Evolution of multi-adaptive discretization intervals for a rule-based genetic learning system

Jaume Bacardit and Josep Maria Garrell

Intelligent Systems Research Group
Enginyeria i Arquitectura La Salle,
Universitat Ramon Llull,
Psg. Bonanova 8, 08022-Barcelona,
Catalonia, Spain, Europe. {jbacardit,josepmg}@salleURL.edu

Abstract. Genetic Based Machine Learning (GBML) systems traditionally have evolved rules that only deal with discrete attributes. Therefore, some discretization process is needed in order to teal with real-valued attributes. There are several methods to discretize real-valued attributes into a finite number of intervals, however none of them can efficiently solve all the possible problems. The alternative of a high number of simple uniform-width intervals usually expands the size of the search space without a clear performance gain. This paper proposes a rule representation which uses adaptive discrete intervals that split or merge through the evolution process, finding the correct discretization intervals at the same time as the learning process is done.

1 Introduction

The application of Genetic Algorithms (GA) [10, 8] to classification problems is usually known as Genetic Based Machine Learning (GBML), and traditionally it has been addressed from two different points of view: the Pittsburgh approach, and the Michigan approach, early exemplified by LS-1 [20] and CS-1 [11], respectively.

The classical knowledge representation used in these systems is a set of rules where the antecedent is defined by a prefixed finite number of intervals to handle real-valued attributes. The performance of these systems is tied to the right election of the intervals.

In this paper we use a rule representation with adaptive discrete intervals. These intervals are splitted and merged through the evolution process that drives the training stage. This approach avoids the higher computational cost of the approaches which work directly with real values and finds a good discretization only expanding the search space with small intervals when necessary. This representation was introduced in [1] and the work presented in this paper is its evolution, mainly focused on generalizing the approach and simplifying the tuning needed for each domain.

This rule representation is compared across different domains against the traditional discrete representation with fixed intervals. The number and size of the fixed intervals approach is obtained with two methods: (1) simple uniform-width intervals and (2) intervals obtained with the Fayyad & Irani method [7], a well-known discretization algorithm. The aim of this comparison is two-fold: measure the accuracy performance and the computational cost.

The paper is structured as follows. Section 2 presents some related work. Then, we describe the framework of our classifier system section 3. The adaptive intervals rule representation is explained in section 4. Next, section 5 describes the test suite used in the comparison. The results obtained are summarized in section 6. Finally, section 7 discusses the conclusions and some further work.

2 Related work

There are several approaches to handle real-valued attributes in the Genetic Based Machine Learning (GBML) field. Early approaches use discrete rules with a large number of prefixed uniform discretization intervals. However, this approach has the problem that the search space grows exponentially, slowing the evolutionary process without a clean accuracy improvement of the solution [2]

Lately, several alternatives to the discrete rules have been presented. There are rules composed by real-valued intervals (XCSR [22], [4], COGITO [18]). MOGUL [5], uses a fuzzy reasoning method. This method generates sequentially: (1) fuzzy rules, and then (2) fuzzy membership functions. Recently, GALE [15] proposed a knowledge independent method for learning other knowledge representations like instance sets or decision trees. All those alternatives present better performance, but usually they also have higher computational cost [18].

A third approach is to use a heuristic discretization algorithm. Some of these methods work with information entropy [7], the χ^2 statistic [14] or multi-dimensional non-uniform discretization [13]. These algorithms are usually more accurate and faster than the uniform discretization. However, they suffer a lack of robustness across some domains [1].

3 Framework

In this section we describe the main features of our classifier system. GAssist (*Genetic Algorithms based claSSIfier sySTem*) [9] is a Pittsburgh style classifier system based on GABIL [6]. Directly from GABIL we have borrowed the representation of the discrete rules (rules with conjunctive normal form (CNF) predicates), the semantically correct crossover operator and the fitness computation (squared accuracy).

Matching strategy: The matching process follows a “if ... then ... else if ... then...” structure, usually called *Decision List* [19].

Mutation operators: The system manipulates variable-length individuals, making more difficult the tuning of the classic gene-based mutation probability. In order to simplify this tuning, we define p_{mut} as the probability i of mutating an individual. When an individual is selected for mutation (based on p_{mut}), a random gene is chosen inside its chromosome for mutation.

