International Journal of Soft Computing 2 (1): 124-128, 2007
© Medwell Journals, 2007

GA Optimized Negative Association Rule Mining

[1]Sanjeev Sharma, [2]Sudhir Sharma and [1]Jitendra Agrawal
[1]School of Information Technology,
[2]University Institute of Technology, Rajiv Gandhi Technological University,
Bhopal (M.P), India

Abstract: This study discusses the use of genetic algorithm in mining negative association rules. In general the rule generated by Association Rule Mining technique do not consider the negative occurrences of attributes in them, but by using Genetic Algorithms (GAs) over these rules the system can predict the rules which contains negative attributes. GA is used for this kind of knowledge discovery because of their ability to search globally and perform well in case of attributes interaction. Their strength is essentially due to updating of the whole population, consisting of normal case of individuals and worst case of individuals of the possible solution in an adaptive way.

Key words: Data mining, association rule mining, negative association rule, genetic algorithms

INTRODUCTION

Today, amount of data stored in the database is growing very rapidly. This data contains knowledge, which can be used to enhance decision-making process of an organization. Data mining refers to extracting or mining knowledge from large amounts of data. Knowledge Discovery in Databases or KDD is another synonym for data mining. This knowledge discovery can be done in various ways, like decision trees, association rule mining, bayesian classifier and so on. In this authors have considered association rule mining and try to optimize this technique by applying GA.

Association rules: With massive amount of data continuously being collected and stored, many industries are becoming interested in mining association rules from their databases. Association rules have numerous applications in real world such as decision support, understanding customer behavior, telecommunication alarm diagnosis and prediction etc. Department stores also can use association rules in many fields such as catalog design, add on sales layout etc.
In brief, an association rule is an expression:

$$X \rightarrow Y$$

Where X and Y are item sets and $X \Lambda Y = \Phi$.

Meaning of the above rule is: given a database D containing say N transactions where T belongs to D is a transaction, then $X \rightarrow Y$ expresses that wherever a transaction T contains X than T probably also contains Y. The support of the rule is the probability that X and Y hold together among all the possible presented cases. The probability or confidence is defined as the percentage of transaction containing Y in addition to X with regard to overall number of transactions containing X. This probability can be represented as conditional probability $P(Y \xi T / X \xi T)$. Above rule is introduced on the basis of the similarity with market basket data where rules like A customer buys milk and bread will also buy butter with the probability of A%.

All the traditional association rule-mining algorithms were developed to find positive associations between items. Positive associations refers to associations between items existing in transactions (i.e., items bought). What about associations of the type: customers that buy Coke do not buy Pepsi or customers that buy juice do not buy bottled water? In addition to the positive associations, the negative association can provide valuable information, in devising marketing strategies. Interestingly, very few have focused on negative association rules due to the difficulty in discovering these rules.

In order to generate these kinds of rules and to evolve quality rules, this study is using Genetic Algorithms.

NEGATIVE ASSOCIATION RULES

Example: Figure 1 shows synthetic data for vehicle purchase information.

Corresponding Author: Sanjeev Sharma, School of Information Technology Tel: +91-755-2678821

S.No	Name	Age	Vehicle
101	Shikha	23	Car
102	Sumit	45	Truck
103	Anil	50	Van
104	Altaf	27	Bus
105	Mukesh	38	Bus
106	Ritesh	20	Car
107	Aashish	41	Truck
108	Puneet	22	Car
109	Himanshu	39	Car
110	Shailesh	40	Bus

Fig. 1: Vehicle purchase information

Item Set	Support
Age>30	40%
Age<30	60%
Car	40%
Bus	30%
Age<30,Car	30%

Fig. 2: Large item sets with minimum support of 30%

Assumption 1: The minimum support is 30% and minimum confidence is 70%. Assumption 2: The numeric attribute AGE ranges from 18 to 70 and is quantized into two groups - less than thirty and over thirty. Item sets that satisfy *minsup* (= 30%) are listed in Fig. 2.

