World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:1, 2009

Data Mining Using Learning Automata

M. R. Aghaebrahimi, S. H. Zahiri, and M. Amiri

***Abstract*—**In this paper a data miner based on the learning automata is proposed and is called LA-miner. The LA-miner extracts classification rules from data sets automatically. The proposed algorithm is established based on the function optimization using learning automata. The experimental results on three benchmarks indicate that the performance of the proposed LA-miner is comparable with (sometimes better than) the Ant-miner (a data miner algorithm based on the Ant Colony optimization algorithm) and CNZ (a well-known data mining algorithm for classification).

***Keywords*—**Data mining, Learning automata, Classification rules, Knowledge discovery.

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/6146/pdf

I. Introduction

DATA mining is referred to knowledge extraction from data and its basic core is at the intersection of machine learning, statistics, and databases.

There are several data mining tasks, including classification, regression, clustering, dependence modeling, etc. Each of these tasks may be regarded as a kind of problem that can be solved by a data mining algorithm.

In this paper, we propose a data mining algorithm for the classification task of data mining. In this task, the goal is to assign of each case (object, record, or instance) to one distinct class. The proposed algorithm is established based on a function optimization algorithm using learning automata (LA) and is called LA-miner.

Learning automata is referred to an automaton acting in an unknown random environment which improves its performance to obtain an optimal action. An action is applied to a random environment and the random environment evaluates the applied action and gives a fitness value to the selected action of automata. The response of the environment is used by automaton in order to select its next action. This procedure is continued until the optimal action is reached.

LA has been used in several tasks of engineering problems such as graph partitioning [1], adaptive control [2], and etc.

According to the best of our knowledge, the use of LA to discover classification rules in the context of data mining is a research area still remaining unexplored.

Actually, the only LA algorithm for pattern classification (in general usage) that we are aware of, is an algorithm for decision hyper-plane estimation in the feature space [3], which is very different from the rule discovery task addressed in this paper.

The discovered knowledge by LA-miner is expressed in the form of IF-THEN rules.

The rules antecedent contain a set of conditions, usually connected by logical conjunction operator (AND). Each condition is usually referred to a term. The consequent rule (THEN part) specifies the class predicted for cases which their predictor attributes satisfy all the terms specified in the rule antecedent. The rest of this paper is organized as follows:

Section II, consists of a brief overview of basic concepts of learning automata and its application to function optimization task. Section III introduces the proposed LA-miner for discovering classification rules, which is proposed in this work. In Section IV, the computational and experimental results are presented to evaluate the performance of the proposed LA-miner. In particular, in this Section a comparison between LA-miner and two novel and well-known algorithms (Ant-miner [4] and CNZ [5]) across six data sets is given. Finally, Section V concludes the paper.

II. Basic Concepts

A. Learning Automata

Learning automata (LA) are adaptive decision making units that can be learned to choose the optimal function from a set of actions by interaction with an environment (search space). In other words, a learning automaton is a stochastic automaton that is connected as a feedback connection to a random environment. Each learning automaton is characterized by a set of internal states, input actions, state probability distributions, and reinforcement scheme, which is connected in a feedback loop to the environment, as shown in Fig. 1.

One of the main advantages of the learning automaton is that it needs no important knowledge of the environment in which it operates or any analytical knowledge of the function to be optimized.

A finite learning automata and the environment is generally defined by $< A, Q, R, L >$ and $< A, R, D >$ respectively, where $A = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ is defined as all actions of the automaton where $\alpha(k)$ is the automaton at the instant k and $\alpha(k) \in A$ for $k = 0,1,2,...$ and r is the total number of actions.

M. R. Aghaebrahimi is with the University of Birjand, Birjand, Iran (98-561-2227044; fax: 98-561-2227795; e-mail: Aghaebrahimi@Birjand.ac.ir).