Control of the individuals length: Dealing with variable-length individuals arises some serious considerations. One of the most important ones is the control of the size of the evolving individuals [21]. This control is achieved ins GAssist using two different operators:

- *Rule deletion* This operator deletes the rules of the individuals that do not match any training example. This rule deletion is done after the fitness computation and has two constraints: (a) the process is only activated after a predefined number of iterations, to prevent a massive diversity loss and (b) the number of rules of an individual never goes below a lower threshold. This threshold is assigned to the number of classes of the domain.
- *Selection bias using the individual size* Selection is guided as usual by the fitness (the accuracy). However, it also gives certain degree of relevance to the size of the individuals, having a policy similar to multi-objective systems. We use tournament selection because its local behavior lets us implement this policy. The criterion of the tournament is given by an operator called "size-based comparison" [2]. This operator considers two individuals similar if their fitness difference is below a certain threshold (d_{comp}). Then, it selects the individual with fewer number of rules.

4 Discrete rules with adaptive intervals

This section describes the rule representation based on discrete rules with adaptive intervals. First we describe the problems that traditional discrete rules present. Then, we explain the adaptive intervals rules proposed and the changes introduced in order to enable the GA to use them.

4.1 Discrete rules and unnecessary search space growth

The traditional approach to solve problems with real-valued attributes using discrete rules has been done using a discretization process. This discretization can be done using algorithms which determine the discretization intervals analyzing the training information or we can use a simple alternative like using an uniform-width intervals discretization.

In the latter method, the way to increase the accuracy of the solution is to increase the number of intervals. This solution brings a big problem because the search space to explore grows in an exponential degree when more intervals are added. The improvement in accuracy expected increasing the number of intervals does not exist sometimes, because the GA spends too much time exploring areas of the search space which do not need to be explored.

If we find a correct and minimal set of intervals, the solution accuracy will probably increase without a huge increase of the computational cost.

4.2 Finding good and minimal intervals

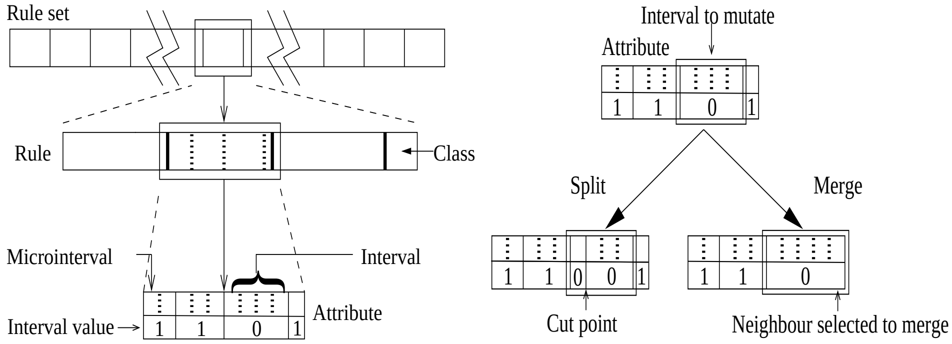
Our aim is to find good discretization intervals without a great expansion of the search space. In order to achieve this goal we defined a rule representation [1] with discrete adaptive intervals where the discretization intervals are not fixed. These intervals are evolved through the iterations, merging and splitting between them.

To control the computational cost and the growth of the search space, we define the next constraints:

- A number of “low level” uniform and static intervals is defined for each attribute called *micro-intervals*.
- The adaptive intervals are built joining together *micro-intervals*.
- When we split an interval, we select a random point in its *micro-intervals* to break it.
- When we merge two intervals, the value of the resulting interval is taken from the one which has more *micro-intervals*. If both have the same number of *micro-intervals*, the value is chosen randomly.
- The number and size of the initial intervals is selected randomly.

The adaptive intervals as well as the split and merge operators are shown in figure 1.

Fig. 1. Adaptive intervals representation and the split and merge operators.



