Association Rule Mining Using Genetic Algorithm

[1]Angarkar Sharwari. S

Abstract

The process involving discovering interesting and unexpected rules from large data sets namely, association rule mining make strong simplifying assumptions about the form of rules. The measure of rule quality is now simplified to properties such as support and confidence. The interestingness of the generated rules thus is limited comprehensibly. The association rule mining problem is modeled as a multi-objective problem getting objects from various data sets, even extended to a distributed environment. These rules are then optimized to get the best rules that can be comprehended as required results. The pre-requirement involves large complexities and optimizations hence become a candidate to be using an evolutionary algorithm rule mining method based on Genetic algorithms.This proposed work is suitable for knowledge extraction from large and possibly noisy databases/data repositories. It uses the genetic algorithm technique for acquiring the best association rules

Keywords: Data mining, Association rule mining, Genetic algorithm

Introduction

Data mining, also known as KDD, or Knowledge Discovery in Databases, refers to the attempt to extract previously unknown and potentially useful relations and other information from databases and to present the acquired knowledge in a form that is easily comprehensible to humans [1]. The different techniques used in data mining are classification, clustering, association rule mining etc.In recent years; there have been numerous attempts to apply evolutionary algorithms (EAs) in data mining to tackle the problem of knowledge extraction and classification or to accomplish tasks in various domains. Among the different areas in data mining, EAs have achieved vast popularity in the application of rule induction, since the sets of IF-THEN rules can be easily represented by choosing an encoding of rules that allocate specific substrings for each rule pre-condition and post-condition [2].

We come across a very voluminous dataset including large number of rules and the problem becomes NP hard even for simple searching for an interesting data item. This high complexity makes the system a candidate to be solved by using Genetic Algorithms, where reducibility will be done through generations.

GAs are known to be one of the best methods for searching and optimization. By applying genetic operators (reproduction, crossover, and mutation) in a population of individuals (sets of unknown parameters properly coded), they achieve the optimum value of the fitness function, which corresponds to the most suitable solution. As a result, they converge to the (near) optimal solution by evolving the best individuals in each generation. The main advantage of the GAs is that they use the parameters values instead of the parameters themselves. In this way they search the whole parameter space.

Association rule mining (ARM) is one of the core data mining techniques. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories [3][4]. The major aim of ARM is to find the set of all subsets of items or attributes that frequently occur in many database records or transactions, and additionally, to extract rules on how a subset of items influences the presence of another subset.

The process of discovering interesting and unexpected rules from large data sets is known as association rule mining. The typical approach is to make strong simplifying assumptions about the form of the rules, and limit the measure of rule quality to simple properties such as support or confidence.

Support and confidence limit the level of interestingness of the generated rules.

In our GA based data mining system the association rules are represented as chromosomes. Here predicates in association rules are genomes in the chromosomes. Initial population is randomly selected. Fitness function is based on confidence and support values of association rules If a rules confidence =0 or if it covers no new data, it gets 0 fitness. Thus during rule generation rule fitness depends primarily on confidence and support values. After evaluating fitness function GA operators (crossover and mutation) are applied. If the new individuals are better than their parents then they form the new population and take part in reproduction .Again this entire procedure is repeated until the best rules are obtained. This iterative process ends with set of best rules [5][6].

Methodology

This data mining system is based on Genetic algorithm. The infrastructure of distributed framework is a remote method invocation system, which allows an object running in one virtual machine to invoke methods on an object running on another. We may use the client server environment. A typical server application creates some remote objects, makes references to them accessible and waits for clients to invoke methods on these remote objects. The client in our scenario is a data mining task, which is in charge of identifying the remote compute engine, dispatching the workload and collecting the final results. The client is a major computational component that involves the rule population. Users can freely choose among workloads. The rule evolving process is launched in the client, which is continued until all the resultant rule sets are sent back from the remote compute engines. During the course of rule evolution, a main rule pool is set up in the client and is maintained by feeding with rules from the rule population. In the rule population, chromosomes representing a single rule that is variable in length are evaluated for data set before Genetic algorithm is applied and the final result of the rule set is evolved in the remote engines. At the end of the evolution, each rule set population outputs its 'best' candidate rule set. In this way the order and number of rules in the rule sets can be optimized and determined simultaneously. The local rule pools can be updated periodically by new rules from the main rule pool. This is not simultaneous since communication between the client and the remote engines are conducted individually.

Before rule acquisition using Genetic algorithm can be performed, the variables and predicates with which rules will be built must be defined and a number of options controlling the genetic algorithm component of the system must be chosen. The user must specify the variables and predicates with an identifying label and an English translation for reporting, the type of variable, its range and precision, comparison operators and formula for calculation, if a derived field or a program for evaluation for predicates. Variable class restrictions and rule positions may be of importance. The narrow search spaces by preventing variables of given types from being compared to others. Rule position fixes number of types of elements in the antecedent and consequents.

Every individual (perspective solution) used in the Genetic algorithm is a chromosome. In the rule population, chromosomes are encoded to represent a single rule that is variable in length having symbolic structures i.e. association rules that may contain n-place predicates ($n > 0$). The initial chromosomes are evaluated for their fitness values based upon the training data set before the evaluation is started.

The mating pool is formed by selecting parents from the rule population using tournament selection. The genetic operators, such as crossover and mutation, are then applied to reproduce offspring for the new main population. Token competition can be applied to maintain a pool of good rules that cover solution space well. To maintain a population with a high diversity, a regeneration operator is used to replace chromosomes that are below average fitness in the main population with randomly generated chromosomes at some user specified probability. After regeneration the main rule pool is set for every rule set population. Besides typed crossover and mutation operators, it uses macro mutation as generalization and specialization operators to efficiently explore the space of rules. The evolution of the rule population continues until all the rule set populations have finished their evolutions.

