An Innovative GA-Based Decision Tree Classifier in Large Scale Data Mining

Zhiwei Fu

Robert H. Smith School of Business, University of Maryland, College Park, 20742, USA
zfu@rhsmith.umd.edu

Abstract. A variety of techniques have been developed to scale decision tree classifiers in data mining to extract valuable knowledge. However, these approaches either cause a loss of accuracy or cannot effectively uncover the data structure. We explore a more promising GA-based decision tree classifier, OO-GASC4.5, to integrate the strengths of decision tree algorithms with statistical sampling and genetic algorithm. The proposed program could not only enhance the classification accuracy but assumes the potential advantage of extracting valuable rules as well. The computational results are provided along with analysis and conclusions.

1 Introduction

Data mining has recently become an attractive discipline within the last few years [4][7]. Its goal is to extract pieces of previously unknown, but valuable knowledge or patterns from large data sets. Data mining over large data sets is important due to its commercial potential. Numerous algorithms have been developed with regard to handling large data sets, such as distributed algorithms, restricted search, parallel algorithms and data reduction algorithms. However, the computational costs, the available storage and the retrieval of large data sets are still serious concerns for large-scale data mining. While certain data mining algorithms show consistent performances for some data sets, it is not necessarily true across all problem domains. For example, the performance of different extracted “ideal” scheme for induction based classification/decision problems varies greatly with the characteristics of the data sets to which the algorithms applies. On the other hand, techniques such as discretization can cause a loss of accuracy when scaling up to large data sets [1][8].

In this paper we explore a more promising algorithm using genetic algorithm (GA) and SampleC4.5 for classification problems. The paper begins with introducing the problem and reviewing the decision tree algorithm and GA, followed by the design of Object Oriented Genetic Algorithm with SampleC4.5 (OOGASC4.5). We then show the computational results. After comparison and analysis, we end up with conclusions in the final section.

2 The OOGASC4.5 Program

2.1 C4.5 and SampleC4.5

Decision tree algorithms have long been recognized as a powerful tool in data mining to represent schemes in the studied data sets according to values of variables. Among them, C4.5 [9] has been widely implemented and tested. It uses the top-down induction approach to build the decision tree, which is fitted to training samples by recursively partitioning the data into increasingly homogeneous subsets, based on the values of a variable one at a time. It starts at the root of the tree and moves through it until a leaf is encountered, or no more improvement could be made.

However, C4.5 pursues local greedy search which can quickly converge but at the expense of higher possibility of getting trapped into the local optima. To efficiently uncover the data set, we develop SampleC4.5 by instilling statistical sampling methods into C4.5. The SampleC4.5 algorithm starts with the full original set of variables and a training set of certain starting percentage ($p=p_0$) extracted from raw data. The remaining data $(1-p_0) \times N$ form the test set, where N is the size of the original data set. In the case of large data sets, we would form a separate test set, and an “untouched” validation set beforehand rather than the dynamic $(1-p_0) \times N$ test sets. SampleC4.5 is implemented on the training set for a number of iterations ($n=n_0$). The classification accuracy on test sets over all iterations n is averaged, and the standard error is calculated, after which the algorithm increments p by some step percentage s_0 . The process repeats on a new training set of $(p_0+s_0) \times N$, and a new test set of $(1-(p_0+s_0)) \times N$ or the pre-specified test/validation set for large data sets. Two statistical sampling approaches, simple random sampler and stratified random sampler, are provided. The pseudo code for the algorithm is shown as follows,

```

begin
    initialize  $n=1$ ;
    initialize  $p=p_0$ ;
    while ( $0 < p < 100$ ) loop
        while ( $0 < n \leq n_0$ ) loop
            Do (simple|stratified random sampling)
            run C4.5 on the training set of  $\{p \times N\}_n$ 
            if (large data sets)
                validate on the pre-specified test set
            else
                validate on the test set of  $\{(1-p) \times N\}_n$ 
                increment  $n$ 
            end loop
            average the performance over  $n_0$  iterations;
            calculate the standard error over  $n_0$  iterations;
             $p = p + s_0$ 
        end loop
    end.

```

2.2 Genetic Algorithm

Genetic algorithm (GA) [6] has been using in a wide variety of applications, from combinatorial optimization to data mining. GA is domain independent global search technique in that it exploits accumulating information about an initially unknown search space in order to bias subsequent search into more promising subspace. GA proceeds iterative search from one population of candidate solutions to another, each consisting of m candidate solutions. The global search taken by GA is randomized but structured since during each iteration step, called a generation, the current population is evaluated by the fitness function, and, based on the evaluation and evolving strategies, a new population of candidate solutions is formed. GA then searches through the parameter space in many directions simultaneously and thereby improves the probability of convergence to the global optima [3][5].

