Attribute Selection with a Multiobjective Genetic Algorithm

Gisele L. Pappa, Alex A. Freitas, Celso A.A. Kaestner

Pontifícia Universidade Catolica do Parana (PUCPR),
Postgraduated Program in Applied Computer Science,
Rua Imaculada Conceicao, 1155, Curitiba - PR, 80215-901, Brazil
{gilpappa, alex, kaestner}@ppgia.pucpr.br
http://www.ppgia.pucpr.br/~alex

Abstract. In this paper we address the problem of multiobjective attribute selection in data mining. We propose a multiobjective genetic algorithm (GA) based on the wrapper approach to discover the best subset of attributes for a given classification algorithm, namely C4.5, a well-known decision-tree algorithm. The two objectives to be minimized are the error rate and the size of the tree produced by C4.5. The proposed GA is a multiobjective method in the sense that it discovers a set of non-dominated solutions (attribute subsets), according to the concept of Pareto dominance.

1 Introduction

The amount of data stored in real-world databases grows much faster than our ability to process them, so that the challenge is to extract useful knowledge from data. This is the core idea of the field of data mining, where the goal is to discover, from real-world data sets, knowledge that is accurate, comprehensible, and useful for the user.

The complexity and large size of real-world databases have motivated researchers and practitioners to reduce the data dimensionality, for data mining purposes. This dimensionality reduction can be done in two main directions. First, one can select a random sample of the available records, reducing the number of records to be mined. Second, one can select a subset of the available attributes [10], [11], which can also reduce the number of records to be mined. The latter is the direction followed in this paper.

Attribute selection is important because most real-world databases were collected for purposes other than data mining [8]. Hence, a real-world database can have many attributes that are irrelevant for data mining, so that by discarding the irrelevant attributes one can actually improve the performance of a data mining algorithm. In addition, providing a data mining algorithm with a subset of attributes reduces the computational time taken by that algorithm, by comparison with using the entire set of available attributes. As a result, attribute selection is an active research area in data mining.

The goal of attribute selection is to discover a subset of attributes that are relevant for the target data mining task. In this paper we address the task of classification, where the goal is to predict the class of an example (a record) based on the values of the predictor attributes for that example. In the context of this task, in general two important objectives of attribute selection are to minimize the error rate of the classification algorithm and the complexity of the knowledge discovered by that algorithm. These are the two objectives to be minimized in this work.

Note that attribute selection, like many other data mining problems, involve the "simultaneous" optimization of more than one objective. However, such a simultaneous optimization is not always possible. The objectives to be optimized can be conflicting with one another, and they normally are non-commensurable – i.e., they measure different aspects of the target problem.

In order to solve problems involving more than one objective, recently there has been a growing amount of research in the area of multiobjective optimization [3]. The basic idea is to return to the user, as the result of the problem-solving algorithm, a set of optimal solutions (rather than a single solution) by taking both objectives into account, without *a priori* assigning greater priority to one objective or the other. The ultimate choice about which solution should be used in practice is left to the user, which can use his/her background knowledge and experience to choose the "best" solution for his/her needs *a posteriori*, among all the returned optimal solutions. The motivation for multiobjective optimization will be discussed in more detail in section 3.

Casting the attribute selection problem as a multiobjective optimization problem, this paper proposes a multiobjective genetic algorithm (GA) for attribute selection in the classification task of data mining. The paradigm of GA was chosen for the development of our attribute selection method mainly for the following reasons. First, GAs are a robust search method, capable of effectively exploring large search spaces, which is usually the case in attribute selection. Note that the size of the search space in attribute selection is 2^M, where M is the number of attributes – i.e., the size of the search space grows exponentially with the number of attributes. Second, unlike many search algorithms which perform a local, greedy search, GAs perform a global search [5]. In the context of data mining, this global search means that GAs tend to cope better with attribute interaction than greedy search methods [6], [7]. Finally, it is important to notice that multiobjective optimization requires a problem-solving algorithm that is capable of considering a set of optimal solutions at each iteration, and this requirement is naturally satisfied by GAs, which work with a population of individuals, or candidate solutions [3].

The remainder of this paper is organized as follows. Section 2 reviews the main concepts of attribute selection. Section 3 discusses multiobjective optimization. Section 4 describes the proposed multiobjective GA for attribute selection. Section 5 reports computational results. Finally section 6 presents the conclusions of this work.