The rule that satisfies both minimum support and minimum confidence criterion is age<30 □car, the confidence of which is 75%. However, if we are also looking for negative association rules, there exists a rule: age>30 □not purchasing car, which has a confidence of 83.3%. For the purpose of identifying purchase pattern, it is obvious that the latter has better predictive ability.

The preceding example illustrates that negative association rules are as important as positive ones. However, there are at least two aspects making mining negative rules difficult. First, we cannot simply pick threshold values for minimum support and minimum confidence that are guaranteed to be effective in shifting both positive and negative rules. Second, in a practical database, thousands of items are included in the transaction records. However, there may be a large number of items in the domain of an attribute that do not appear in the database or appear an insignificant number of times. An absent item is defined to be one that occurs an insignificant number of times in the transaction set. If we consider negative rules for which the LHS or RHS is a combination of absent as well as frequently used items, then there will be a great many of them. A valid rule, however, is not necessarily a useful one. This article addresses both issues.

Formal definition of negative association rule: A negative association rule is an implication of the form $X \rightarrow \neg Y$ (or $\neg X \rightarrow Y$) where X and Y are item sets and $X \Lambda Y = \Phi$.

(Note that although rule in the form of $\neg X \rightarrow \neg Y$ contains negative elements, it is equivalent to a positive association rule in the form of $Y \rightarrow X$. Therefore it is not considered as a negative association rule.) In contrast to positive rules, a negative rule encapsulates relationship between the occurrences of one set of items with the absence of the other set of items. The rule has support s% in the data sets, if s% of transactions in T contain item set X while do not contain item set Y. Let U be the set of transactions that contain all items in X. The rule with confidence c%, if c% of transactions in U do not contain item set Y.

Genetic algorithms: Genetic Algorithms (GAs), a biologically inspired technology, are randomized search and optimization techniques guided by the principles of evolution and natural genetics. They are efficient, adaptive and robust search processes producing nearly optimal solution and have a large amount of implicit parallelism. GAs is executed iteratively on a set of coded solutions (genes), called population, with three basic operators: Selection/Reproduction, Crossover and Mutation. They use only fitness function information and probabilistic transition rules for moving to the next iteration.

The main motivation for using GAs in the discovery of high-level prediction rules is that they perform a global search and cope better with attribute interaction than the greedy rule induction algorithms often used in data mining. Their strength is essentially due to updating of the whole population, consisting of normal case of individuals and worst case of individuals of the possible solution in an adaptive way. There are four basic components of GA:

- Representation of individuals.
- Design of genetic operators.
- Determination of fitness function.
- Determination of the probabilities controlling genetic operators.

METRIALS AND METHODS

In this study the genetic algorithms are applied over the rules fetched from association rule mining. At first using a-priori technique implement Association Rule mining. Then GAs are applied to evolve the rules which contains negations in attributes and are of richer quality. In this section the paper discusses each step in detail.

Individual representation: Representation of rules plays a major role in GAs, broadly there are two approaches based on how rules are encoded in the population of individuals (Chromosomes) : Michigan and Pittsburgh. In the *Michigan approach*, all individuals in one population represent one solution. Each individual consists of a *condition* (a conjunction of several blocks) and of a *conclusion* -- one of possible solutions that this method can learn. An example of an individual can be <110|0001|1001> ➔ 3 representing, an expression ``it can walk, it can jump but it cannot fly AND it barks AND it is 90 cm long it is a dog'.

The *Pittsburg approach* closely corresponds with described ways of functioning of genetic algorithms: the solutions are represented by individuals that fight each other; those weaker ones die, those stronger ones survive and they can reproduce on the basis of selection, crossover and mutation. For example, an expression " the head is round and the jacket is red, or the head is square and it is holding a balloon", can be encoded as (<S = R> and <J = R>) (<S = S> and <H = B>).

