INFORMATION
SCIENCES
AN INTERNATIONAL JOURNAL

Information Sciences 163 (2004) 123–133

Multi-objective rule mining using genetic algorithms

Ashish Ghosh [a,*], Bhabesh Nath [b]

[a] *Machine Intelligence Unit, Indian Statistical Institute, 203 B.T. Road, Kolkata 700 108, India*

[b] *Department of Computer Science, Tezpur University, Tezpur, India*

Received 8 May 2002; accepted 17 March 2003

Abstract

Association rule mining problems can be considered as a multi-objective problem rather than as a single objective one. Measures like *support count*, *comprehensibility* and *interestingness*, used for evaluating a rule can be thought of as different objectives of association rule mining problem. *Support count* is the number of records, which satisfies all the conditions present in the rule. This objective gives the accuracy of the rules extracted from the database. *Comprehensibility* is measured by the number of attributes involved in the rule and tries to quantify the understandability of the rule. *Interestingness* measures how much interesting the rule is.

Using these three measures as the objectives of rule mining problem, this article uses a Pareto based genetic algorithm to extract some useful and interesting rules from any market-basket type database. Based on experimentation, the algorithm has been found suitable for large databases.

Keywords: Rule mining; Genetic algorithms; Multi-objective optimization

1. Introduction

With the rapid growth in size and number of available databases, mining for knowledge, regularities or high-level information from data became essential to support decision-making and predict future behavior [5,9]. Data mining techniques, used for achieving the above goals, can be classified into the following categories: classification, clustering, association rule mining, sequential pattern analysis, prediction, data visualization etc. [5,7,9,17,24].

Association rule mining is one of the important tasks of data mining intended towards decision support. Basically it is the process of finding some relations among the attributes/attribute values of a huge database. Inside the huge collection of data stored in a database, some kind of relationships among various attributes may exist. Discovery of such relationships will help in taking some decisions. Finding these relationships within a vast amount of data is not a simple job. The process of extracting these relationships is termed as association rule mining. These relationships can be represented as an *IF–THEN* statement. *IF* <some conditions are satisfied> *THEN* <predict some values of other attribute(s)>. The conditions associated in the IF part is termed as *Antecedent* and those with the THEN part is called the *Consequent*. In this article, we will refer them as A and C, respectively. So, symbolically we can represent this relation as $A \rightarrow C$ and each such relationship that holds between the attributes of records in a database fulfilling some criteria are termed as an association rule.

Another kind of rules, called classification rules, can also be represented in the same structure, but are completely different from association rules. In this kind of rules, the consequent part contains the values of only one attribute and this attribute is predefined. But there is no such restriction in case of association rules. Any attribute and any number of attributes may appear in either side. The only restriction is that the two parts should not have a common attribute, i.e., $A \cap C = \phi$.

A number of algorithms have been developed for searching these rules [2,4,19]. In the next section, we will give a brief introduction to this family of algorithms. However, these algorithms have their limitations. In the present work, we tried to visualize association rule mining as a *multi-objective* [14,27] problem rather than *single objective* one and tried to solve it using genetic algorithms [20,28]; thereby removing some of the limitations of the existing approaches.

The rest of this article is organized as follows. In Section 2 an overview of the existing association rule mining techniques is provided. Section 3 discusses the multi-objective nature of association rules mining problems, and provides a brief introduction to Pareto based genetic algorithms. Section 4 covers the details of the proposed work, including the encoding-decoding scheme. Analysis of results is put in Section 5. Finally, Section 6 includes the concluding remarks.

2. Association rule mining algorithms: An overview

Existing algorithms for mining association rules are mainly based on the approach suggested by Agrawal et al. [2,3]. Apriori [3], SETM [18], AIS [3], Pincer search [21], DIC [6] etc. are some of the popular algorithms based on this approach. These algorithms work on a binary database, termed as *market basket database*. On preparing the market basket database, every record of the original database is represented as a binary record where the fields are defined by a unique value of each attribute in the original database. The fields of this binary database are often termed as an *item*. For a database having a huge number of attributes and each attribute containing a lot of distinct values, the total number of items will be huge. Storing of this binary database, to be used by the rule mining algorithms, is one of the limitations of the existing algorithms.

Another aspect of these algorithms is that they work in two phases [2]. The first phase is for *frequent item-set* generation. Frequent item-sets are detected from all-possible item-sets by using a measure called *support count* (SUP) and a user-defined parameter called *minimum support*. Support count of an item set is defined by the number of records in the database that contains all the items of that set. If the value of *minimum support* is too high, number of frequent item sets generated will be less, and thereby resulting in generation of few rules. Again, if the value is too small, then almost all possible item sets will become frequent and thus a huge number of rules may be generated. Selecting better rules from them may be another problem.