2 Attribute Selection

Attribute selection is one of the main preprocessing tasks for the application of a data mining algorithm [10]. As mentioned in the Introduction, the general goal of attribute selection is to select a subset of attributes that are relevant for the target data mining task, out of all available attributes. In the classification task, which is the task addressed in this work, an attribute is deemed relevant if it is useful for discriminating examples belonging to different classes.

More specific goals of attribute selection are as follows:

- Improving the performance of a data mining algorithm with respect to several criteria, such as reducing the classification error rate and/or complexity (size) of discovered knowledge and reducing the processing time of the data mining algorithm;
- Removing noisy and/or irrelevant attributes, reducing the dimensionality of the data (which not only helps to improve the performance of the data mining algorithm, but also saves storage space, in the case of very large data sets).

There are many methods that can be used for attribute selection. They can be characterized mainly with respect to the search strategy used to explore the space of candidate attribute subsets and with respect to the evaluation function used to measure the quality of a candidate attribute subset.

With respect to the search strategy, two well-known methods are forward sequential selection (FSS) and backward sequential selection (BSS) [10]. In essence, FSS starts with an empty set of selected attributes and it adds one attribute at a time to that set until a stopping criterion is met – e.g., until the quality of the current set of selected attributes cannot be improved by adding another attribute to that set. BSS follows the opposite strategy. It starts with the full set of original attributes, and it removes one attribute at a time from that set until a stopping criterion is met. In both methods, the attribute chosen to be added or removed at each step is the one maximizing the value of some evaluation function. Hence, both are greedy methods, working with one attribute at a time, and therefore having the drawback of being sensitive to attribute interaction.

With respect to the evaluation function, attribute selection methods can be classified into the filter approach or the wrapper approach. This classification is independent of the search strategy used by the attribute selection method. It depends on whether or not the evaluation function uses the target data mining algorithm (which will be eventually applied to the ultimate set of selected attributes) to evaluate the quality of a candidate attribute subset. In the filter approach the attribute selection method does not use the data mining algorithm, whereas in the wrapper approach the attribute selection method uses the data mining algorithm to evaluate the quality of a candidate attribute subset. Note that the data mining algorithm is used as a black box.

The wrapper approach tends to obtain a predictive accuracy better than the filter approach, since it finds an attribute subset "customized" for a given data mining algorithm. However, the wrapper approach is considerably more computationally expensive than the filter approach, since the former requires many runs of the data mining algorithm.

3 Multiobjective Optimization

Many real-world problems involve the optimization of multiple objectives. However, the majority of methods used to solve these problems avoids the complexities associated with multiobjective optimization. As a result, many methods have been proposed to convert multiobjective problems into single objective problems [3]. Some of them can be found in [9], [14].

Once several conversion methods are available, it has been forgotten that, in reality, a single objective optimization problem is a degenerated case of a multiobjective optimization problem, and that there are crucial differences between the two kinds of problems. The main difference concerns the desired number of optimal solutions. In single objective optimization one usually wants to discover a single optimal solution. By contrast, assuming that the different objectives to be optimized represent conflicting goals (such as improving the quality of a product and reducing its cost), in multiobjective optimization the optimization of each objective corresponds to an optimal solution. Therefore, in multiobjective optimization one usually wants to discover several optimal solutions, taking all objectives into account, without assigning greater priority to one objective or the other. The ultimate choice about which solution should be used in practice is left to the user, which can use his/her background knowledge and experience to choose the "best" solution for her needs, among all returned optimal solutions.

In other words, in a multiobjective optimization framework the user has the advantage of being able to choose the solution representing the best trade-off between conflicting objectives *a posteriori*, after examining a set of high-quality solutions returned by the multiobjective problem-solving algorithm. Intuitively, this is better than forcing the user to choose a trade-off between conflicting goals *a priori*, which is what is done when a multiobjective optimization problem is transformed into a single-objective one.

In multiobjective optimization, in order to take all the objectives into account as a whole during the search for optimal solutions, one uses the concept of Pareto dominance, as follows. A given solution x_1 dominates another solution x_2 if and only if:

1. Solution x_1 is not worse than solution x_2 in any of the objectives;
2. Solution x_1 is strictly better than solution x_2 in at least one of the objectives.