S. H. Zahiri is with the University of Birjand, Birjand, Iran (e-mail: Hzahiri@ Birjand.ac.ir).

M. Amiri is with the University of Birjand, Birjand, Iran (e-mail: M.Amiri@ Birjand.ac.ir).

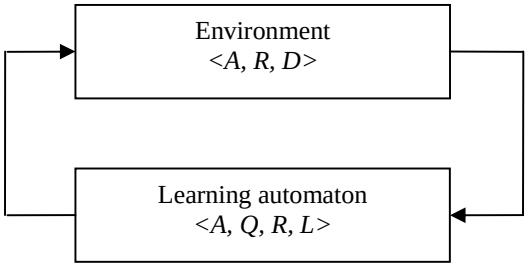


Fig. 1 Automaton operating in the environment

In fact, A is the set of outputs of the automaton and it is also the set of inputs to the environment. R is the domain of the environment responses. Let $\beta(k)$ denote the response received by the automaton at instant k where $\beta(k) \in R \ \forall k$. $\beta(k)$ is the output of the environment and it is also the input to the automaton. $D = \{d_1, d_2, \ldots, d_r\}$ is the set of reward probabilities, where $d_i(k) = E[\beta(k) \mid \alpha(k) = \alpha_i]$. The reward probabilities are unknown for the automaton. Q is the state of the automaton defined by $Q(k) = [P(k), \hat{D}(k)]$, where:

$P(k) = [p_1(k), p_2(k), \ldots, p_r(k)]$ is the action probability vector $(0 \le p_i(k) \le 1 \quad and \sum_{i=1}^{r} p_i(k) = 1, \forall k)$ and $\hat{D}(k) = [\hat{d}_1(k), \hat{d}_2(k), \ldots, \hat{d}_r(k)]$ is the vector of estimates of the reward probabilities at instant k. T is the learning algorithm or the reinforcement scheme which is used by the automaton in order to update its state. In fact: $Q(k+1) = L(Q(k), \alpha(k), \beta(k))$.

At each instant k, the automaton selects an action $\alpha(k)$ from the set of all actions A. This selection depends on the current action probability vector $P(k)$. The selected action $\alpha(k)$ becomes input to the environment and the environment gives a random response $\beta(k)$ as the input of the automaton whose expected value is $d_i(k)$ if $\alpha(k) = \alpha_i$. Next, the automaton calculates $Q(k+1)$ using the reinforcement scheme L.

This procedure is continued until the optimal action to the environment is found. We denote the optimal action as α_m with expected value of $d_m = Max\{d_i\}$ for all $i = 1,2,...,r$. It is desired that the action probability corresponding to α_m approaches unity as the time k approaches infinity.

B. Function Optimization Using LA

In [3] a function optimization technique based on the LA is used to estimate the decision hyper-plane in the feature space. We utilize this technique to establish our proposed LA-miner. Thus, in this subsection, the explanations about this function optimization algorithm are presented.

In this algorithm, at first the solution space is uniformly divided into r hyper-cubes, each corresponding to one action of the learning automaton. Then, using continuous Pursuit algorithm the action probabilities and estimates of reward probabilities are updated at each period by calculating the function value of a randomly selected sample corresponding to the current action. If the estimate of a reward probability is smaller than a predefined threshold, the corresponding hypercube is then evaluated according to the samples whose function values have been calculated. If both the mean value and the variance of these function values are small enough, this hypercube is considered as stable and useless. Then, this hypercube is removed and the optimization continues with the remaining r-1 hyper-cubes. Otherwise, this hypercube is considered as unstable and the rising and falling (peaks and valleys) of the function are estimated in this hypercube from the samples inside it. Next, this hypercube is divided into a number of sub-hyper-cubes each only containing ascending or descending samples and the original hypercube is replaced by the best rewarded sub-hypercube. The other sub-hyper-cubes are considered as useless and then removed. In this way, the number of actions is unchanged. This procedure is repeated until a predefined precision condition is satisfied. Then, original hyper-cubes are either removed or converge to several values in which are included a quasi global optimum, i.e. a solution whose function value is rather close to that of the global optimum. Like other stochastic optimization algorithms, this method aims at finding a compromise between exploration and exploitation, i.e. converging to the nearest local optima and exploring the function behavior in order to discover global optimal region.