2.3 OOGASC4.5

Taking advantage of GA and statistical sampling, we develop the OOGASC4.5 program to build better decision tree(s) for classification problems. In OOGASC4.5, there are three principal issues involved in GA.

1. *The initial population.* An appropriate representation of the problem space is a key issue for GA to achieve efficiency. Usually the initial population is chosen randomly. We create genetic tree chromosomes for GA rather than map the high-dimensional decision tree onto a linear genetic chromosome.
2. *Genetic operations.* There are two types of crossover operations in our program, subtree-to-subtree and subtree-to-leaf crossover, which randomly exchange between subtree and subtree or subtree and leaf from two parent trees. Mutation operation is implemented as an asexual crossover by randomly exchanging any subtree or leaf with other subtree or leaf with only one tree involved.
3. *Evaluation.* Appropriately defining the evaluation/objective function is the main issue of successfully applying GA. In the program, we employ the accuracy metric to evaluate the performance of the generated decision trees. This is the percentage of correctly classified observations in the test data sets.

The OOGASC4.5 program is shown in Fig. 1. The object oriented design was conducted on Rose98 from Rational Software Corp. The program was implemented in Microsoft Visual C++ 5.0 on Windows95 with 128.0MB RAM and 400 MHz.

In the program, we first build the diversified tree models from different (not necessarily mutually exclusive) subsets of the original data set by SampleC4.5, then take the output trees as inputs to GA. In the process of evolving, feasibility of the generated trees is checked since the trees generated after genetic operations are probably not feasible and only those feasible trees are of interest. After feasibility check, the fitness values of the generated trees are calculated over the test set.

Fig. 1. The OOGASC4.5 program

3 Experimental Results

3.1 Parameter Settings

In the experiments, we set the probability of crossover at 1.0, mutation rate 0.01, and used 50 generations as the stopping criterion. The sampling percentages vary from 10% to 60%, and the population size from 10 to 100.

3.2 SocioOlympic Data

We implement OOGASC4.5 on SocioOlympic data collected from the 1996 Atlanta Summer Olympic Games, in which 11,000 athletes from 197 nations participated in 271 events from 26 sports. While more countries than ever appeared in the final medal count (78 countries received at least one medal), some countries regularly win more than others do. Many attempts have been made in the literature to explain these differences. Population is cited most frequently as a factor affecting Olympic success. However, we only need to compare recent performances of nations such as Cuba and India to see that other factors are involved (in 1996, Cuba won 25 medals while India won 1). The total score for each country is calculated by assigning points to the top eight placing countries for each of the 271 Olympic events. Socioeconomic information (e.g., population and GNP) is used to classify the countries into categories based on the total scores attained and predicted [2].

Table 1. Computational results for SampleC4.5 and OOGASC4.5

Parameters			Population size						
			10	50	60	70	80	90	100
Best of the Initial Population (SampleC4.5 Trees)	Sampling Percentage	5%	0.6462	0.6462	0.6462	0.6462	0.6462	0.6462	0.6462
		10%	0.8462	0.8462	0.8462	0.8462	0.8462	0.8462	0.8462
		20%	0.8359	0.8462	0.8462	0.8462	0.8462	0.8462	0.8462
		30%	0.8462	0.8769	0.8769	0.8769	0.8769	0.8769	0.8769
		40%	0.8769	0.8872	0.8872	0.8974	0.8974	0.8974	0.8974
		50%	0.8410	0.9128	0.9128	0.9128	0.9128	0.9128	0.9128
		60%	0.8974	0.9026	0.9026	0.9026	0.9128	0.9128	0.9128
Best Output GASampleC4.5 Tree	Sampling Percentage	5%	0.7385	0.7385	0.7385	0.7385	0.7385	0.7385	0.7385
		10%	0.8667	0.8769	0.8872	0.8872	0.8974	0.8769	0.8974
		20%	0.8462	0.8923	0.8976	0.8769	0.8769	0.8923	0.8976
		30%	0.8667	0.8821	0.9128	0.9179	0.8872	0.9128	0.9179
		40%	0.8974	0.9128	0.9128	0.9128	0.9128	0.9128	0.9128
		50%	0.8821	0.9128	0.9231	0.9282	0.9128	0.9179	0.9231
		60%	0.9077	0.9077	0.9179	0.9179	0.9231	0.9179	0.9333
Convergence (No. of Generations)	Sampling Percentage	5%	1	1	1	1	1	1	1
		10%	9	8	7	15	6	11	12
		20%	4	9	16	9	8	14	13
		30%	9	11	17	24	13	26	18
		40%	14	16	8	7	20	3	30
		50%	40	1	26	33	1	3	6
		60%	19	7	12	11	22	4	25
Computing Time for GAOOSC4.5 (seconds)	Sampling Percentage	5%	14.8	70.6	86.8	98.4	112.4	130.3	144.7
		10%	15.0	72.9	87.8	101.9	116.8	131.1	146.3
		20%	15.1	73.2	88.7	102.3	117.4	131.5	146.9
		30%	15.1	73.6	89.8	102.8	118.3	132.8	147.9
		40%	15.2	74.6	89.5	103.7	119.0	133.4	148.5
		50%	15.3	74.7	90.1	104.9	119.8	135.2	149.6
		60%	15.5	75.3	90.8	105.8	121.5	138	150.5

