CHAPTER 4

ASSOCIATION RULE MINING BASED ON GENETIC ALGORITHM

4.1 Introduction

In recent years, the growth of information has proceeded at an explosive rate. It is beyond the capability of the human mind to analyse a huge amount of data. Due to the size of the data, it is extremely difficult to draw meaningful conclusions about the data. Data mining technology has been introduced to overcome the above scenario with the large amount of data; it will mine the knowledge and produce interesting patterns.

The genetic algorithm has been applied on data mining step of the KDD process to get more meaningful data. An important area of data mining is strong rule generation. Using Genetic Algorithm, the rules generated by Association Rule Mining have been optimized. In this proposed method, a general Apriori algorithm is used to generate the rules. After that the genetic algorithm for optimization techniques is used.

Genetic algorithm is one of the best ways to optimize the rules. A new fitness function has been designed for the optimization of the rule set that uses the concept of supervised learning. Then the GA will be able to generate the stronger rule set.

Concepts of mining associations are given as follows. Let I = {I1 , I2... Im } be a set of items and D = {t1,t2,…………tn} be a set of transactions, where t_i is a set of items such that $t_i \subset I$. An association rule is an implication of the form $X \Rightarrow Y$, where $X,Y \subset I$ and $X \cap Y = ¢$.

The rule $X \Rightarrow Y$ holds in the set D with support and confidence, where support is the percentage of transactions in D that contain both X and Y and confidence is the percentage of transactions in D containing X that also contain Y. An association rule must satisfy user-set minimum support (min_sup) and minimum confidence (min_conf).

The rule $X \Rightarrow Y$ is called a strong association rule if support ≥ min_sup and confidence ≥ min_conf. Usually association analysis is not given decision attributes so that find association and dependence between attributes to the best of our abilities. But the aimless analysis may take much time and space.

Decision attributes determined can reduce the amount of candidate sets and searching space, and then improve the efficiency of algorithms to some extent. In addition, users are not interested in all association rules, but they are just concerned about the associations among condition attributes and decision attributes.

The association rules have been mined with decision attributes, in that, consider the attributes, which users are concerned about as decision attributes and other attributes as condition attributes. If mining association rules form continuous attributes data, the continuous attributes have to be discretized first. The essence of discretization is to use the selected cut-points to divide the values of the continuous attributes into intervals.

The methods of dividing determine the quality of association rules. Several methods are there to optimize the rule, while the genetic algorithm has good optimization properties. Genetic algorithm is used to search the cut-points of the continuous attributes.

4.2 Genetic Algorithm

The Genetic Algorithm was developed by John Holland in 1970 and it is based on the genetic processes of biological organisms. Over many years, the principles of natural selection, natural population evolve and survival of the fittest was first clearly stated by Charles Darwin in the Origin of Species. Genetic Algorithm is an adaptive method which may be used to solve search and optimization problems [62] & [93].

Genetic algorithm is a type of searching algorithm. It searches a solution space to find an optimal solution to a problem. Genetic Algorithms are powerful in their ability to model both qualitative and quantitative search spaces. The algorithm creates a “population” of possible solutions to the problem, and lets them “evolve” over multiple generations to find better and better solution.

Algorithm stars with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and are used to form a new population. The unsuccessful individuals would result in a value of zero for the fitness function. Successful individuals result in a maximum value for the fitness function.

Some individuals result in intermediate values of the fitness function. Individuals with higher fitness have a higher probability of being selected for mating, and individuals having low fitness are killed off with a low probability of mating. The genetic processes of crossover and mutation are applied to the individuals in the mating population, and a new generation is created.

All offsprings are made by crossover. If crossover probability is 100%, and new generation is made from exact copies of chromosomes from old population. If the crossover probability is 0% and the mutation is performed, then the part of chromosome is changed. If mutation probability is 100%, whole chromosome will be changed. If it is 0%, no change has occurred.

The algorithm operates through a simple cycle

- Creation of a population of strings.
- Evolution of each string.
- Selection of the best string.
- Genetic manipulation to create a new population of strings.

This new form of solution is called offspring, which are selected according to their fitness and this process is repeated until the condition is satisfied.

4.2.1. Genetic Algorithm operators

The Genetic operators determine the search capability and convergence of the algorithm. Genetic operators hold the selection, encoding of chromosomes, crossover and mutation on the population and generate the new population. In selection, Chromosomes are selected from the population. The problem is how to select these chromosomes. According to Darwin's evolution theory, the best one should survive and create new offspring. There are many methods to select the best chromosomes.