The pros and cons are as follows: Pittsburgh approach leads to syntactically longer individuals, which tends to make fitness computation more computationally expensive.

In addition, it may require some modifications to standard genetic operators to cope with relatively complex individuals. By contrast, in the Michigan approach the individuals are simpler and syntactically shorter. This tends to reduce the time taken to compute the fitness function and to simplify the design of genetic operators. However, this advantage comes with a cost. First of all, since the fitness function evaluates the quality of each rule separately, now it is not easy to compute the quality of the rule set as a whole - i.e. taking rule interactions into account. In this study Michigan's approach is opted i.e. each individual encodes a single rule. For example let's consider a rule If a customer buys milk and bread then he will also buy butter, which can be simply written as:

If milk and bread then butter

Now, following Michigan's approach and binary encoding, for simplicity sake, this rule can be represented as 00 01 01 01 10 01 where, the bold di-digits are used as product id, like 00 for milk, 01 for bread and 10 for butter and the normal di-digits are 00 or 01 which shows absence or presence, respectively. Now this rule is ready for further computations.

Genetic operators: Second, area of concern is Genetic Operators. Mainly three operations are to be performed selection, crossover and mutation to robustly search the rule space for various options. Selection involves selecting two fit parents for evolving new children rules, which are fit than the parents and in this manner the average fitness of the rules can be increased. Crossover and mutation provides the ways to evolve new rules. For *selection* the authors suggests to use Roulette Wheel Sampling procedure. In this procedure, the parents for crossover and mutations are selected based on their fitness, i.e. if a candidate has more fitness function value more will be its chance to get selected. The implementation of Roulette Wheel Sampling can be done by first normalizing the values of all candidates so that, there probabilities lie between 0 and 1 and then by using random number generator, a random number is generated and then corresponding to this value and the fitness normalized value, the candidate is selected. Then the generalizing/specializing crossover operators can be implemented as the logical OR and the logical AND, respectively. This is illustrated in Fig. 3, where the above mentioned bitwise logical functions are used to compute the values of the bits between the two crossover points denoted by the | symbol. Children produced by children produced by parents generalizing crossover specializing crossover.

Fitness function: Third area of concern is fitness function. Since, the discovered rules should:

- Have a high predictive accuracy;
- Be comprehensible;
- Be interesting, thus choice of this function is very important to get the desired results.

The authors have proposed the same method described in[1]. Let a rule be of the form:

IF A THEN C,

Where A is the antecedent (a conjunction of conditions) and C is the consequent predicted class). A 2×2 matrix, sometimes called a confusion matrix, as illustrated in the following Fig. 4 summarize the predictive performance of a rule.

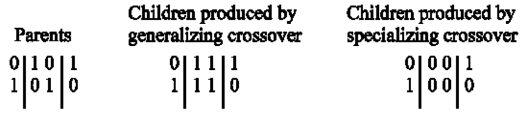
Parents
Children produced by generalizing crossover
Children produced by specializing crossover
0|1 0|1
1|0 1|0
0|1 1|1
1|1 1|0
0|0 0|1
1|0 0|0

Fig. 3: Example of generalizing/specializing crossover

		Actual class	
		C	not C
Predicted class	C	TP	FP
	not C	FN	TN

Fig. 4: Confusion matrix for a classification rule

The labels in each quadrant of the matrix have the following meaning:

TP	=	True Positives = Number of examples satisfying A and C
FP	=	False Positives = Number of examples satisfying A but not C
FN	=	False Negatives = Number of examples not satisfying A but satisfying C
TN	=	True Negatives = Number of examples not satisfying A nor C