The schedule of the function optimization based on the learning automata is shown in Fig. 2.

III. The Proposed LA-Miner

As it was mentioned in Section I, our proposed LA-miner is designed based on a function optimization algorithm which uses a learning automata technique.

To explain how LA-miner extracts the IF-THEN rules, consider the structure of a rule:

$$IF\ (att_1\ is\ quant_1)\ AND\ (att_2\ is\ quant_2)\ AND\ \ldots (att_n\ is\ quant_n)\ THEN\ x\ belongs\ to\ Class\ j. \quad (1)$$

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/6146/pdf

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/6146/pdf

Where att_i ($i = 1,2,...,n$) is ith attribute of the training point and $x = (att_1, att_2, ..., att_n)$ is the training point which has been expressed in the feature space. The values of the $quant_i$ are the logical expressions of numerical quantities of ith attribute. For this reason, we code the quantities of each attribute as follows.

Suppose Max_i and Min_i are the maximum and minimum values of att_i respectively, we code the quantity of att_i to logical expression using these relations:

$$att_i = \begin{cases} Low \quad if \quad Min_i < quant_i < \dfrac{Max_i}{3} \\ Medium \quad if \quad \dfrac{Max_i}{3} < quant_i < \dfrac{2Max_i}{3} \\ High \quad if \quad \dfrac{2Max_i}{3} < quant_i < Max_i \end{cases} \tag{2}$$

The basic duty of LA-miner is to extract K distinct rules for classifying M reference classes in the n-dimensional feature space. In fact, for each rule there are $(n+1)$ unknown parameters (n logical expressions of $quant_i$ and j is the index of a reference class).

Obviously, for K rules, LA-miner should extract $K \times (n+1)$ unknown parameters from the solution space. Thus, LA-miner searches for unknown sets of antecedents and consequences of the form:

$$\begin{aligned} \text{Rule_Set} = \{ & \text{quant}_{11}, \text{quant}_{12}, ..., \text{quant}_{1n}, j_1, \\ & \text{quant}_{21}, \text{quant}_{22}, ..., \text{quant}_{2n}, j_2, ..., \text{quant}_{k1}, \text{quant}_{k2} \\ & , ..., \text{quant}_{kn}, j_k \} \end{aligned} \tag{3}$$

To evaluate each Rule-Set obtained by the LA-miner, we define a fitness function as below:

$$fit(Rule_Set) = T - miss; \tag{4}$$

Where, T is the total number of training points, miss is the number of miss-classified points by the Rule_Set, and $fit(.)$ is the fitness value of the Rule_Set.

According to the above descriptions, the proposed LA-miner uses the function optimization algorithm based on the learning automata (Section 2) and maximizes the function defined by (4). The output of this algorithm is the Rule_Set which has the minimum miss-classified points.

Step 1: Initialization of internal parameters

r: Number of hypercubes which divide the feature space
δ : Threshold of action probabilities
s: Normalized factor of convergence
ε : Error band

Step 2: Divide the feature space into *r* hypercubes

Each hypercube is supposed as an action denoted by $\alpha_i, i \in \{1,2,\ldots,r\}$.

Corresponding to each action there are some parameters introduced as below:

$\eta_i(n)$: Total reward obtained by the action α_i until nth sampling instant.

$z_i(n)$: Number of times the action α_i is chosen until nth sampling instant.

$$\xi_i(n) = \frac{\eta_i(n)}{z_i(n)},$$

$$\xi_m(n) = Max_i\{\xi_i(n)\}$$

$$\xi_l(n) = Min_i\{\xi_i(n)\}$$

$p(n)$: Action probability distribution of α_i at nth sampling instant with initial value of $\frac{1}{r}$.