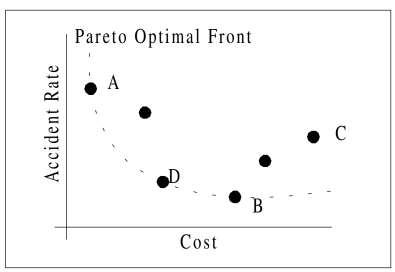

Fig. 1: Example of Pareto dominance in a two-objective problem [2]

The solutions that are not dominated by any other solution are considered Pareto-optimal solutions. Figure 1 shows a set of possible solutions for a hypothetical problem with two objectives to be minimized, namely accident rate and cost. Note that solution A has a small cost but a large accident rate. Solution B has a large cost but a small accident rate. Assuming that minimizing both objectives is important, one cannot say that solution A is better than B, nor vice-versa. In addition, solution D cannot be considered better than A or B. The three solutions A, B, and D are Pareto-optimal solutions: none of them is dominated by other solutions. These solutions are included in the Pareto front, represented by the dotted line in Figure 1. Note that solution C is not a Pareto-optimal solution, since it is dominated, for instance, by solution B (which is better than C with respect to both objectives).

4 The Proposed Multiobjective GA for Attribute Selection

A genetic algorithm (GA) is a search algorithm inspired by the principle of natural selection. The basic idea is to evolve a population of individuals, where each individual is a candidate solution to a given problem. Each individual is evaluated by a fitness function, which measures the quality of its corresponding solution. At each generation (iteration) the fittest (the best) individuals of the current population survive and produce offspring resembling them, so that the population gradually contains fitter and fitter individuals – i.e., better and better candidate solutions to the underlying problem. For a comprehensive review of GAs in general the reader is referred to [12], [5]. For a comprehensive review of GAs applied to data mining the reader is referred to [7].

This work proposes a multiobjective genetic algorithm (GA) for attribute selection. As mentioned in the Introduction, our motivation for developing a GA for attribute selection, in a multiobjective optimization framework, was that: (a) GAs are a robust search method, capable of effectively exploring the large search spaces often associated with attribute selection problems; (b) GAs perform a global search [7], [4], so that they tend to cope better with attribute interaction than greedy search methods [6], [7], which is also an important advantage in attribute selection; and (c) GAs already work with a population of candidate solutions, which makes them naturally suitable for multiobjective problem solving [3], where the search algorithm is required to consider a set of optimal solutions at each iteration.

The goal of the proposed GA is to find a subset of relevant attributes that leads to a reduction in both the classification error rate and the complexity (size) of the rule set discovered by a data mining algorithm (improving the comprehensibility of discovered knowledge).

In this paper the data mining algorithm is C4.5 [15], a very well-known decision tree algorithm. The proposed GA follows the wrapper approach, evaluating the quality of a candidate attribute subset by using the target classification algorithm (C4.5). Hence, the fitness function of the GA is based on the error rate and on the size of the decision tree built by C4.5. These two criteria (objectives) are to be minimized according to the concept of Pareto dominance. The main aspects of the proposed GA are described in the next subsections.

4.1 Individual Encoding

In the proposed GA, each individual represents a candidate subset of selected attributes, out of all original attributes. Each individual consists of M genes, where M is the number of original attributes in the data being mined. Each gene can take on the value 1 or 0, indicating that the corresponding attribute occurs or not (respectively) in the candidate subset of selected attributes.

4.2 Fitness Function

The fitness (evaluation) function measures the quality of a candidate attribute subset represented by an individual. Following the principle of multiobjective optimization, the fitness of an individual consists of two quality measures: (a) the error rate of C4.5;

and (b) the size of the decision tree built by C4.5. Both (a) and (b) are computed by running C4.5 with the individual's attribute subset only, and by using a hold-out method to estimate C4.5's error rate, as follows. First, the training data is partitioned into two mutually-exclusive data subsets, the building subset and the validation subset. Then we run C4.5 using as its training set only the examples (records) in the building subset. Once the decision tree has been built, it is used to classify examples in the validation set. The two components of the fitness vector are then the error rate in the validation set and the size (number of nodes) of the tree built by C4.5.

4.3 Selection Method and Genetic Operators

At each generation (iteration) of the GA, the selection of individuals to reproduce is performed as follows. First the GA selects all the non-dominated individuals (the Pareto front) of the current population. These non-dominated individuals are passed unaltered to the next generation by elitism [1]. Elitism is a common procedure in GAs, and it has the advantage of avoiding that good individuals disappear from the population due to the stochastic nature of selection.