After detecting the frequent item-sets in the first phase, the second phase generates the rules using another user-defined parameter called *minimum confidence* (which again affects the generation of rules). Confidence factor or predictive accuracy of a rule is defined as

$$\text{Confidence} = \text{SUP}(A \cup C)/\text{SUP}(A).$$

Another limitation of these algorithms is the encoding scheme where separate symbols are used for each possible value of an attribute. This encoding scheme may be suitable for encoding the categorical valued attributes, but not for encoding the numerical valued attributes as they may have different values in every record. To avoid this situation, some ranges of values may be defined. For each range of values an item is defined. This approach is also not suitable for all situations. Defining the ranges will create yet another problem, as the range of different attributes may be different.

Apart from these, another problem of these algorithms is that while generating the rules, the orders of the items also play an important role [1]. In these algorithms, it is not possible to generate a rule of the following format, I_1 I_2 I_6 I_8 I_{10} $I_{12} \rightarrow I_4$ I_5 I_9 (suffix indicates the order of appearance of the item in the binary database); though these relationships may be present inside the database. The proposed approach is free from this limitation also.

Existing algorithms, try to measure the quality of generated rule by considering only one evaluation criterion, i.e., *confidence factor* or *predictive accuracy*. This criterion evaluates the rule depending on the number of occurrence of the rule in the entire database. More the number of occurrences better is the rule. The generated rule may have a large number of attributes involved in the rule thereby making it difficult to understand [11]. If the generated rules are not understandable to the user, the user will never use them. Again, since more importance is given to those rules, satisfying number of records, these algorithms may extract some rules from the data that can be easily predicted by the user. It would have been better for the user, if the algorithms can generate some of those rules that are actually hidden inside the data. These algorithms do not give any importance towards the rare events, i.e., interesting rules [15,22].

In the present work we used the *comprehensibility* and the *interestingness* measure of the rules in addition to predictive accuracy. In the next section, we will discuss about them in detail. Using these three measures—comprehensibility, interestingness and the predictive accuracy, some previously unknown, easily understandable rules can be generated. Hence, the rule-mining problem is not a single objective problem; rather we visualize them as a multi-objective problem.

3. Multi-objective optimization and rule mining problems

It is always difficult to find out a single solution for a multi-objective problem. So it is natural to find out a set of solutions depending on non-dominance criterion [8,12,14,29]. At the time of taking a decision, the solution that seems to fit better depending on the circumstances can be chosen from the set of these candidate solutions. A solution, say a, is said to be dominated by another solution, say b, if and only if the solution b is better or equal with respect to all the corresponding objectives of the solution a, and b is strictly better in at least one objective. Here the solution b is called a *non-dominated* solution. So it will be helpful for the decision-maker, if we can find a set of such non-dominated solutions. Vilfredo Pareto suggested this approach of solving the multi objective problem. Optimization techniques based on this approach are termed as Pareto optimization techniques.

Based on this idea, several genetic algorithms were designed to solve multi-objective problems [13,25,26]. Multiple-Objective Genetic Algorithm (MOGA) [13] is one of them. Here the *chromosomes* are selected (using standard selection scheme, e.g. *roulette wheel* [28] selection) using the fitness value. Fitness value is calculated using their ranks, which are calculated from the non-dominance property of the chromosomes. The ranking step tries to find the non-dominated solutions, and those solutions are ranked as one. Among the rest of the chromosomes, if p_i individuals dominate a chromosome then its rank is assigned as $1 + p_i$. This process continues till all the chromosomes are ranked. Then fitness is assigned to the chromosomes such that the chromosomes having the smallest rank gets the highest fitness and the chromosomes having the same rank gets the same fitness. After assigning the fitness to the chromosomes, *selection*, *replacement*, *crossover* and *mutation* operators are applied to get a new set of chromosomes, as in standard GAs.

As mentioned earlier, in the present work we used the *comprehensibility* and the *interestingness* measure of rules in addition to predictive accuracy (which is already discussed in the previous section) as objectives of multi-objective GAs. Let us discuss them here. It is very difficult to quantify understandability or comprehensibility. A careful study of an association rule will infer that if the number of conditions involved in the antecedent part is less, the rule is more comprehensible. To reflect this behavior, an expression was derived as comp $= N -$ (number of conditions in the antecedent part) [11]. This expression serves well for the classification rule generation [16] where the number of attributes in the consequent part is always one. Since, in the association rules, the consequent part may contain more than one attribute, this expression is not suitable for the association rule mining. We require an expression where the number of attributes involved in both the parts of the rule has some effect. The following expression can be used to quantify the comprehensibility of an association rule,

$$\text{Comprehensibility} = \log(1 + |C|)/\log(1 + |A \cup C|).$$

Here, $|C|$ and $|A \cup C|$ are the number of attributes involved in the consequent part and the total rule, respectively.

