Advances in Information Mining, ISSN: 0975–3265, Volume 1, Issue 2, 2009, pp-01-04

Optimized association rule mining using genetic algorithm

Anandhavalli M.*, Suraj Kumar Sudhanshu, Ayush Kumar and Ghose M.K.
Department of Computer Science Engineering, Sikkim Manipal Institute of Technology, East Sikkim, India, anandhim@yahoo.com, suraj.smit06@gmail.com, sudhanshu.smit06@gmail.com, mkghose2000@yahoo.com

Abstract- In general the rule generated by association rule mining algorithms like priori, partition, pincer-search, incremental, border algorithm etc, does not consider negation occurrence of the attribute in them and also these rules have only one attribute in the consequent part. By using Genetic Algorithm (GAs) the system can predict the rules which contain negative attributes in the generated rules along with more than one attribute in consequent part. The major advantage of using GAs in the discovery of prediction rules is that they perform global search and its complexity is less compared to other algorithms as the genetic algorithm is based on the greedy approach. The main aim of this paper is to find all the possible optimized rules from given data set using genetic algorithm.
Keywords - Genetic Algorithm (GA), Association Rules, Support, Confidence, Data Mining

Introduction

The rapid development of computer technology, especially increased capacities and decreased costs of storage media, has led businesses to store huge amounts of external and internal information in large databases at low cost. Mining useful information and helpful knowledge from these large databases has thus evolved into an important research area [1, 2]. Among the areas of data mining, the problem of deriving associations from data has received a great deal of attention.

Association rules are used to identify relationships among a set of items in database. These relationships are not based on inherent properties of the data themselves (as with functional dependencies), but rather based on co-occurrence of the data items. Association rules were introduced first in [4]. The subsequent paper [3] is considered as one of the most important contributions to the subject. Its main algorithm, Apriori, not only influenced the association rule mining community, but it affected other data mining fields as well.

In this paper, an attempt has been made to generate optimized association rules with multiple consequents by applying genetic algorithm on the frequent itemsets generated by Apriori association rule mining method.

A. Association Rule

In general, the association rule is an expression of the form X=>Y, where X is antecedent and Y is consequent. Association rule shows how many times Y has occurred if X has already occurred depending on the support and confidence value.

Support: It is the probability of item or item sets in the given transactional data base: Support(X) = n(X)/n where n is the total number of transactions in the database and n(X) is the number of transactions that contains the item set X.

Confidence: It is conditional probability, for an association rule X=>Y and defined as

Confidence(X=>Y) = support(X and Y)/support(X).

All the traditional association rule mining algorithms were developed to find positive associations between items. Positive associations refer to associations between items existing in transactions. In addition to the positive associations, negative associations can provide valuable information. In practical there are many situations where negation of products plays a major role. For examples:

If I study in SMIT then I do not study in HIMALAYAN PHARMACY.

If I am a male then I am not a female.

Negative association rule is an implication of the form X=>¬Y where X and Y are item sets and X∩Y=Φ.

Mining association rules can be broken down into the following two sub-problems:

1. Generating all itemsets that have support greater than, or equal to, the user specified minimal support. That is, generating all large itemsets.
2. Generating all the rules that have minimum confidence.

We can generate the association rule with more than one number of consequent items is generated by the following method:

1. Find the rule in which number of consequents =1.
2. For the given rules p(x → y) and p(x → z), the rule p(x → yz) is generated by the intersection of both the association rules and get a new rule p(x → yz) =p (xyz)/p(x).

B. Genetic Algorithm

Genetic Algorithm (GA) [5] was developed by Holland in 1970. It incorporates Darwinian evolutionary theory with sexual reproduction. GA is stochastic search algorithm modelled on the process of natural selection, which underlines biological evolution [6]. GA has been successfully applied in many search, optimization, and machine learning problems. GA works in an iteration manner by generating new populations of strings from old ones. Every string is the encoded binary, real etc., version of a candidate solution. An evaluation function associates a fitness measure to every string

indicating its fitness for the problem. Consider the following example,

If Desktop and UPS then printer and not laptop.

It can be represented with two types in the binary string.

Method 1: *1 1 1 0*

This type of representation is relative to position. Presence of 1 at i^{th} position indicates occurrence of the item[i]. Similarly presence of 0 at j^{th} position indicates absence of item[j]. In this method, n bits are used to represent each transaction.

Method 2: *00 1 01 1 10 1 11 0*

If there are n products then we use (log n/log 2) bits to represent each item. One extra bit is needed to represent the presence or absence of item. Here two bits strings represents product id and one bit represent presence or absence of that product. Here 00 is used for desktop, 01 is used for UPS, 10 is used for printer and 11 is used for laptop. This method of representation is most popular because in case if a database contains n item (generally n is a large quantity) and mostly transaction involves p items. Then we need only p*(logn+1) bit to store each transaction.