Let N be the total population size (which is fixed for all generations, as usual in GAs), and let N_{elit} be the number of individuals reproduced by elitism. Then the other N - N_{elit} individuals to reproduce are chosen by performing N - N_{elit} times a tournament selection procedure [12], as follows. First, the GA randomly picks k individuals from the current population, where k is the tournament size, a user-specified parameter which was set to 2 in all our experiments. Then the GA compares the fitness values of the two individuals playing the tournament and selects as the winner the one with the best fitness values.

The selection of the best individual is based on the concept of Pareto dominance, taking into account the two objectives to be minimized (error rate and decision tree size). Given two individuals I_1 and I_2 playing a tournament, there are two possible situations. The first one is that one of the individuals dominates the other. In this case the former is selected as the winner of the tournament.

The second situation is that none of the individuals dominates the other. In this case, as a tie-breaking criterion, we compute an additional measure of quality for each individual by taking both objectives into account. Following the principle of Pareto dominance, care must be taken to avoid that this tie-breaking criterion assigns greater priority to any of the objectives. Hence, we propose the following tie-breaking criterion. For each of the two individuals I_i, i=1,2, playing a tournament, the GA computes X_i as the number of individuals in the current population that are dominated by I_i, and Y_i as the number of individuals in the current population that dominate Ii. Then the GA selects as the winner of the tournament the individual I_i with the largest value of the formula: X_i - Y_i. Finally, if I_1 and I_2 have the same value of the formula X_i-Y_i (which is rarely the case), the tournament winner is simply chosen at random.

Individuals selected by tournament selection undergo the action of two standard genetic operators, crossover and mutation, in order to create new offspring [12]. In essence, crossover consists of swapping genes (bits, in our individual encoding) between two individuals, whereas mutation replaces the value of a gene with a new randomly-generated value. In our individual encoding, where each gene is a bit, mutation consists simply of flipping the value of a bit. These operators are applied with user-specified probabilities. In all our experiments the probabilities of crossover and mutation were set to 80% and 1%, respectively, which are relatively common values in the literature.

The population size N was set to 100 individuals, which evolve for 50 generations. These values were used in all our experiments. The pseudocode of the GA is shown, at a high level of abstraction, in Algorithm 1. (Note that this pseudocode abstracts away details such as the fact that crossover and mutation are applied with user-defined probabilities. It shows only an overview of the flow of processing of the GA.)

```
Create initial population
FOR EACH generation DO
 FOR EACH Individual DO
Run C4.5 with attribute subset represented by the individual
Compute multiobjective fitness // error rate and tree size
 END FOR
 Add non-dominated individuals to next generation's population
 FOR i ← 1 to (N - N_elit)/2 DO
   Perform tournament selection twice, to select two parent
    individuals, P1 and P2
   Perform crossover of P1 and P2, producing children C1 and C2
   Perform mutation on C1 and C2
 Add C1 and C2 to the next generation's population
 END FOR
END FOR
Compute fitness of the individuals of the last generation
Return all non-dominated individuals of the last generation
```

Algorithm 1. Pseudocode of the proposed multiobjective GA

5 Computational Results

We have performed experiments with six public-domain, real-world data sets obtained from the UCI (University of California at Irvine)'s data set repository [13]. The number of examples, attributes and classes of these data sets is shown in Table 1.

Table 1. Main characteristics of the data sets used in the experiments

Data Set	# examples	# attributes	# classes
Dermatology	366	36	6
Vehicle	846	18	4
Promoters	106	57	2
Ionosphere	351	34	2
Crx	690	15	2
Arritymia	452	269	16

All the experiments were performed by using a well-known 10-fold stratified cross-validation procedure, as follows. For each data set, all the available data is divided into 10 mutually-exclusive and exhaustive partitions having approximately the same size. In addition, each partition has approximately the same class distribution (stratified cross validation). Then the GA and C4.5 are run 10 times. In the i-th run of the algorithms, i=1,...,10, the i-th partition is used as the test set and the other 9 partitions are used as the training set. All results reported in this paper refer to average results in the test set over the 10 iterations of the cross-validation procedure.