Since association rule mining is a part of data mining process that extracts some hidden information, it should extract only those rules that have a comparatively less occurrence in the entire database. Such a surprising rule may be more interesting to the users; which again is difficult to quantify. For classification rules it can be defined by information gain theoretic measures [15]. This way of measuring interestingness for the association rules will become computationally inefficient. For finding interestingness the data set is to be divided based on each attribute present in the consequent part. Since a number of attributes can appear in the consequent part and they are not predefined, this approach may not be feasible for association rule mining. So a new expression is defined which uses only the support count of the antecedent and the consequent parts of the rules, and is defined as

$$\begin{aligned}\text{Interestingness} = {} & [\mathrm{SUP}(A \cup C)/\mathrm{SUP}(A)] \times [\mathrm{SUP}(A \cup C)/\mathrm{SUP}(C)] \\ & \times [1 - (\mathrm{SUP}(A \cup C)/|D|)].\end{aligned}$$

Here $|D|$ is the total number of records in the database.

This expression contains three parts. The first part, $[\text{SUP}(A \cup C)/\text{SUP}(A)]$, gives the probability of generating the rule depending on the antecedent part, the second part, $[\text{SUP}(A \cup C)/\text{SUP}(C)]$, gives the probability of generating the rule depending on the consequent part, and $(\text{SUP}(A \cup C)/|D|)$ gives the probability of generating the rule depending on the whole data-set. So complement of this probability will be the probability of ***not generating*** the rule. Thus, a rule having a very high support count will be measured as less interesting.

4. The proposed method

In the present work we tried to solve the association rule-mining problem with a Pareto based genetic algorithm. The first task for this is to represent the possible rules as chromosomes, for which a suitable encoding/decoding scheme is required. For this, two approaches can be adopted. In the *Pittsburgh approach* each chromosome represents a set of rules, and this approach is more suitable for classification rule mining; as we do not have to decode the consequent part and the length of the chromosome limits the number of rules generated. The other approach is called the *Michigan approach* where each chromosome represents a separate rule. In the original Michigan approach we have to encode the antecedent and consequent parts separately; and thus this maybe an efficient way from the point of space utilization since we have to store the empty conditions as we do not known a priori which attributes will appear in which part. So we followed a new approach that is better than this approach from the point of storage requirement. With each attribute we associate two extra tag bits. If these two bits are **00** then the attribute next to these two bits appears in the antecedent part and if it is **11** then the attribute appears in the consequent part. And the other two combinations, **01** and **10** will indicate the absence of the attribute in either of these parts. So the rule $ACF \rightarrow BE$ will look like **00A 11B 00C 01D 11E 00F**. In this way we can handle variable length rules with more storage efficiency, adding only an overhead of $2k$ bits, where k is the number of attributes in the database.

The next step is to find a suitable scheme for encoding/decoding the rules to/from binary chromosomes. Since the positions of attributes are fixed, we need not store the name of the attributes. We have to encode the values of different attribute in the chromosome only. For encoding a categorical valued attribute, the market basket encoding scheme is used. As discussed earlier this scheme is not suitable for numeric valued attributes. For a real valued attribute their binary representation can be used as the encoded value. The range of value of that attribute will control the number of bits used for it. Decoding will be simply the reverse of it. The length of the string will depend on the required accuracy of the value to be encoded. Decoding can be performed as:

$$\text{Value} = \text{Minimum value} + (\text{maximum value} - \text{minimum value}) \times \left(\left(\sum (2^{i-1} \times i\text{th bit value})\right)/(2^n - 1)\right)$$

Where $1 \leqslant i \leqslant n$ and n is the number of bits used for encoding; and *minimum* & *maximum* are minimum and maximum values of the attribute.

Using these encoding schemes values of different attributes can be encoded into the chromosomes. Since in the association rules an attribute may be involved with different relational operators [23], it is better to encode them also within the rule itself. For example, in one rule a numeric attribute A may be involved as $A \geqslant value_1$, but in another rule it may be involved as $A \leqslant value_2$. Similarly, a categorical attribute may be involved with either *equal to* $(=)$ or *not equal to* $(\neq)$. To handle this situation we used another bit to indicate the operators involved with the attribute. *Equality* and *not equality* are not considered with the numerical attribute. In this way the whole rule can be represented as a binary string, and this binary string will represent one chromosome or a possible rule.

