, but he didn't discuss the algorithm's efficiency. Alatas and Akin (2006) did not discuss the efficiency either while they provided a method to deduce all negative rules based on association rules. The presented algorithm is restricted to extract all negative rules in order to not decrease the algorithm's efficiency.

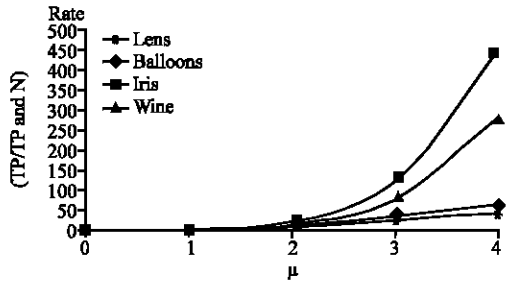

Fig. 3: The rate of algorithm with different μ

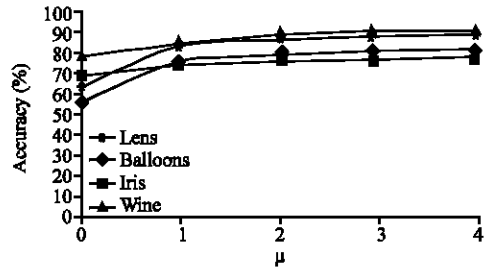

Fig. 4: The accuracy of algorithm with different μ

In this study, we discuss the negative rule form as: $\wedge_j\neg[a_j = v_k]\rightarrow\neg d$. The other form like $\wedge_j\neg[a_j = v_k]\rightarrow d$ is not discussed. But Alatas and Akin (2006) got negative rules like $\neg D\in[d_1, d_2]\Rightarrow C\in[c_1, c_2]$ based on association rule. We will present the RGCA in a later version later, which include the negative rule extraction like form $\wedge_j\neg[a_j = v_k]\rightarrow d$.

The negative rules extraction is much more complexity than positive rules. So, in order to balance the classification accuracy and efficiency, the negative rule's attribution number should be restricted. For the negative rules as $\wedge_j\neg[a_j = v_k]\rightarrow\neg d$, the a_j may come from the same attributes or different attributes. Intuitionally, the number of a_j has relation with the number of attribute number and the record number. How to decide the number of a_j is diverse with different dataset. We estimate the time of RGCA according to several values of μ (as shown in Eq. 5). We describe the efficiency of algorithm with rate = $T_P/T_{P\&N}$, where the T_P means the time of inducing positive rules, $T_{P \text{ and } N}$ means the time of inducing positive and negative rules. In Fig. 3, it showed that rate of time cost is increased hundredfold while μ>2. While μ>2, the classification accuracy is not increased significantly, as shown in Fig. 4. This is the reason why we assigned μ = 1 in the presented algorithm. While we put forward the formulation (5), the μ should be decided according to its dataset's attribute and sample numbers. This is another work we will do in the later.

CONCLUSIONS

Traditional algorithms treat consistent and inconsistent rules separately. The rule deduction based on classification attributes can deal with consistent and inconsistent rules in parallel. Assume there are inconsistent rules $R\rightarrow d_1$ and $R\rightarrow d_2$, traditional algorithms will not generate one rule that includes R. Thus the result of R is unknown. In RGCA, there exists a negative rule of d_1 and d_2, which condition part includes ¬R.

According to a decision attribute d, we assume there are conditions attributes $C = \{a_1, a_2, \ldots a_n\}$ Then we will get positive rule as $\wedge_j[a_j = v_k]\rightarrow d$, i = 1, ...n through the rule deduction based on classification attributes. The inversion of these positive rules' combination will lead to a negative rule $\neg[\vee[\wedge_j[a_i = v_k]]]\rightarrow\neg d$, i = 1,...n. And the equivalent class of $\cup[\wedge_i[a_i = v_k]]$ is mapped to the d's positive region.

For small samples and low dimension, the cost of deducing negative rules is not high but the effect of negative rules' classification is noticeable.

ACKNOWLEDGMENTS

This study is partly supported by the oversea study scholarship of jiangsu government and jiangsu youth project, Natural Science Foundation of Nanjing University of Information and Science Technology (20080302).

REFERENCES

Alatas, B. and E. Akin, 2006. An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules. Soft Comput.-A Fusion Foundat. Methodol. Appl., 10: 230-237.

Hu, Q., X. Li and D. Yu, 2006. Analysis on Classification Performance of Rough Set Based Reducts. In: PRICAI 2006: Trends in Artificial Intelligence, Yang, Q. and G. Webb (Eds.). Springer-Verlag, Berlin Heidelberg, pp: 423-433.

Ji, L. and K.L. Tan, 2004. Mining gene expression data for positive and negative co-regulated gene clusters. Bioinformatics, 20: 2711-2718.

Pawlak, Z. and A. Skowron, 2007. Rudiments of rough sets. Inform. Sci., 177: 3-27.

Tsumoto, S., 2001. Mining Positive and Negative Knowledge in Clinical Database Based on Rough Set Model. In: Principles of Data Mining and Knowledge Discovery, De Raedt, L. and A. Siebes (Eds.). Springer-Verlag, Berlin Heidelberg, pp: 460-471.

Wu, X., C. Zhang and S. Zhang, 2004. Efficient mining of both positive and negative association rules. ACM Trans. Inform. Syst., 22: 381-405.