3.3 Results

Table 1 shows experiment results for SampleC4.5 and OOGASC4.5 from different respects. For any combinations of the population size and the sampling percentage, GASampleC4.5 output trees perform better than SampleC4.5 trees. The improvement mainly varies from 2% to 9%, which is significantly indicative of the robustness of OOGASC4.5. OOGASC4.5 can effectively achieve at least the same level performance as SampleC4.5 at a much lower sampling percentage, which will be beneficial for large-scale data mining. The sampling percentages should be set up problem-specific, considering the manageable size of the sampled subsets. In the case of large data sets, we can reasonably set the sampling percentage as low as 0.1%, or even 0.01%, to build up the initial tree models. The population size seems more dominant in computing time than the sampling percentage since as the number of candidate decision trees increases, more genetic operations and evaluations will be done by GA. On the other hand, that the sampling percentage does not significantly affect total computing time is also due to the fact that the extracted number of observations by sampling from the original data is not significantly large.

OOGASC4.5 has some limitations. First, the embedded GA search will entail many evaluations of the fitness functions and, consequently, much computing time. The second limitation is that convergence of GA does not necessarily occur at the global optima, but close enough. OOGASC4.5 converges relatively fast as illustrated in Table 1, but does not assume any specific trend.

4 Conclusion

The OOGASC4.5 program demonstrates that better decision trees could be built using a well-designed GA. The decision trees built by OOGASC4.5 can achieve better performance than those by traditional approaches. The generated trees usually differ from those by SampleC4.5, and the difference is often extended to the variable at the root node. This could be due to the difference in the nature of searching method. Decision tree algorithms like C4.5 pursue local greedy search through the data space, by selecting the most significant variable to partition one level down at a time. However, the variables that most significantly partition the data sets at each step are not necessarily the combination that leads to the best decision trees.

While OOGASC4.5 is designed for large data sets, we use the SocioOlympic data to test the efficacy of various sampling levels and parameter settings, and to validate the accuracy of our approach. Using the learning with the small data set we intend to run OOGASC4.5 on very large data sets. Part of the next work would be extended to extracting, aggregating and enhancing the valuable rules discovered by the program. Exploration of more effective genetic crossover and mutation operations is also part of the future work.

References

1. Chattratchat, J. et al. Large Scale Data Mining: Challenges and Responses. *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, AAAI Press, Menlo Park, CA, 1997.
2. Condon, E. *Predicting the Success of Nations in the Summer Olympics Using Neural Networks*, Master's Thesis. University of Maryland, College Park, MD, 1997.
3. DeJong, K. Adaptive system design: a genetic approach. *IEEE Transaction on Systems, Man, and Cybernetics*, Vol. SMC-10, No. 9, pp.566-574, Sept. 1980.
4. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P. *Advances in Knowledge Discovery and Data Mining*, MIT Press, 1996.
5. Goldberg, D. *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA 1988.
6. Holland, J. H. *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.
7. Piatetsky-Shapiro, G., Frawley, W. *Knowledge Discovery in Databases*, MIT Press, 1991.
8. Provost, F. Scaling up Inductive Algorithm: An Overview. *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, p239-242. AAAI Press, Menlo Park, CA, 1997.
9. Quinlan, J. R. *C4.5: Programming for Machine Learning*, Morgan Kaufman Publishers, 1993.