Step 3: Search loop

Repeat

- Pick up an action $\alpha(n) = \alpha_i(n)$ according to $p(n)$.
- Randomly select $X = (x_1, x_2, \ldots, x_n)$ from the hypercube corresponding to α_i .
- Calculate $f(X)$.
- Update $\xi_i(n)$ as follows:

If $\alpha(n) = \alpha_i$, Then

$$\eta_i(n+1) = \eta_i(n) + \frac{M - f(X)}{M - L},$$

Where M and L are the estimated upper and lower of $f(x)$ respectively, permitting to normalize the estimates of reward probabilities to the interval [0, 1].

$$z_i(n+1) = z_i(n) + 1,$$

$$\xi_i(n+1) = \frac{\eta_i(n+1)}{z_i(n+1)},$$

For all $j \neq i$

$$\eta_j(n+1) = \eta_j(n),$$

$$z_j(n+1) = z_j(n),$$

$$\xi_j(n+1) = \xi_j(n)$$

- Update $p(n)$ as follows:

$$p(n+1) = (1 - s * \xi_m(n)) * p(n) + s * \xi_m(n) * e_m,$$

:(e_m is a r-dimensional vector with mth component unity and all others zero)

- If $p_l(n) = Min_i\{p_i(n)\} < \delta$, Then
  Go to the next step,
- Else
  n=n+1,

End Repeat;

Step 4: Enhancing the search process in *l*th hypercube.

Fig.2 The schedule of the function optimization based on the learning automata

IV. Computational and Comparative Results

In this section, performance evaluation of the LA-miner is investigated. Also the comparative results with Ant-miner and CNZ is presented. Ant-miner is an algorithm for classification rule discovery which uses ant colony optimization [4]. CNZ is a famous classification rule discovery algorithm. CNZ searches for a rule list in an incremental fashion [5].

Three pattern classification problems with different augmented feature vector dimensions (5,14,129), are used for performance evaluation and comparison of the results. A description of the data sets is given in the next subsection.

Open Science Index, Computer and Information Engineering Vol:3, No:1, 2009 publications.waset.org/6146/pdf

A. Data Sets

data[1]: The Iris data contains 50 measurements of four features of each three species: Iris setosa, Iris versicolor, and Iris virginica. Features are sepal length, sepal width, petal length and petal width.

Wine data[2]: The Wine data contains the chemical analysis of wines grown in the same region in Italy but derived from different cultivars. The 13 continuous attributes are available for classification. The number of classes is three and the numbers of instances in each class are: 59, 71 and 48, respectively.

Cancer data: This breast cancer database, obtained from the University of Wisconsin Hospital, Madison, has 683 breast mass samples belonging to two classes Benign and Malignant, in a nine dimensional feature space[3].

B. Experimental Results

The proposed LA-miner, Ant-miner, and CNZ, are tested on the data sets described in previous section.

Fifty percent of each benchmark is considered as the training points and the rest forms the testing data. The experiment presented ten times for each benchmark, and the average of results is reported.

Table I presents the obtained recognition score by the Rule_Sets which have been extracted by three data miner for training points. The numbers of the rules which have been used by each algorithm are presented in this table, too.

It can be seen from Table I that, the obtained average score of recognition by LA-miner is better than CNZ for all benchmarks in training phase. Also, the proposed algorithm outperforms the Ant-miner for Iris and Cancer data. Only, the performance of the Ant-miner is better than the LA-miner in the case of Wine data set by a negligible value of 0.3%. The average number of obtained Rule- Sets for LA-miner is less than the other two data miners, which shows more effectiveness of LA-miner in comparison with Ant-miner and CNZ.

TABLE I
THE AVERAGE SCORES OF RECOGNITION (%) AND, THE NUMBER OF RULES FOR ANT-MINER, CNZ, AND LA-MINER FOR THREE BENCHMARKS IN TRAINING PHASE

	Ant-miner		*CNZ*		*LA-miner*	
	Rec. Score (%)	No. of Rules	Rec. Score (%)	No. of Rules	Rec. Score (%)	No. of Rules
Iris	94.5	4.2	92.3	6.3	95.3	3.75
Wine	89.2	6.7	85.3	8.2	88.9	5.8
Cancer	95.4	3.6	93.7	4.7	96.2	3.5

Table II shows the score of recognition (%) obtained by three algorithms for testing points.