At each iteration of the cross-validation procedure, a GA run is performed as follows. For each individual of the GA, out of the 9 partitions used as the training set, 8 partitions (the building subset mentioned in subsection 4.2) are used by C4.5 to build a decision tree, and the remaining partition (the validation subset mentioned in subsection 4.2) is used to compute the error rate of C4.5. We emphasize that the examples in the test set are never used during the evolution of the GA.

Finally, for each iteration of the cross-validation procedure, once the GA run is over we compare the performance of C4.5 using all the original attributes with the performance of C4.5 using only the attributes selected by the GA. In both runs of C4.5, the decision tree is built using the entire training set (9 partitions), and then we measure C4.5's error rate in the test set.

Therefore, the GA can be considered successful to the extent that the attributes subsets selected by it lead to a reduction in the error rate and size of the tree built by C4.5, by comparison with the use of all original attributes.

There is a final point concerning the evaluation of the solutions returned by the GA. It should be noted that, as explained before, the solution for a multiobjective optimization problem consists of all non-dominated solutions (the Pareto front). Hence, each run of the GA outputs the set of all non-dominated solutions (attribute subsets) present in the last generation's population. In a real-world application, it would be left to the user the final choice of the solution to be used in practice. However, in our research-oriented work, involving public-domain data sets, no user was available. Hence, in order to evaluate the quality of the non-dominated attribute subsets found by the GA in an automatic, data-driven manner – as usual in the majority of the data mining and machine learning literature – we measure the error rate and the size of the decision tree built by C4.5 using each of the non-dominated attribute subsets returned by the GA. The ultimate results associated with the attributes selected by the GA, which are the results reported in the following, are the corresponding arithmetic average over all non-dominated solutions returned by the GA.

The results of our experiments are reported in Table 2. The first column indicates the name of the data set. The second and third columns indicate the error rate obtained by C4.5 using only the attributes selected by the GA and using all original attributes, respectively. The fourth and fifth columns indicate the size of the decision tree built by C4.5 using only the attributes selected by the GA and using all original attributes, respectively. In each cell of the table, the value before the "±" symbol is the average result over the 10 iterations of the cross-validation procedure, and the value after the "±" symbol is the corresponding standard deviation. In addition, in the second and fourth columns the values of a given cell are shown in bold when the corresponding result in that cell is significantly better than the result in the third and fifth columns, respectively. A result is considered significantly better than another when the corresponding intervals, taking into account the standard deviations, do not overlap.

As shown in Table 2, the error rate associated with the attributes selected by the GA is better than the one associated with all attributes in three data sets, a nd the difference between the two error rates is significant in one data set. In the other three data sets, although the error rate associated with the attributes selected by the GA is somewhat worse than the one associated with all attributes, the differences between the two error rates are not significant – i.e., the corresponding intervals (taking into account the standard deviations) overlap.

Table 2. Computational Results with 10-fold stratified cross validation

Data Set	Error Rate (%)		Decision Tree Size	
	C4.5 + GA	C4.5 alone	C4.5 + GA	C4.5 alone
Dermatology	5.5 ± 1.46	4.2 ± 0.96	**14.8 ± 1.08**	17.1 ± 0.34
Vehicle	29.9 ±0.70	29.6 ±1.15	**151.9 ± 8.32**	181 ±3.24
Promoters	**14.1 ± 4.02**	21.2 ± 3.05	16.8 ± 1.32	17.6 ± 0.99
Ionosphere	10.2 ± 1.16	8.5 ±1.20	**20.8 ±1.62**	24 ±1.2
Crx	14.4 ± 1.38	16.3 ± 1.2	**8.6 ± 0.71**	69.4 ± 2.72
Arritymia	31.6 ± 2.6	32.0 ± 2.36	**64.1 ± 2.3**	75.4 ± 1.7

In Table 2 we also note that the tree size associated with the attributes selected by the GA is better than the one associated with all attributes in all the six data sets, and the difference is significant in five data sets.

In summary, the use of the GA has led to a significant reduction in the size of the trees built by C4.5 in five data sets, without significantly increasing C4.5's error rate in any data set – and even significantly reducing C4.5's error rate in one data set.