To apply the split and merge operators we have added to the GA cycle two special phases applied to the offspring population after the mutation phase. For each phase (split and merge) we have a probability (p_{split} or p_{merge}) of applying a split or merge operation to an individual. If an individual is selected for splitting or merging, a random point inside its chromosome is chosen to apply the operation.

Finally, this representation requires some changes in some other parts of the GA:

- The crossover operator can only take place in the attribute boundaries.
- The “size-based comparison” operator uses the length (number of genes) of the individual instead of the number of rules, because now the size of a rule can change when the number of intervals that it contains change. This change also makes the GA prefer the individuals with fewer intervals in addition to fewer rules, further simplifying them.

4.3 Changes to the adaptive intervals rule representation

One of the main drawbacks of the initial approach was the sizing of the number of *micro-intervals* assigned to each attribute term of the rules. This parameter is difficult to tune because it is domain-specific.

In this paper we test another approach (*multi-adaptive*) which consists in evolving attribute terms with different number of *micro-intervals* in the same population. This enables the evolutionary process to select the correct number of micro-intervals for each attribute term of the rules. The number of *micro-intervals* of each attribute term is selected from a predefined set in the initialization stage.

The initialization phase has also changed. In our previous work the number and size of the intervals was uniform. We have changed this policy to a total random initialization in order to gain diversity in the initial population.

The last change introduced involves the split and merge operators. In the previous version these operators were integrated inside the mutation. This made the sizing of the probabilities very difficult because the three operators (split, merge and mutation) were coupled. Using an extra recombination stage in this version we eliminate this tight linkage.

5 Test suite

This section summarizes the tests done in order to evaluate the accuracy and efficiency of the method presented in this paper. We also compare it with some alternative methods. The tests were conducted using several machine learning problems which we also describe.

5.1 Test problems

The selected test problems for this paper present different characteristics in order to give us a broad overview of the performance of the methods being compared.

The first problem is a synthetic problem (*Tao* [15]) that has non-orthogonal class boundaries. We also use several problems provided by the University of California at Irvine (UCI) repository [3]. The problems selected are: Pima-indians-diabetes (*pima*), *iris*, *glass* and breast-cancer-winsconsin (*breast*). Finally we will use three problems from our own private repository. The first two deal with the diagnosis of breast cancer based of biopsies (*bps* [17]) and mammograms (*mamm* [16]) whereas the last one is related to the prediction of student qualifications (*lrn* [9]). The characteristics of the problems are listed in table 1. The partition of the examples into the train and test sets was done using the *stratified ten-fold cross-validation* method [12].

5.2 Configurations of the GA to test

The main goal of the tests are to evaluate the performance of the adaptive intervals rules representation. In order to compare this method with the traditional

Table 1. Characteristics of the test problems.

Dataset	Number of examples	real attributes	discrete attributes	classes
tao	1888	2	-	2
pima	768	8	-	2
iris	150	4	-	3
glass	214	9	-	6
breast	699	-	9	2
bps	1027	24	-	2
mamm	216	21	-	2
lrn	648	4	2	5

discrete representation, we use two discretization methods, the simple uniform-width intervals method and the Fayyad & Irani method [7].

We analyze the adaptive intervals approach with two types of runs. The first one assigns the same number of *micro-intervals* to all the attribute terms of the individuals. We call this type of run **adaptive**. In the second one, attributes with different number of *micro-intervals* coexist in the same population. We well call this type **multi-adaptive**.

The GA parameters are shown in table 2. The reader can appreciate that the sizing of both p_{split} and p_{merge} is the same for all the problems except the tao problem. Giving the same value to p_{merge} and p_{split} produce solutions with too few rules and intervals, as well as less accurate than the results obtained with the configuration shown in table 2. This is an issue that needs further study.

Another important issue of the p_{split} and p_{merge} probabilities for some of the domains is that they are greater than 1. This means that for these domains at least one split and merge operation will be surely done to each individual of the population. Thus, p_{split} and p_{merge} become expected values instead of probabilities. The tuning done produces a reduction of the number of iterations needed.