Standard GA apply genetic operators such selection, crossover and mutation on an initially random population in order to compute a whole generation of new strings. GA runs to generate solutions for successive generations. The probability of an individual reproducing is proportional to the goodness of the solution it represents. Hence the quality of the solutions in successive generations improves. The process is terminated when an acceptable or optimum solution is found. GA is appropriate for problems which require optimization, with respect to some computable criterion.

The function of genetic algorithm is as follows:

1) *Selection:* The selection of the member from the population can be done with the help of Roulette Wheel sampling method. Roulette Wheel selection is a process of choosing members from the population of chromosomes in a way that is proportional to their fitness. It does not guarantee that the fittest member goes through to the next generation, merely which it has a very good chance of doing so. It works like this: Imagine that the population's total fitness score is represented by a pie chart, or roulette wheel. Now you assign a slice of the wheel to each member of the population. The size of the slice is proportional to that chromosomes fitness score. i.e. the fitter a member is the bigger the slice of pie it gets. Now, to choose a chromosome all you have to do is spin the ball and grab the chromosome at the point it stops.

2) *Crossover* is performed by selecting a random gene along the length of the chromosomes and swapping all the genes after that point. For example, given two chromosomes,

10001001110010010

01010001001000011

Choose a random bit along the length, say at position 9, and swap all the bits after that point. The resultant chromosomes become

10001001101000011

01010001010010010

3) *Mutation* alters the new solutions so as to add stochasticity in the search for better solutions. This is the chance that a bit within a chromosome will be flipped (0 becomes 1, 1 becomes 0). Whenever chromosomes are chosen from the population the algorithm first checks to see if crossover should be applied and then the algorithm iterates down the length of each chromosome mutating the bits if applicable.

The population is ranked with the help of fitness function. We apply genetic algorithm on the selected population from the database and compute the fitness function after each step until the genetic algorithm is terminated.

For a rule: if item set A then item set B where A and B contains some item present or its negation. The predictive performance of a rule is summarized in Table 1. From Table 1, it is known that higher the values of TP and TN and lower the values of FP and FN, the better is the rule.

Confidence Factor, CF = TP / (TP + FN)

We also introduce another factor completeness measure for computing the fitness function.

Comp=TP/ (TP+FP) Fitness=CF *Comp

The fitness function shows that how much we near to generate the rule.

Methodology

In this paper the genetic algorithm is applied over the rules fetched from Apriori association rule mining. The proposed method for generating association rule by genetic algorithm is as follows:

1. Start
2. Load a sample of records from the database that fits in the memory.
3. Apply Apriori algorithm to find the frequent itemsets with the minimum support. Suppose A is set of the frequent item set generated by Apriori algorithm.
4. Set B= Φ where B is the output set, which contains the association rule.
5. Input the termination condition of genetic algorithm.
6. Represent each frequent item set of A as binary string using the combination of representation specified in method 2 above.
7. Select the two members from the frequent item set using Roulette Wheel sampling method.

8. Apply the crossover and mutation on the selected members to generate the association rules.
9. Find the fitness function for each rule x→y and check the following condition.
10. if (fitness function > min confidence)
11. set B = B U {x →y}
12. If the desired number of generations is not completed, then go to Step 3.
13. Stop

If a data set contains p items, the total number of association rule is 3p-2(p+1)+1 (without considering the negation of item). The total no. of rule generated for a database with n item is $(2p-2)*2p+(2(p-1)-2)*2(p-1)+\ldots+(22-2)*22$. In general we can say that number of rule generated including the negation of itemset is

$$\sum_{i=2}^{p}(2^i-2)*(2^i)$$

Results

We use the Lens database from UCI (http://www.ics.uci.edu/~mlearn) to show the effectiveness of the proposed algorithm. Lens database contains the following four attributes namely,

1. Age of the patient with three values:
(i) young, (ii) pre- presbyopic, (iii) presbyopic
2. Spectacle prescription with two values:
(i) myope (ii) hypermetrope
3. Astigmatic with two values:
(i) no (ii) yes
4. Tear production rate with values:
(i) reduced, (ii) normal

For each attribute we need $\log_2 m$ bits, where m is the number of attribute values. Let us represent the member of data base in the binary string with the hybrid combination of above representation.

