Intelligent Information Systems VIII
Proceedings of the Workshop held in
Ustroń, Poland, June 14-18 , 1999

An evolutionary algorithm integrating discretization of continuous-valued attributes with learning decision rules

Wojciech Kwedlo and Marek Krętowski

Institute of Computer Science
Technical University of Białystok
Wiejska 45a, 15-351 Białystok, Poland
e-mail: {wkwedlo, mkret}@ii.pb.bialystok.pl

Abstract. A new method of learning decision rules from databases, which uses an evolutionary algorithm, is proposed. The main difference between our approach and the others described in the literature is the way of processing of continuous-valued attributes. Most decision rule learners process separately these attributes when searching for threshold values, which may decrease the performance. In contrast to them, our method globally searches for threshold values for all continuous-valued attributes in the rule at the same time. The initial results of the experiments on some real-life datasets are presented.

Keywords: machine learning, evolutionary algorithms, decision rules

1 Introduction

Discovery of decision rules from databases is one of the most important problems in machine learning and data mining. The advantages of the rule-based approach include natural representation and ease of integration of discovered rules with background knowledge.

Whenever a dataset contains some numerical (continuous-valued) attributes a decision rule learner must search for threshold values to create conditions (selectors) concerning these attributes. The search for the thresholds is performed either before the induction of rules, usually by *discretization* of each continuous-valued attribute independently from the others, or during the generation of inductive hypothesis. However, even the systems, which use the latter method (e.g. C4.5 [8], CN2 [1]) are very often unable to search for threshold values for more than one attribute at the same time.

The separate search for thresholds for continuous-valued attributes, although computationally effective, runs the risk of missing information necessary for correct classification. For instance, it is possible, that two continuous-valued attributes A_1, A_2 discriminate together the target class very well, while both of them considered independently from each other have very low discriminating power. In such case the existing algorithms are very likely to fail to find the proper attribute thresholds.

In the paper we propose a new method, which is able to overcome the aforementioned problem by integrating the two search steps, namely the simultaneous search for threshold values for *all* continuous-valued attributes and the discovery of decision rules. The search is performed by an *evolutionary algorithm* [6] (EA). Several systems, which employ EAs for learning decision rules (e.g. GABIL [2], GIL [4], EDRL [5]) were proposed. According to our knowledge all of them either use only nominal attributes or discretize continuous-valued ones prior to induction of rules.

2 Description of the method

We assume that a learning set $E = \{e_1, e_2, \ldots, e_M\}$ consists of M examples. Each example $e \in E$ is described by N attributes (features) $A_1(e), A_2(e), \ldots, A_N(e)$ and labeled by a class $c(e) \in C$. The domain of a nominal (discrete-valued) attribute A_i is a finite set $V(A_i)$, while the domain of a continuous-valued attribute A_j is an interval $V(A_j) = [a_j, b_j]$. For each class $c_k \in C$ by $E^+(c_k) = \{e \in E : c(e) = c_k\}$ we denote the set of *positive examples* and by $E^-(c_k) = E - E^+(c_k)$ the set of *negative examples*. A *decision rule* R takes the form $t_1 \wedge t_2 \wedge \ldots \wedge t_r \rightarrow c_k$, where $c_k \in C$ and the left-hand side (LHS) is a conjunction of $r (r \leq N)$ conditions $t_1, t_2, \ldots, t_r$; each of them concerns one attribute.

For each class the set of decision rules is generated in a sequential manner. At the beginning a decision rule is found by a global optimization algorithm, namely the EA. The *fitness function*, which express "the goodness" of the rule prefers rules consisting of few conditions, which cover many positive examples and very few negative ones. The positive examples covered by the rule are removed and the search process is iterated on the remaining examples. The sequential covering is stopped when either all the positive examples are covered or a satisfactory decision rule cannot be found.

2.1 Representation

The left-hand side of a decision rule is encoded by a fixed-length chromosome, which is composed of *substrings*. Each substring (Fig. 1) represents a condition related to one attribute. The LHS is the conjunction of these conditions. In case of a continuous-valued attribute A_i the substring encodes the lower l_i and the upper u_i threshold of the condition $l_i < A_i \leq u_i$. It is possible that $l_i = -\infty$ or $u_i = +\infty$. Setting both $l_i = -\infty$ and $u_i = +\infty$ is equivalent to not including the condition into the LHS. Both l_i and u_i are selected from the finite set of all *boundary thresholds*. A boundary threshold for the attribute A_i is defined (Fig. 2) as a midpoint between such a successive pair of

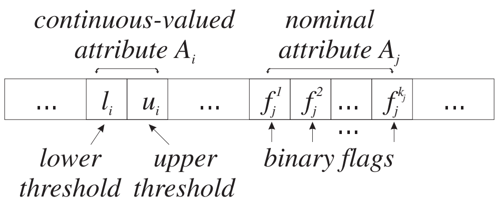

Fig. 1. Representation of the left-hand side of a decision rule ($k_j = |V(A_j)|$).

examples in the sequence sorted by the increasing value of A_i, that one of the examples is positive and the other is negative. Fayyad and Irani proved [3], that evaluating only the boundary thresholds is sufficient for finding the maximum of *information gain* measure. This property also holds for the fitness function (1).

For a nominal attribute A_j the substring consists of binary flags: $f_j^1, f_j^2, \ldots, f_j^{|V(A_j)|}$; each of them corresponds to one value of the attribute. If e.g. the domain of attribute A_j is $\{low, moderate, high\}$ then the pattern 011 represents condition $A_j = (moderate \vee high)$, which stands for: "the value of A_j equals *moderate* or *high*". Setting all the bits means that the condition is not present on the LHS.