For example, roulette wheel selection, Boltzmann selection, tournament selection, rank selection and, steady state selection is some of the methods for selecting the better chromosomes. Roulette wheel selection is a way of choosing members from population of chromosomes and there is no guarantee that the fittest member goes through to the next generation. To choose a chromosome, spin the ball and grab the chromosome at the point it stops.

Each chromosome has one binary string and each bit in this string represents some characteristic of the solution. There are many ways of encoding, and it depends mainly on the solved problem. For example, one can encode directly integer or real numbers; sometimes it is useful to encode some permutations and so on. The most used way of Encoding of chromosomes is a binary string and it is shown in the figure 4.1

Chromosome 1	1101100100110110
Chromosome 2	1101111000011110

Figure 4.1 Encoding of Chromosomes

Crossover selects gene from parent chromosomes and creates a new offspring. There are different ways like single point crossover and other is multipoint crossover. Single point crossover, is to choose a random crossover point and everything before this point copy from a first parent and then everything after a crossover point copy from the second parent. In multipoint cross over more than one crossover points will be chosen. The figure 4.2 shows the example of single point crossover.

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

Figure 4.2 Crossover of Chromosomes

Next to crossover is Mutation. The mutation operator introduces a certain amount of randomness to the search. It can help the search to find the better solutions. Mutation changes randomly the new offspring.

The mutation depends on the encoding as well as the crossover. For binary encoding we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Figure 4.3 shows the mutation operation.

Original offspring 1	1101111000011110
Original offspring 2	1101100100110110
Mutated offspring 1	1100111000011110
Mutated offspring 2	1101101100110110

Figure 4.3 Mutation of Chromosomes

4.2.2 Steps in Genetic Algorithm

1. Generate random population of n chromosomes.
2. Evaluate the fitness function f(x) of each chromosome x in the population
3. Create a new population by repeating following steps until the new population is completed

   Selection: According to the fitness function, select two parent chromosomes from a population. Chromosomes which are having better fitness will be selected.

   Crossover: Cross over the parents to form a new offspring with a crossover probability. Offspring is an exact copy of parents, if no crossover was performed

   Mutation: Mutate new offspring at each position in chromosome with the a mutation probability.

   Accepting: Place new offspring in a new population

4. Use the newly generated population for a further processing of algorithm.
5. If the end condition is satisfied, stop further processing of the algorithm, and return the best solution in current population.
6. Go to step 2.

A typical flowchart of a genetic algorithm is shown in Figure 4.4. One iteration of the algorithm is referred to as a generation.