Fitness Function**:** Rule fitness depends primarily on Confidence and support, and optionally on a parsimony reward based on data coverage. For each proposed rule, Confidence and Support are calculated based on the available Confidence = 0 or if it covers new data it gets Fitness = 0, otherwise the distance of its Confidence and Support from the target values is determined, using a Quasi Euclidian distance.

Fitness = 100 – sqrt [(Confidence- Confidence target) ^2 + (Support – Support target) ^2]

The resulting value is normalized as a percentage, it is adjusted by rewarding by a small constant for each conjunct by which the maximum antecedent size exceeds the rules antecedent size so that

Fitness = Fitness+ (max Ant size – Ant size) * reward.

In classification tasks, fitness value is further adjusted by taking the rule's data coverage into account. The rules are maintained in descending order of fitness. For rules covering same data, only the rule with the highest Confidence and Support get credited for these data. This encourages an extensive exploration of the data set. Data coverage refers to perfect and estimated coverage. These measures promote default hierarchy.

Population size: Between generations the population can grow to more than twice the present size, because the rules are copied along with two macro computations into the next generation. The initial and overall maximum antecedent size specifies the maximum number of conjugations in the antecedent, in the initial population, and in the course of evolution. The initial and overall maximum consequent sizes for the classification are both 1. The crossover rate is set to 0.8. Macro mutations operate within conjuncts and macro mutations achieve generalizations and specialization by deleting and adding conjuncts. The rate of both mutations varies from 0.06 to 0.75 and 0.01 to 0.1 respectively.

Terminal condition: Processing stops after a specific number of generations, or when full data coverage is achieved.

Processing of one generation:

Step 1: Copying one or more of the current best individuals (rules) to the next generation to guarantee that the top fitness level will not drop between generations.

Step 2: Fitness proportional selection, crossover and macro and micro mutations are applied until the new population is complete, rule pairs are repeatedly selected and possibly crossed over. Crossover splits rules only between conjunctions. This permits rules to grow and shrink during this process. All rules are subjected to micro and macro mutations. All rules with positive confidence are macro mutated and population size grows during generations.

Step 3: Intergeneration processing is carried out to ensure validity and non redundancy. Rules are compared and discarded if redundant. After enough valid rules have been selected, modified and inserted into the new population, the evolution for the current generation is complete. Shortest rule set is preferable for discovering association rules

Flow chart for the proposed work is as shown in figure 1:

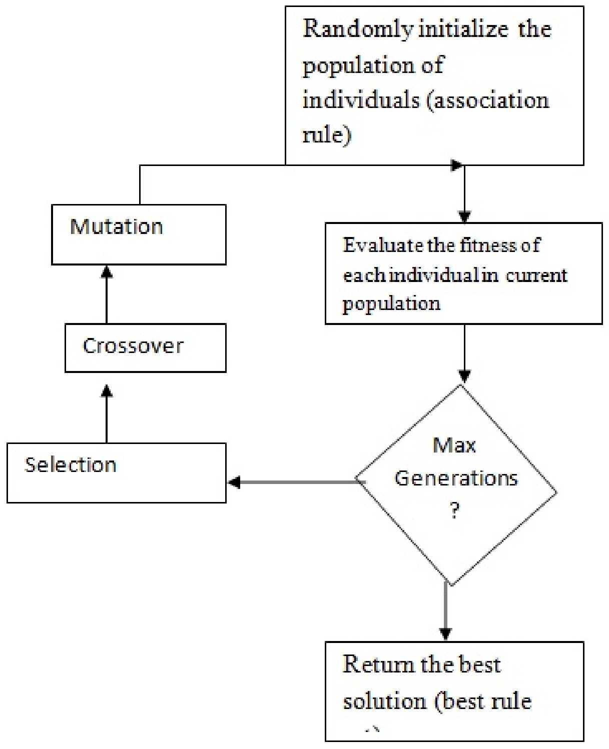
Randomly initialize the population of individuals (association rule)
Evaluate the fitness of each individual in current population
Max Generations ?
Selection
Crossover
Mutation
Return the best solution (best rule

Figure 1: Flow chart for proposed work

Results of Proposed Work

The proposed system can produce the best association rules from the large dataset by applying Genetic algorithm technique. The results can be compared for accuracy and error rate with the Olex_GA - a plug in WEKA for text classification.

Conclusion

Now we are concluding here that our proposed system can perform the rule analysis on the large data repositories using Genetic algorithm technique and will produce the best set of rules. The large data repositories make the problem NP hard. We choose genetic Algorithms to find a solution since it works in generations and caters to reducibility of the problem.

References

• [1]. Holland, J.H., "Adaptation in Natural and Artificial Systems", MIT Press, 1975.

[2]. T. M. Mitchell, Machine Learning: McGraw Hill, 1997.

[3]. 1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A. I.: Fast Discovery of Association Rules. In Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park (1996) 307-328

[4]. Rule Acquisition with a Genetic Algorithm-0-7803-5536-9/99/$10.00 0 1999 IEEE

[5]. Ishibuchi, H., Yamamoto, T.: Fuzzy Rule Selection by Multi-Objective Genetic Local Search Algorithms and Rule Evaluation Measures in Data Mining. Fuzzy Sets and Systems 141 (2004) 59-88

[6]. A Distributed Evolutionary Classifier for Knowledge Discovery in Data Mining-K. C. Tan, Member, IEEE, Q. Yu, and T. H. Lee, Member, IEEE .IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 35, NO. 2, MAY 2005 131

Author's details

[1]M.E. Scholar-C.S.E, Pune, Maharashtra, India, Email: sharwarisjoshi@gmail.com