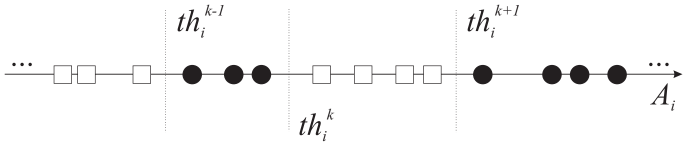

Fig. 2. An example illustrating the notion of boundary threshold. The boundary thresholds $th_i^1, \ldots, th_i^k, \ldots, th_i^{NT_i}$ for the continuous-valued attribute A_i are placed between groups of positive (•) and negative (□) examples.

Each chromosome in the population is initialized using a randomly chosen positive example. The initialization method assures that decision rule corresponding to the chromosome will cover given example.

2.2 The fitness function

Consider a string S encoding a decision rule, which covers POS positive examples and NEG negative ones. Its fitness is defined [5] by the equation:

$$f(S) = \frac{POS^{\alpha}}{L/N + 1} PE(x), \tag{1}$$

where

$$x = \frac{NEG}{POS + NEG} - \beta,$$

L is the number of conditions constituting the left-hand side of the rule, α and β ($0 \leq \beta \leq 1$) are two users-supplied parameters, $PE(x)$ is the function, which significantly degrades the fitness of the rule when the proportion of the number of covered negative examples to the total number of covered examples is greater than β. It takes the form of the sigmoidal function:

$$PE(x) = \frac{1}{1 + \exp(\gamma_1 (x - \gamma_2))},$$

where $\gamma_1 = 30$ and $\gamma_2 = 0.05$.

2.3 Evolutionary operators

Three genetic search operators are employed by the EA: *crossover*, *changing condition* and *insertion*.

The first operator is a modification of the *uniform crossover* , which exchanges complete substrings rather than single genes.

The changing condition is a mutation-like operator, which alters a single condition related to an attribute A_i. If A_i is nominal a flag randomly chosen from $f_i^1, f_i^2, \ldots, f_i^{|V(A_i)|}$ is flipped. For a continuous-valued attribute a threshold (l_i or u_i) is replaced by a random boundary threshold.

The insertion operator modifies a chromosome allowing the corresponding decision rule to cover a new random positive example e. All conditions in the rule, which conflict with e have to be altered. In case of a condition related to a nominal attribute A_i the flag, which corresponds to $A_i(e)$, is set. If A_i is a continuous-valued attribute and the condition $l_i < A_i \leq u_i$ is not satisfied because $u_i < A_i(e)$ the threshold u_i is replaced by $\hat{u}_i$, where $\hat{u}_i$ is the smallest boundary threshold such that $\hat{u}_i > A_i(e)$. The case when $A_i \leq l_i$ is handled in a similar way.

As a *selection operator* we use the proportional elitist selection with linear scaling.

3 Experiments

In this section some initial experimental results are presented. We have tested our method on several datasets from UCI repository [7] . Table 1 describes these datasets and shows the classification accuracy. The accuracy was estimated by running ten times the complete ten-fold crossvalidation. The mean of ten runs and the standard deviation of this mean are presented.

4 Conclusions

The results of initial experiments are promising. Moreover, we are currently enhancing our method to improve the performance.

Several directions of future research exist. At present the EA is able to search only for one decision rule. The set of (possible many) rules describing the class is generated

Dataset	Attributes	Examples	Classes	Accuracy
breast-w	9	699	2	95.7 ± 0.5
pima	8	768	2	73.9 ± 0.4
wine	13	178	3	93.7 ± 0.8
iris	4	150	3	95.0 ± 0.8
australian	14	690	2	85.6 ± 0.5
glass	9	214	7	67.3 ± 2.7

Table 1. The results of the experiments. The classification accuracy was estimated by the mean of ten runs of 10-fold crossvalidation.

sequentially by multiple calls for the EA. We intend to modify the EA in a way, which will allow it to search for complete rulesets rather than single rules. Since the number of rules in the optimal ruleset covering a class is not known, the EA will be adapted to processing of variable-length chromosomes.

In some cases the number of boundary thresholds for an attribute is excessively high, which potentially decreases the performance of the EA. A possible solution to this problem could be the exclusion of some boundary thresholds from the search.

Acknowledgments The authors are grateful to Prof. Leon Bobrowski for his support and useful comments. This work was supported by the grant W/II/1/97 from Technical University of Bialystok.

References

1. Clark, P., Niblett, T.: The CN2 induction algorithm. *Machine Learning* 3 (1989) 261-283.
2. De Jong, K., Spears, W.M., Gordon, D.F.: Using genetic algorithm for concept learning. *Machine Learning* 13 (1993) 168-182.
3. Fayyad, U., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. of IJCAI'93.* Morgan Kaufmann (1993) 1022-1027.
4. Janikow, C.: A knowledge intensive genetic algorithm for supervised learning. *Machine Learning* 13 (1993) 192-228.
5. Kwedlo, W., Krętowski, M.: Discovery of decision rules from databases: an evolutionary approach. In *Principles of Data Mining and Knowledge Discovery.* 2^{nd} *European Symposium PKDD'98.* Springer LNCS 1510 (1998) 370-378.
6. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3^{rd} edn. Springer (1996).
7. Murphy, P., Aha, D.: *UCI repository of machine-learning databases*, available on-line: http://www.ics.uci.edu/pub/machine-learning-databases.
8. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann (1993).