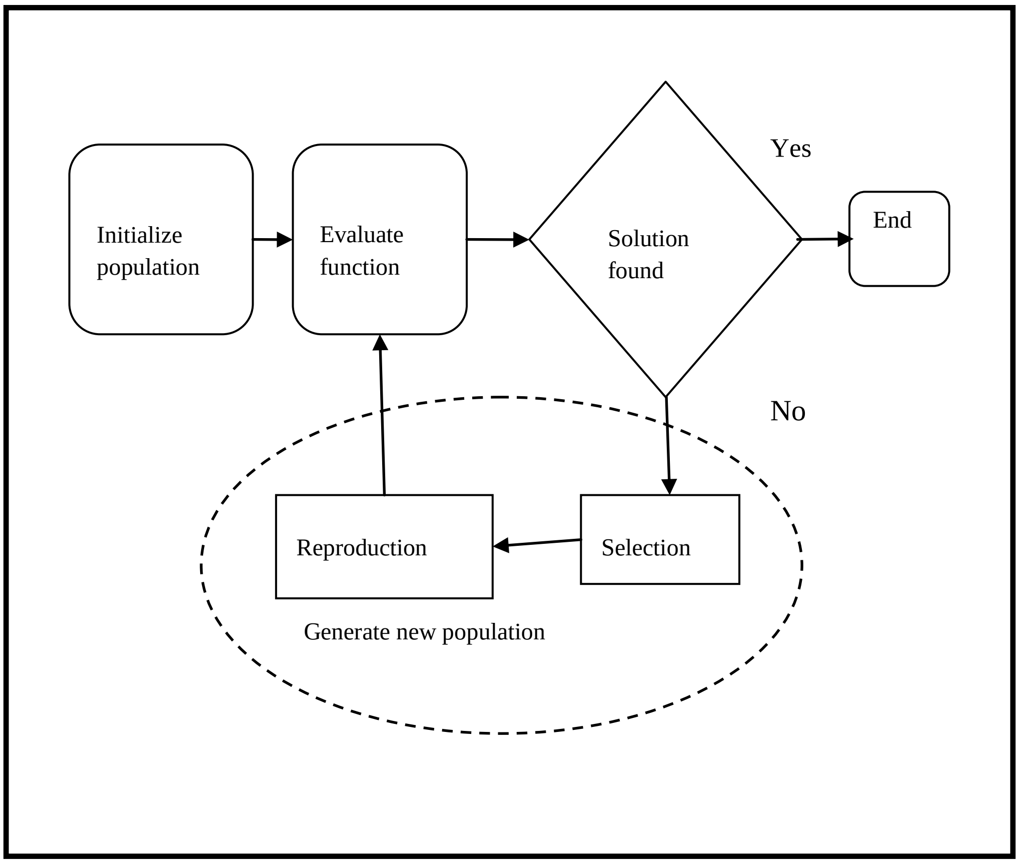

Figure 4.4 Flow chart of Genetic Algorithm

4.3 Rule Optimization by Genetic Algorithm

Rule is generated using Apriori Algorithm and association rule is optimized by applying genetic algorithm. It involves data encoding, fitness function, selection strategies, and genetic operation

4.3.1 Data encoding

Abandon Dataset basically has been used for the classification. In this Database, 6 attributes are taken. It is provided by the MCI. Data is encoded in the form of Binary representation technique 0 and 1, because Genetic Algorithm will not directly work on the raw data.

4.3.2 Fitness function

The most important part of Genetic Algorithm is the design of Fitness function:

$$f(x) = \frac{\text{Support}(x)}{\text{min\_support}} \begin{cases} p(\text{support}(x) > \text{min\_support}) \\ q(\text{support}(x) < \text{min\_support}) \end{cases}$$

Support is the support of new rule generated through genetic operations. In normal case, the value of q (Support(x) < min_support) is rejected for the better performance of genetic algorithm.

The class-learned classifiers have been used for the prediction for those rejected values which are near to the maximum value.

The value of q class is divided into two parts C_1 and C_2.

$q = \{C_1, C2\}$

C_1 = {those value or Data min_support less than 0.5}
C_2 = {those value or Data min_support greater than 0.5}

Now

$$f(q) = \frac{\text{Support }(C_2)}{\text{min\_support}} \begin{cases} \alpha = (\text{Support }(C_2)) > C_1 \\ \beta = (\text{Support}(C_2)) = C_1 \end{cases}$$

4.3.3 Selection Strategies

The selection strategy is based on the basis of individual fitness, and concentration p_i is the probability of selection of an individual whose fitness value is greater than one, and f(α) is that value whose fitness value is less than one, but near to the value of 1.

Now

$$Pi = \frac{f(x_i)}{\sum_1^M f(x_j)} \quad e^{-\alpha f(\alpha)}$$

Where α is an adjustment factor.

4.3.4 Genetic operation

The Genetic operators determine the search capability, and convergence of the algorithm. Genetic operators hold the selection, crossover and mutation on the population and generate the new population.

In selection operation, the algorithm restores each chromosome in the population to the corresponding rule, and then calculate selection probability pi for each rule, based on above formula. In crossover, multipoint crossovers are used, and classified the cut point of each continuous attributes into one group.

The crossover carried out between the corresponding groups of two individuals by a certain rate. In mutation operation, any bit in the chromosomes is mutated by a certain rate, is, changing “0”to”1”,”1”to”0”.

The frequent rules are generated according to the fitness function and genetic operators. In order to mine the strong association rules finally, these rules must be extracted again. Extraction criteria are: output the rule which meets the minimum confidence given by users, otherwise abandon it.

4.4. Rule Extraction

Using Apriori algorithm, Frequent Itemsets have been generated, and using Apriori association property, association rules among the items have been generated. By using the genetic operators like selection, encoding, crossover and mutation, association rules have been optimized.

Finally these association rules are extracted again by using certain criteria and that extraction criteria are: output the rule which meets the minimum confidence given by users, otherwise abandon it.

4.5. Implementation and Result

The experiment uses abandon dataset obtained from UCI machine learning repository. The data set has 4000 samples. It is composed of a discrete attribute and 8 continuous attributes. Here mining is applied to such association rules $X \Rightarrow Y$ that Y was age.

The initial population size taken is 1000, (ie) 1000 parent chromosomes have been taken for this experiment, and the maximum number of generations created is fixed as 500, so that, not more than 500 generations are created, and probability of cross over fixed as 0.7. According to that, crossover has happened and the mutation probability is fixed as 0.05 for this experiment.