This Table indicates that the ability of generalization of obtained Rule_Sets of LA-miner outperforms Ant-miner and CNZ for Iris and Wine data sets, because the score of recognition obtained by LA-miner is better than the other two data miners for these data sets. Only for Cancer data set, the performance of Ant-miner is better than LA-miner by only 0.7%.

TABLE II
THE AVERAGE SCORES OF RECOGNITION (%) FOR ANT-MINER, CNZ, AND LA-MINER FOR THREE BENCHMARKS IN TESTING PHASE

	Ant-miner	CNZ	LA-miner
Iris	89.8	87.6	91.7
Wine	83.4	81.0	85.5
Cancer	92.5	91.0	91.8

V. CONCLUSION

A new data miner based on the learning automata for rule discovery in data classification task is proposed. The proposed algorithm, named LA-miner, searches the solution space (rule space in our case), to find the optimum Rule_Sets in order to minimize the number of miss-classified data points.

The performance of the LA-miner is tested on three well-known benchmarks and compared with Ant-miner and CNZ. The comparative results show that the recognition score of LA-miner is comparable (sometimes better than) Ant-miner and CNZ for both training and testing phases. Furthermore, the average numbers of rules obtained by LA-miner is less than the other two data miners.

REFERENCES

[1] B. J. Oommen, E.V. de St. Criox, "Graph partitioning using learning automata," IEEE Trans. Comput., vol. 45, pp. 195-208, 1996.
[2] H. Beigy, M.R. Meybodi, "Backpropagation algorithm adaptation parameters using learning automata," Int. J. Neural Syst., vol. 11, pp.219-228, 2001.
[3] S.H. Zahiri, "Learning automata based classifier," Pattern Recognition Letters, vol. 9, pp.40-48, 2008.
[4] R.S. Parepinelli, H.S. Lopes, A. Freitas, "An ant colony algorithm for classification rules discovery," IEEE Trans. Evol. Comp., vol. 6, No.4, pp.321-332, 2002.
[5] P. Clark, T.Niblet, "The CNZ induction algorithm," Mach. Learn., vol.3, no.4, pp.261-283, 1989.

M. R. Aghaebrahimi was born in Iran. He is an Assistant Professor with the Department of Power Engineering, University of Birjand, Iran. He received his B.Sc. in electrical engineering from Mashhad Ferdowsi University, Iran, in 1989. He also received his M.Sc. and Ph.D. in electrical engineering from University of Manitoba, Canada, in 1993 and 1997, respectively. His areas of interest include HVDC Transmission Systems and Electric Machinery. He is currently involved in research in the area of Renewable Energy Resources, especially Wind Energy.

S. H. Zahiri was born in Iran. He is an Assistant Professor with the Department of Power Engineering, University of Birjand, Iran. He received the B.Sc., M.Sc. and Ph. D. degrees in electronics engineering from the Sharif University of Technology, Tehran, Iran, Tarbiat Modarres University, Tehran, and Ferdowsi University of Mashhad, Mashhad, Iran, in 1993, 1995and 2005, respectively. His research interests include pattern recognition, evolutionary algorithms, and swarm intelligence algorithms.

M. Amiri was born in Iran. He received his B.Eng. from Power and Water University of Technology (PWUT), Tehran, Iran, in 2003. He is currently a student of M.Sc. at department of power engineering, University of Birjand, Iran. His areas of interest include: voltage stability, distributed generation (DG), Reactive power control and Artificial Intelligence.

[1] This data set is available at University of California, Irvine, via anonymous ftp ftp.ics.uci.edu/pub/machine-learning-databases.
[2] Same site.
[3] This data set is available at: http;//www.ics.uci.edu/~mlearn/MLRepository.html.