Clearly, the higher the values of TP and TN and the lower the values of FP and FN, the better the rule. Confidence Factor,

$$CF = TP / (TP + FP) \quad (1)$$

Now measure the predictive accuracy of a rule by taking into account not only its CF but also a measure of how "complete" the rule is, i.e. what is the proportion of examples having the predicted class C that is actually covered by the rule antecedent. The rule completeness measure, denoted Comp, is computed by the formula:

$$Comp = TP / (TP + FN) \quad (2)$$

In order to combine the CF and Comp measures one can define a fitness function such as:

$$Fitness = CF \times Comp \quad (3)$$

Although this fitness function does a good job in evaluating predictive performance, it has nothing to say about the comprehensibility of the rule. We can extend this fitness function (or any other focusing only on the predictive accuracy of the rule) with a rule comprehensibility measure in several ways. A simple approach is to define a fitness function such as:

$$Fitness = w1 * (CF * Comp.) + w2 * Simp$$

Where Simp is a measure of rule simplicity (normalized to take on values in the range 0..1) and w1 and w2 are user-defined weights. The Simp measure can be defined in many different ways, depending on the application domain and on the user. In general, its value is inversely proportional to the number of conditions in the rule antecedent-i.e., the shorter the rule, the simpler it is. Many other fitness functions are given by Giordana and Neri[2] and Goldberg[3].

Probabilities: For Mutation there are two probabilities, usually called as p_m, this probability will be used to judge whether mutation has to be done or not, when the candidate fulfills this criterion it will be fed to another probability and that is, locus probability that is on which point of the candidate the mutation has to be done.

For Crossover also two probabilities are there, one for the whether crossover has to be performed or not, i.e., p_c and other for finding the location, the point where, crossover must be done.

RESULTS AND DISCUSSION

In essence, data mining consists of the (semi-) automatic extraction of knowledge from data. This discovered knowledge should satisfy, three general properties namely, accuracy, comprehensibility and interestingness. The basic idea is to predict the value that some attribute(s) will take on in the future, based on previously observed data. In this context, the discovered knowledge should have a high predictive accuracy rate. Comprehensibility is necessary whenever the discovered knowledge is to be used for supporting a decision to be made by a human being. If the discovered knowledge is just a black box, which makes predictions without explaining them, the user may not trust it. The third property, knowledge interestingness, is the most difficult one to define and quantify, since it is, to a large extent, subjective. However, there are some aspects of knowledge interestingness that can be defined in objective terms. The aim of this paper is to present GA as a solution to discover truly interesting rules. GA can be used as an optimization technique to mine negative association rules, which are also as important as positive association rules.

REFERENCES

1. Alex A. Freitas, A survey of evolutionary algorithms for data mining and knowledge discovery postgraduate program in computer science, Brazil.

2. Giordana, A. and F. Neri, 1995. Search-intensive concept induction. Evolutionary Computation 3: 375-416.
3. Goldberg, D.E., 1989. Genetic Algorithms in search, optimization and machine learning. Addison-Wesley.
4. Pei, M., E.D. Goodman and W.F. Punch, 1997. Pattern Discovery from Data Using Genetic Algorithms. Proc. 1st Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD-97).
5. Rakesh Agrawal, 1993. Tomasz Imielinaki and Arun Swami, Mining association rules between sets of items in large databases, Proc. of the ACM SIGMOD Conference on Management of Data, Washington, D.C.
6. Rakesh Agrawal and Ramakrishnan Srikant, 1994. Fast Algorithms for Mining Association Rules, Proc. of the 20th Intl. Conf. on Very Large Databases, Santiago, Chile.
7. Wu, X., C. Zhang and S. Zhang, 2002. Mining Both Positive and Negative Association Rules. In: Proc. of ICML., pp: 658-665.
8. Teng, W., M. Hsieh and M. Chen, 2002. On the Mining of Substitution Rules for Statistically Dependent items. In: Proc. of ICDM., pp: 442-449.
9. Savasere, A., E. Omiecinski and S. Navathe, 1998. Mining for Strong Negative Associations in a Large Database of Customer Transactions. In: Proc. of ICDE., pp: 494-502.
10. Kumar, P.V. and J. Shrivastava, 2002. Selecting the Right Interestingness Measure for Association Patterns. In: Proc. of SIGKDD., pp: 32-41.