One disadvantage of the use of the GA is that it is computationally expensive. In the two largest data sets used in our experiments, Vehicle (with the largest number of examples) and Arritymia (with the largest number of attributes, viz. 269), a single run of the GA took about 25 minutes and 5 hours and 15 minutes, respectively; whereas a single run of C4.5 took less than one minute and one and half minute, respectively. (The results were obtained in a Pentium-IV PC with clock rate of 1.7 GHz and 512 Mb of RAM.) We believe the increase in computational time associated with the GA is a relatively small price to pay for its associated increase in the comprehensibility of discovered knowledge. Data mining is typically an off-line task, and it is well-known that in general the time spent on running a data mining algorithm is a small fraction (less than 20%) of the total time spent with the entire knowledge discovery (KD) process. Hence, in many applications, even if a data mining algorithm is run for several days, this is acceptable, at least in the sense that it is not the bottleneck of the KD process. In any case, if necessary the computational time associated with the GA can be greatly reduced by using parallel processing techniques, since GAs can be easily parallelized [7].

6 Conclusions and Future Work

In this paper we have proposed a multiobjective genetic algorithm (GA) for attribute selection in the classification task of data mining. The goal of the GA is to select a subset of attributes that minimizes both the error rate and the size of the decision tree built by C4.5. The latter objective involves a commonplace measure of simplicity (or comprehensibility) in the data mining and machine learning literature. The smaller the size of a decision tree, the simpler it is, and so the more comprehensible to the user it tends to be. We emphasize that, in data mining, maximizing comprehensibility tends to be at least as important as minimizing error rate [7], [15]. In order to minimize the two objectives at the same time, the GA uses the concept of Pareto dominance, so that each GA run returns, as its output, the set of all non-dominated solutions found during the search.

We have done experiments with six data sets, comparing the error rate and the size of the decision tree built by C4.5 in two cases: using only the attributes selected by the GA and using all attributes. The results of these experiments have shown that, on average over all non-dominated solutions (attribute subsets) returned by the GA, the use of the GA as an attribute selection method has led to: (a) a significant reduction of the size of the tree built by C4.5 in five out of the six data sets; and (b) a significant reduction of C4.5's error rate in one data set. There was no case where the use of the GA as an attribute selection method has led to an error rate or tree size significantly worse than the ones associated with the use of all attributes.

With respect to future research, we have noted that in some cases the GA population converges very fast, possibly corresponding to a premature convergence of the population. We are currently investigating the use of a niching method to promote greater population diversity, in order to reduce this premature convergence problem.

References

1. Bhattacharyya, S., Evolutionary Algorithms in Data mining: Multi-Objective Performance Modeling for Direct Marketing. In: Proc KDD-2000, ACM Press (2000) 465-471
2. Deb, K, Multi-Objective Evolutionary Algorithms: Introducing Bias Among Pareto-Optimal Solutions. Kanpur Genetic Algorithms Laboratory Report nº 99002, India (1999)
3. Deb, K., Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, England (2001)
4. Fidelis, M.V., Lopes, H.S., Freitas, A.A., Discovering Comprehensible Classification Rules with a Genetic Algorithm. In: Proc. Congress on Evolutionary Computation (2000)
5. Goldberg, D. E., Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company (1989)
6. Freitas, A. A.; Understanding the Crucial Role of Attribute Interaction in Data Mining. In: Artificial Intelligence Review 16, Kluwer Academic Publishers (2001) 177-199
7. Freitas, A.A., Data Mining and Knowledge Discovery with Evolutionary Algorithms (forthcoming book). Springer-Verlag (2002)
8. Holsheimer, M., Siebes, A., Data Mining – The Search for Knowledge in Databases. Report CS-R9406, Amsterdam: CWI (1991)
9. Ishibuchi, H., Nakashima, T., Multi-objective Pattern and Feature Selection by a Genetic Algorithm. In: Proc. Genetic and Evolutionary Computation Conf. (GECCO–2000), Morgan Kaufmann (2000) 1069-1076
10. Liu, H.; Motoda, H., Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers (1998)
11. Martín-Bautista, M. J., Vila, M. A., A Survey of Genetic Feature Selection in Mining Issues. In: Proc. IEEE Conference on Evolutionary Computation, Washington (1999) 1314-1321.
12. Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag (1996)
13. Murphy, P.M., Aha, D.W., UCI Repository of Machine Learning databases. [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Departament of information and Computer Science (1994)
14. Rozspyal, A., Kubat, M., Using Genetic Algorithm to Reduce the Size of a Nearest-Neighbor Classifier and Select Relevant Attributes. Proc. Int. Conf. Machine Learning (ICML-2001), Morgan Kauf. (2001)
15. Quinlan, J.R., C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)