6 Results

In this section present the results obtained. The aim of the tests was to compare the method presented in the paper in three aspects: accuracy and size of the solutions as well as the computational cost. Foreach method and test problem we show the average and standard deviation values of: (1) the cross-validation accuracy, (2) the size of the best individual in number of rules and intervals per attribute and (3) the execution time in seconds. The tests were executed in an AMD Athlon 1700+ using Linux operating system and C++ language.

The results were also analyzed using the two-sided t-test [23] to determine if the two adaptive methods outperform the other ones with a significance level of 1%. Finally, for each configuration, test and fold, 15 runs using different random seeds were done. Results are shown in table 3. The column titled t-test show a • beside the Uniform or Fayyad & Irani method if it was outperformed by the

Table 2. Common and problem-specific parameters of the GA.

Parameter	Value
Crossover probability	0.6
Iter. of rule eliminating activation	30
Iter. of size comparison activation	30
Sets of *micro-intervals* in the multi-adaptive test	5,6,7,8,10,15,20,25
Tournament size	3
Population size	300
Probability of mutating an individual	0.6

Code	Parameter
#iter	Number of GA iterations
d_{interv}	Number of intervals in the uniform-width discrete rules
a_{interv}	Number of *micro-intervals* in the adaptive test
d_{comp}	Distance parameter in the "size-based comparison" operator
p_{split}	Probability of splitting an individual (one of its intervals)
p_{merge}	Probability of merging an individual (one of its intervals)

Problem	Parameter					
	#iter	d_{interv}	a_{interv}	d_{comp}	p_{merge}	p_{split}
tao	600	12	48	0.001	1.3	2.6
pima	500	4	8	0.01	0.8	0.8
iris	400	10	10	0.02	0.5	0.5
glass	750	4	8	0.015	1.5	1.5
breast	325	5	10	0.01	3.2	3.2
bps	500	4	10	0.015	1.7	1.7
mamm	500	2	5	0.01	1	1
lrn	700	5	10	0.01	1.2	1.2

adaptive methods. The adaptive methods were never outperformed in the tests done, showing a good robustness.

The results are summarized using the ranking in table 4. The ranking for each problem and method is based on the accuracy. The global i rankings are computed averaging the problem rankings.

Table 3 shows that in two of the tests the best performing method was the Fayyad & Irani interval discretization technique. However, in the rest of the tests its performance is lower, showing a lack of robustness across different domains. The two adaptive tests achieved the best results of the ranking. Nevertheless, the goal of improving the rule representation with the *multi-adaptive* configuration has not been achieved. It is only better than the original *adaptive* configuration in three of the eight test problems. The computational cost is clearly the main drawback of the adaptive intervals representation. The Fayyad & Irani method is in average 2.62 times faster than it.

7 Conclusions and further work

This paper focused on an adaptive rule representation as a a robust method for finding a good discretization. The main contribution done is provided by the used of adaptive discrete intervals, which can split or merge through the evolution process, reducing the search space where it is possible.

The use of a heuristic discretization method (like the Fayyad & Irani one) outperform the adaptive intervals representation in some test problem. Never-

Table 3. Mean and deviation of the accuracy (percentage of correctly classifier examples), number of rules, intervals per attribute and execution time for each method tested. Bold entries show the method with best results for each test problem. A • mark a significant out-performance based on a t-test