After getting the chromosomes, various genetic operators can be applied on it. Presence of large number of attributes in the records will results in large chromosomes, thereby needing multi-point crossover.

There are some difficulties to use the standard multi-objective GAs for association rule mining problems. In case of rule mining problems, we need to store a set of better rules found from the database. If we follow the standard genetic operations only, then the final population may not contain some rules that are better and were generated at some intermediate generations. It is better to keep these rules. For this task, a separate population is used [10]. In this population no genetic operation is performed. It will simply contain only the non-dominated chromosomes of the previous generation. The user can fix the size of this population. At the end of first generation, it will contain the non-dominated chromosomes of the first generation. After the next generation, it will contain those chromosomes, which are non-dominated among the current population as well as among the non-dominated solutions till the previous generation.

The suggested approach will work as follows:

1. Load a sample of records from the database that fits in the memory.
2. Generate N chromosomes randomly.
3. Decode them to get the values of the different attributes.
4. Scan the loaded sample to find the support of antecedent part, consequent part and the rule.
5. Find the confidence, comprehensibility and interestingness values.
6. Rank the chromosomes depending on the non-dominance property.
7. Assign fitness to the chromosomes using the ranks, as mentioned earlier.

8. Bring a copy of the chromosomes ranked as 1 into a separate population, and store them if they are non-dominated in this population also. If some of the existing chromosomes of this population become dominated, due to this insertion, then remove the dominated chromosomes from this population.
9. Select the chromosomes, for next generation, by roulette wheel selection scheme using the fitness calculated in Step 7.
10. Replace all chromosomes of the old population by the chromosomes selected in Step 9.
11. Perform multi-point crossover and mutation on these new individuals.
12. If the desired number of generations is not completed, then go to Step 3.
13. Decode the chromosomes in the final stored population, and get the generated rules.

5. Implementation and results

The proposed technique was implemented on different data sets with satisfactory results. Here we present the results on one such data set having 38 attributes and 8330 records. Crossover and mutation probabilities were taken respectively as 0.8 and 0.02; 5 point crossover operator was used and the population size was kept fixed as 40. Sample size and number of rules generated are put in the following table.

Sample size	Number of generations	Number of rules generated
1000	100	24
	200	31
	300	31
1000	100	35
	200	40
	300	40
1000	100	27
	200	36
	300	37
2000	100	35
	200	40
	300	40
2000	100	35
	200	40
	300	40

From the rule sets generated for different samples and for different number of generations it is observed that after 200 generations it ceases to generate more rules; in other words after that number of generations the GA converges. From the results given above it can be seen that only for the third sample, it gives an extra rule at the cost of 100 additional generations. Moreover, only a very few number of attributes (3–4 attributes on both the antecedent and consequent parts) got involved in the rules, which means that all the attributes are not equally important; and the rules are simple to understand (comprehensible). The generated rules were not that much interesting (interestingness value was order of 0.005).

If the confidence of the rule is used as one measure, sometimes some rules with $\mathrm{SUP}(A) = 1$, $\mathrm{SUP}(C) = 1$, and $\mathrm{SUP}(AC) = 1$ may be generated. That rule will have a confidence 100%. So there is a chance that the rule may be declared as a non-dominated rule. But the records satisfying that rule may be noise also. Current algorithms do not face this problem, because the user parameter called minimum support eliminates the probability of generation of such rules. Instead of the confidence, we used the support of the rule as one measure to evaluate the rule thereby overcoming this problem.

6. Discussions and conclusion

Association rule mining is viewed as a *multi-objective* problem rather than *single objective* one as assumed by most of the existing algorithms. This article uses a *Pareto* based genetic algorithm to solve the multi-objective rule mining problem using three measures—comprehensibility, interestingness and the predictive accuracy. We adopted a variant of the *Michigan* approach to represent the rules as chromosomes, where each chromosome represents a separate rule.

To improve the efficiency of this algorithm, some refinement may be required. For example, this algorithm works on a sample of the original database, and the sample may not truly reflect the actual database. In the present work, we used the *random sampling* method. Using some other sampling techniques like *regression based* sampling or *cluster-based* sampling, a good sample can be found. A perfect sample will improve the correctness of the rules generated by the algorithm. Moreover, we tested the approach only with the numerical valued attributes. It must be tested with the categorical attributes also.

References

[1] J.M. Adamo, Data Mining for Association Rules and Sequential Patterns, Springer-Verlag, New York, 2001.