The setting of parameters:

Initial population size—1000;

Maximum number of generations—500;

Probability of crossover—0.7;

Mutation probability—0.05;

Initial range of real-valued Strings—(0.000 067; 0.0002);

Table 4.1. Performance Evaluation between Apriori and Genetic Algorithm

Attribute	**A**	**B**	**C**	**D**	**E**	**F**
Min support	2	4	6	8	9	7
Min confidence	0.1	0.2	0.3	0.1	0.2	0.5
Apriori algorithm	121	137	50	98	84	28
GA	94	115	42	75	70	22

The experiment was executed on Celeron CPU 3.0GHz machine and software was MATLAB. Table 4.1 shows the result of those rule generated by Apriori and genetic algorithm. Here, 6 different values of support and confidence for 6 different attributes have been specified and number of rules generated by Apriori and Genetic algorithm were calculated and presented in the table 4.1.

When the min support for attribute A is 2 and min confidence is 0.1, the Apriori Algorithm generates 121 association rules, whereas Genetic Algorithm generates 94 association rules. For the attribute C, with the minimum support 6 and the minimum confidence 0.3, Apriori algorithm generates 50 association rules, and genetic algorithm generates 42 association rules.

For the attribute D, with the minimum support 8 and the minimum confidence 0.1, Apriori algorithm generates 98 association rules, and genetic algorithm generates 75 association rules. . For the attribute E, with the minimum support 9 and the minimum confidence 0.2, Apriori algorithm generates 84 association rules and genetic algorithm generates 70 association rules.

When the min support for F is 7 and min confidence is 0.5, then the association rules generated by Apriori Algorithm is 28, whereas 22 association rules have been generated by Genetic Algorithm. As a result genetic algorithm has generated less number of strong association rules when compared to Apriori algorithm. Thus the Genetic Algorithm has better performance over Apriori Algorithm in generating the association rules.

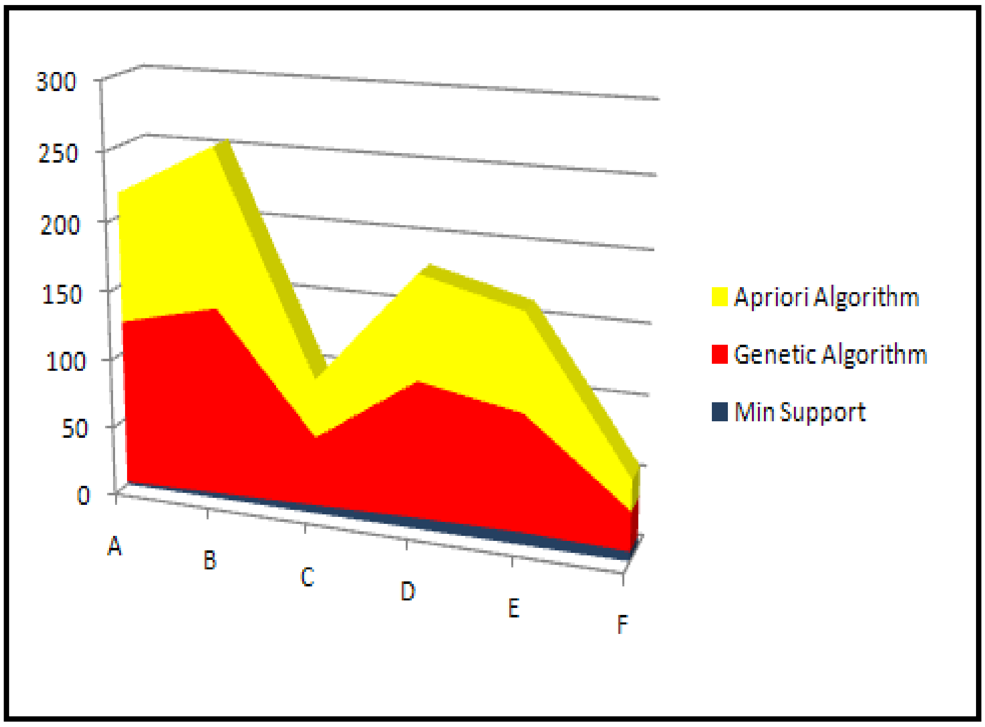

Figure 4.5 Rule Generation Graph of Apriori/GA

Figure 4.5 shows the Apriori/Genetic Algorithm rule generation graph. Using table 4.1, graph has been plotted for Apriori and GA. In this graph, the attributes were taken in the x-axis and number of rules generation was taken in the y-axis. The graph shows the association rules have been optimized using Genetic Algorithm. The performance graph shows that Genetic Algorithm approach is efficient and better when compared to Apriori algorithm.