Problem	Configuration	Accuracy	Number of Rules	Intervals per Rule	Time	t-test
tao	Uniform	93.7±1.2	8.8±1.6	8.3±0.0	36.0±3.5	
	Fayyad	87.8±1.1	**3.1±0.3**	**3.4±0.1**	**24.2±1.4**	•
	Adaptive	**94.6±1.3**	22.5±5.6	7.7±0.4	96.6±14.7	
	Multi-Adaptive	94.3±1.0	19.5±4.9	6.0±0.6	94.5±13.9	
pima	Uniform	73.8±4.1	6.3±2.2	3.7±0.0	**23.2±2.8**	
	Fayyad	73.6±3.1	6.6±2.6	2.3±0.2	26.4±3.0	
	Adaptive	**74.8±3.5**	6.2±2.6	2.0±0.4	56.2±9.4	
	Multi-Adaptive	74.4±3.1	**5.8±2.2**	**1.9±0.4**	59.7±8.9	
iris	Uniform	92.9±2.7	3.8±1.1	8.2±0.0	**5.2±0.7**	
	Fayyad	94.2±3.0	**3.2±0.6**	2.8±0.1	5.5±0.1	
	Adaptive	94.9±2.3	3.3±0.5	**1.3±0.2**	9.2±0.4	
	Multi-Adaptive	**96.2±2.2**	3.6±0.9	**1.3±0.2**	9.0±0.8	
glass	Uniform	60.5±8.9	8.7±1.8	3.7±0.0	**13.9±1.5**	
	Fayyad	**65.7±6.1**	8.1±1.4	2.4±0.1	14.0±1.1	
	Adaptive	64.6±4.7	**5.9±1.7**	**1.7±0.2**	35.1±5.2	
	Multi-Adaptive	65.2±4.1	6.7±2.0	1.8±0.2	38.4±5.0	
breast	Uniform	94.8±2.6	4.8±2.5	4.6±0.0	6.5±1.0	
	Fayyad	95.2±1.8	4.1±0.8	3.6±0.1	**5.8±0.4**	
	Adaptive	**95.4±2.3**	2.7±1.0	1.8±0.2	15.7±2.1	
	Multi-Adaptive	95.3±2.3	**2.6±0.9**	**1.7±0.2**	17.4±1.5	
bps	Uniform	77.6±3.3	15.0±7.0	3.9±0.0	50.8±9.0	
	Fayyad	80.0±3.1	7.1±3.8	2.4±0.1	**37.7±6.0**	
	Adaptive	**80.3±3.5**	**4.7±3.0**	2.1±0.4	106.6±21.1	
	Multi-Adaptive	80.1±3.3	5.1±2.0	**2.0±0.3**	115.9±20.5	
mamm	Uniform	63.2±9.9	2.6±0.5	2.0±0.0	**7.8±1.0**	
	Fayyad	65.3±11.1	**2.3±0.5**	2.0±0.1	8.5±0.7	
	Adaptive	**65.8±5.3**	4.4±1.7	**1.8±0.2**	27.6±4.9	
	Multi-Adaptive	65.0±6.1	4.4±1.9	1.9±0.2	27.4±5.5	
lrn	Uniform	64.7±4.9	17.8±5.1	4.9±0.0	29.2±4.0	
	Fayyad	**67.5±5.1**	14.3±5.0	4.4±0.1	**26.5±3.4**	
	Adaptive	66.1±4.6	14.0±4.6	3.6±0.3	58.9±7.9	
	Multi-Adaptive	66.7±4.1	**11.6±4.1**	**3.4±0.2**	53.9±7.2	

Table 4. Performance ranking of the tested methods. Lower number means better ranking.

Problem	Fixed	Fayyad	Adaptive	Multi-Adaptive
tao	3	4	1	2
pima	3	4	1	2
iris	4	3	2	1
glass	4	1	3	2
breast	4	3	1	2
bps	4	3	1	2
mam	4	2	1	3
lrn	4	1	3	2
Average	3.25	2.625	1.625	2
Final rank	4	3	1	2

theless, the performance increase is not significant. On the other hand, when the adaptive intervals outperform the other methods, the performance increase is higher, showing a better degree of robustness.

The overhead of evolving discretization intervals and rules at the same time is quite significant, being its main drawback. Beside the cost of the representation itself (our implementation uses twice the memory of the discrete representation for the same number of intervals) the main difference is the significant reduction of the search space achieved by a heuristic discretization.

Some further work should use the knowledge provided by the discretization techniques in order to reduce the computational cost of the adaptive intervals representation. This process should be achieved without losing robustness. Another important point of further study is how the value of p_{split} and p_{merge} affect the behavior of the system, in order to simplify the tuning needed for each domain.

Finally, it would also be interesting to compare the adaptive intervals rule representation with some representation dealing directly with real-valued attributes, like the ones described in the related work section. This comparison should follow the same criteria used here: comparing both the accuracy and the computational cost.