[2] R. Agrawal, R. Srikant, Fast algorithms for mining association rules, in: Proceedings of the 20th International Conference on Very Large Databases, Santiago, Chile, 1994.

[3] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases. in: Proceedings of ACM SIGMOD Conference on Management of Data, 1993, pp. 207–216.

[4] A.M. Ayad, A new algorithm for incremental mining of constrained association rules. Master Thesis, Department of Computer Sciences and Automatic Control, Alexandria University, 2000.

[5] M.J. Berry, G. Linoff, Data Mining Techniques for Marketing, Sales and Customer Support, John Wiley and Sons, New York, 1997.

[6] S. Brin, et al., Dynamic itemset counting and implication rules for market basket data, in: Proceedings of ACM SIGMOD International Conference on Management of Data, Tucson, AZ, 1997.

[7] K.J. Cios, W. Pedrycz, R.W. Swiniarski, Data Miming Methods for Knowledge Discovery, Kluwer Academic Publishers, Boston, MA, 2000.

[8] C.A.C. Coello, A comprehensive survey of evolutionary-based multi-objective optimization technique, Knowledge and Information Systems 1 (1999) 269–308.

[9] M.S. Chen, J. Han, P.S. Yu, Data mining an overview from a database perspective, IEEE Transactions on Knowledge and Data Engineering 6 (1996) 866–883.

[10] L.J. Eshelman, J.D. Schaffer, Preventing premature convergence in genetic algorithms by preventing incest., in: Proceedings of the 4th International Conference on GA, 1991.

[11] M.V. Fedelis et al., Discovering comprehensible classification rules with a genetic algorithm, in: Proceedings of Congress on Evolutionary Computation, 2000.

[12] C.M. Fonseca, P.L. Fleming, An overview of evolutionary algorithms in multi-objective optimization, Evolutionary Computation 3 (1995) 1–16.

[13] C.M. Fonseca, P.L. Fleming, Genetic algorithms for multi-objective optimization: formulation, discussion and generalization, in: Fifth International Conference on Genetic Algorithms 1993, pp. 416–423.

[14] A.A. Freitas, A survey of evolutionary algorithms for data mining and knowledge discovery, in: A. Ghosh, S. Tsutsui (Eds.), Advances in Evolutionary Computing, Springer-Verlag, New York, 2003, pp. 819–845.

[15] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer-Verlag, New York, 2002.

[16] A.A. Freitas, Understanding the crucial role of attribute interaction in data mining, Artificial Intelligence Review 16 (2001) 177–199.

[17] W. Frawley, G. Piatasky-Saprio, C. Matheus, Knowledge Discovery in Data Bases: An Overview, AAAI/MIT Press, 1991.

[18] A. Houtsma and M. Swami, Set-oriented mining of association rules, Research Report, 1993.

[19] J. Hipp et al., Algorithms for association rule mining—a general survey and comparison, SIGKDD Explorations 2 (1) (2000) 1.

[20] C. Lance, Practical Handbook of Genetic Algorithms, CRC Press, 1995.

[21] D.I. Lin and Z.M. Kedem, Pincer-search: an efficient algorithm for discovering the maximal frequent set, Proceedings of 6th European Conference on Extending Database Technology, 1998.

[22] E. Noda et. al., Discovering interesting prediction rules with a genetic algorithm, Proceedings of the Congress on Evolutionary Computation 1999, pp. 1322–1329.

[23] J.R. Oliver, Discovering individual decision rules: an application of genetic algorithms, Fifth International Conference on Genetic Algorithms, Urbana-Champaign, 1993.

[24] J. Han, M. Kamber, Data Mining—Concepts and Techniques, Morgan Kaufmann, 2001.

[25] J.D. Shaffer, Multiple objective optimization with vector evaluated genetic algorithm, First International Conference on Genetic Algorithms 1985, pp. 93–100.

[26] K. Deb, Multi-objective Optimization using Evolutionary Algorithms, Wiley, New York, 2001.

[27] H. Vafaie, K. DeJong, Improving the performance of a rule induction system using genetic algorithms, in: R. Michalski, G. Tecuci (Eds.), Machine Learning—A Multistrategy Approach, vol. IV, Morgan Kaufmann Publishers, San Francisco, 1994, pp. 453–470.

[28] M.D. Vose, The Simple Genetic Algorithm: Foundations and Theory, MIT Press, Cambridge, 1999.

[29] E. Zitzler, K. Deb, L. Thiele, Comparison of multi-objective evolutionary algorithms: empirical results, Evolutionary Computation 8 (2000) 125–148.