For example, the binary string 01 00 1 01 0 11 1 11 in which bold letter represent the presence (11), absence (00), not consider (01). The above binary string is encoded as
Age = not pre-presbyopic,
Spectacle prescription: =not considered,
Astigmatic: =yes,
Tear production rate: =normal
The columns in the database are subjects and their numerical value depends on attributes which are selected is shown in Table 2.
As described earlier, the implementation of GAs is applied to the rules obtained by applying Apriori association rule mining on Lense database. The database is chosen randomly. The crossover and mutation probabilities were taken respectively as 0.1 and 0.005.
The frequent item set generated for the given database in Table 2 is as follows.

{a=1},{a=2},{a=3},{b=1},{b=2},{c=1},{c=2},{d=1},{d=2},
{a=1,b=2},{a=1,c=1},{a=1,d=2},{a=2,b=1},{a=2,b=2},{a=2,c=2},
{a=2,d=1},{a=3,b=1},{a=3,c=1}{a=3,d=1},{b=1,c=1},{b=1,d=1},
{b=2,c=1},{b=2,c=c=2},{c=1,d=1},{c=1,d=2},{c=2,d=1},
{a=1,b=2,c=1},{a=1,b=2,d=2},{a=1,c=1,d=2}, {a=2,b=1,d=1},
{a=2,c=2,d=1},{a=3,b=1,c=1},{a=3,b=1,d=1},{a=3,c=1,d=1},
{b=1,c=1,d=1},{b=2,c=1,d=2},{a=1,b=2,c=1,d=2},{a=3,b=1,c=1,d=1}.

Here a, b, c, d represent the age of patients, spectacle prescription, astigmatic, and tear production rates respectively. The numerical value represents the attribute value of corresponding attribute.
Table 3 shows the rules evolved from generated frequent itemsets with minimum support >=20% and minimum confidence >=60%.

Conclusion and future work

We have dealt with a challenging association rule mining problem of finding optimized association rules. The frequent itemsets are generated using the Apriori association rule mining algorithm. The genetic algorithm has been applied on the generated frequent itemsets to generate the rules containing positive attributes, the negation of the attributes with the consequent part consists of single attribute and more than one attribute. The results reported in this paper are very promising since the discovered rules are of optimized rules. We also intend to extend the proposed algorithm in this paper to minimize the complexity of the genetic algorithm and scanning of database by applying Baye's theorem on the generated rule.

References

[1] Agrawal R., Imielinksi T. and Swami A. (1993) *Database mining: a performance perspective, IEEE Transactions on Knowledge and Data Engineering* 5 (6), 914–925.
[2] Chen M.S., Han J. and Yu P.S. (1996) *Data Mining: An Overview from a Database Perspective, IEEE Trans. Knowledge and Data Eng.*, 866-883.
[3] Agrawal R. and Srikant R. (1994) *Fast algorithms for mining association rules. In VLDB'94, Santiago, Chile*, 487-499.
[4] Agrawal R., Imielinski T. and Swami A. (1993) *Mining Association rules between sets of items in large databases, In the Proc. of the ACM SIGMOD Int'l Conf. on Management of Data (ACM SIGMOD '93), Washington, USA*, 207-216.
[5] Pei M., Goodman E.D., Punch F. (2000) *Feature Extraction using genetic algorithm, Case Center for Computer-Aided Engineering and Manufacturing W. Department of. Computer Science*.
[6] Stuart J. Russell, Peter Norvig (2008) *Artificial Intelligence: A Modern Approach*.

Table 1- Predictive performance matrix of a rule

Predicted class/ actual class	Item set A	Not Item set A
Item set B	TP	FP
Not Item set B	FN	TN

The labels in each quadrant of the matrix have the following meaning:
TP = True Positives = Number of examples satisfying item set A and item set B
FN = False Positives = Number of examples satisfying item set A but not item set B
FP = False Negatives = Number of examples not satisfying item set A but satisfying item set B
TN = True Negatives = Number of examples not satisfying item set A nor item set B

Table 2- Glimpse of lens database

Instance id /attribute	age of the patient	spectacle prescription	Astigmatic	tear production rate
1	1	2	2	1
2	2	2	2	2
3	1	2	1	2
4	3	1	1	1
5	3	1	1	1
6	2	1	1	1
7	2	1	2	1
8	2	2	2	1
9	3	1	1	2
10	1	2	1	2

Table 3- Association rules generated by genetic algorithm

Association rules	Support (%)	Confidence (%)
Age= Young and spectacle prescription:= hypermetrope astigmatic=no and tear production rate:=not reduced	20	66.66
age of the patient:= pre-presbyopic spectacle prescription:= myope and tear production rate:=not normal	20	100
age of the patient:= pre-presbyopic and spectacle prescription:= myope astigmatic=no	30	100
age of the patient:= pre-presbyopic and spectacle prescription:= myope tear production rate:=reduced	20	100
tear production rate:=reduced and astigmatic=no age of the patient:= pre-presbyopic	20	66.66