Acknowledgments

The authors acknowledge the support provided under grant numbers 2001FI 00514, CICYT/Tel08-0408-02 and FIS00/0033-02. The results of this work were partially obtained using equipment cofunded by the *Direcció General de Recerca de la Generalitat de Catalunya (D.O.G.C. 30/12/1997)*. Finally we would like to thank Enginyeria i Arquitectura La Salle for their support to our research group.

References

1. Jaume Bacardit and Josep M. Garrell. Evolution of adaptive discretization intervals for a rule-based genetic learning system. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002) (to appear)*, 2002.
2. Jaume Bacardit and Josep M. Garrell. Métodos de generalización para sistemas clasificadores de Pittsburgh. In *Proceedings of the "Primer Congreso Iberoamericano de Algoritmos Evolutivos y Bioinspirados (AEB'02)"*, pages 486–493, 2002.
3. C. Blake, E. Keogh, and C. Merz. Uci repository of machine learning databases, 1998. Blake, C., Keogh, E., & Merz, C.J. (1998). UCI repository of machine learning databases (www.ics.uci.edu/mlearn/MLRepository.html).
4. A. L. Corcoran and S. Sen. Using real-valued genetic algorithms to evolve rule sets for classification. In *Proceedings of the IEEE Conference on Evolutionary Computation*, pages 120–124, 1994.
5. O. Cordón, M. del Jesus, and F. Herrera. Genetic learning of fuzzy rule-based classification systems co-operating with fuzzy reasoning methods. In *International Journal of Intelligent Systems, Vol. 13 (10/11)*, pages 1025–1053, 1998.

6. Kenneth A. DeJong and William M. Spears. Learning concept classification rules using genetic algorithms. *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 651–656, 1991.
7. Usama M. Fayyad and Keki B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *IJCAI*, pages 1022–1029, 1993.
8. David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Inc., 1989.
9. Elisabet Golobardes, Xavier Llorà, Josep Maria Garrell, David Vernet, and Jaume Bacardit. Genetic classifier system as a heuristic weighting method for a case-based classifier system. *Butlletí de l'Associació Catalana d'Intel.ligència Artificial*, 22:132–141, 2000.
10. John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.
11. John H. Holland. Escaping Brittleness: The possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems. In *Machine learning, an artificial intelligence approach. Volume II*, pages 593–623. 1986.
12. Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, pages 1137–1145, 1995.
13. Alexander V. Kozlov and Daphne Koller. Nonuniform dynamic discretization in hybrid networks. In *Proceedings of the 13th Annual Conference on Uncertainty in AI (UAI)*, pages 314–325, 1997.
14. H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, pages 388–391. IEEE Computer Society, 1995.
15. Xavier Llorà and Josep M. Garrell. Knowledge-independent data mining with fine-grained parallel evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 461–468. Morgan Kaufmann, 2001.
16. J. Martí, X. Cufí, J. Regincós, and et al. Shape-based feature selection for microcalcification evaluation. In *Imaging Conference on Image Processing, 3338:1215-1224*, 1998.
17. E. Martínez Marroquín, C. Vos, and et al. Morphological analysis of mammary biopsy images. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'96)*, pages 943–947, 1996.
18. José C. Riquelme and Jesús S. Aguilar. Codificación indexada de atributos continuos para algoritmos evolutivos en aprendizaje supervisado. In *Proceedings of the "Primer Congreso Iberoamericano de Algoritmos Evolutivos y Bioinspirados (AEB'02)"*, pages 161–167, 2002.
19. Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.
20. Stephen F. Smith. Flexible learning of problem solving heuristics through adaptive search. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence (IJCAI-83)*, pages 421–425, Los Altos, CA, 1983. Morgan Kaufmann.
21. Terence Soule and James A. Foster. Effects of code growth and parsimony pressure on populations in genetic programming. *Evolutionary Computation*, 6(4):293–309, Winter 1998.
22. Stewart W. Wilson. Get real! XCS with continuous-valued inputs. In L. Booker, Stephanie Forrest, M. Mitchell, and Rick L. Riolo, editors, *Festschrift in Honor of John H. Holland*, pages 111–121. Center for the Study of Complex Systems, 1999.
23. Ian H. Witten and Eibe Frank. *Data Mining: practical machine learning tools and techniques with java implementations*. Morgan Kaufmann, 2000.