Changing representation of learning examples while inducing classifiers based on decision rules

Jerzy Stefanowski
Institute of Computing Science, Poznań University of Technology,
ul. Piotrowo 3A, 60-965 Poznań, Poland
e-mail: Jerzy.Stefanowski@cs.put.poznan.pl

Abstract

Decision rules induced from examples are used to predict classification of new objects. Improving classification accuracy may be obtained by changing the original representation of the learning data. Two different approaches to such a transformation are considered in this paper: selecting the subset of the most relevant attributes with the wrapper approach, and modifying the presence of some learning examples in the learning set by the bagging technique. Both approaches are applied to the rule induction algorithm MODLEM and experimentally evaluated on several data sets.

Keywords: machine learning, rule induction, classification, attribute selection, multiple classifiers, bagging

1. Introduction

The aim of machine learning is to construct systems that automatically improve their performance with analysing experience represented by learning examples. In recent years many successful machine learning applications have been developed, in particular in domain of *data mining* and *knowledge discovery* [11, 18]. One of common tasks performed in knowledge discovery is *classification*. It consists of assigning a decision class label to a set of unclassified objects described by a fixed set of *attributes* (features).

Learning algorithms induce various forms of classification knowledge from learning examples, e.g. decision trees, rules, bayesian classifiers. In this paper we discuss a case, where knowledge is expressed in a form of *decision rules*. They are represented as logical expressions of the following form:

IF (*conditions*) THEN(*decision class*),

where conditions are formed as a conjunction of elementary tests on values of attributes. A number of various algorithms have already been developed to induce such rules (for a review see e.g. [15]). Decision rules are one of the most popular type of knowledge used in practice; one of the main reasons for their wide application is their expressive and easily human-readable representation, see e.g. discussions in [11, 18]. For reviews on applications of rules see e.g. [9] or some chapters in [11, 18]. The author of this paper has also took part in projects, where induction of rules was used for technical diagnostics of rolling bearings, reducers, engines of buses or for an analysis of medical data [15]. Moreover, an interesting study of applying inductive rule learning to mechanical engineering design was considered by Moczulski in [12].

When using decision rules for predicting classification, *classification* (predictive) *accuracy* plays a major role in evaluation of rules. It can be experimentally estimated by applying the rule set for testing examples and calculated as relative frequency of correct classifications [17]. The typical research problem while creating a new classification system is an attempt to improve classification accuracy. Although different rule based classifiers have been proved to be efficient for several learning problems, they may not led to satisfactory classification accuracy for other data sets. These difficulties may be caused, e.g., by existence of noisy examples or irrelevant attributes. Moreover, decision concepts may be too complex, non-linear and difficult to be learned by simple algorithms. One the approaches to improve classification performance is to change representation of the input set of learning examples.

As it is discussed in literature, one of the most important problem while constructing learning systems is to determine the appropriate representation space for learning, e.g. choosing attribute relevant to the problem at hand [10]. Specific transformation methods looking for "better" representation space, possibly integrated with learning algorithms, could improve the learning process.

In this paper, we will consider two different approaches to transform the representation space. These are:

1. changing the set of attributes describing examples,
2. changing the presence of some learning examples in the learning set, given a fixed set of originally predefined attributes.

In the first approach, we will check whether removing irrelevant attributes may lead to improving classification performance. We will use the method for selecting the most relevant attributes, which is based on the forward or backward stepwise search strategy applied inside, the so called, *wrapper model* [7].

In the second approach, the set of originally defined attributes remains unchanged but the presence of learning examples is modified. It is performed by a specific sampling from the original training data considered within the framework *multiple classifiers*. In this paper we focus attention on the *bagging* approach [2], which manipulates the input data to get several different learning sets by using sampling with replacement of learning examples. Then, different classifiers are generated from these learning sets and combined by a voting strategy to form a composite classifier.

The aim of this paper is to experimentally examine the applications of these two different approaches transforming an original representation of data to a rule induction algorithm MODLEM. This algorithm has been previously introduced by the author [14]. It is particularly well suited for

analysing data containing a mixture of numerical and qualitative attributes [15]. In this study we want to check how much the considered approaches could improve the classification accuracy of the rule classifier induced by MODLEM. According to our best knowledge this kind of rule classifier has not been considered yet together with these approaches. The bagging approach was mainly studied for decision trees, while the attribute selection mainly for instance based learning algorithms. The only exception is the previous preliminary author's study on using rule induction with the bagging [16], which gave encouraging results for the current study.

The paper is organised as follows. In Section 2, we begin with a presenting the attribute selection method. Then in Section 3, we describe modification of learning sets by the bagging approach. Section 4 contains a brief description of the MODLEM algorithm. Results of comparative experiments are given in Section 5. The discussion of these results and conclusions are presented in the final section.

2. Selection of attributes

For some learning problems, not all attributes describing examples may be directly relevant to classification and some attributes may be irrelevant or even redundant. If there are too many irrelevant attributes in the initial description of examples, the complexity of a learning process may increase. Moreover, they may decrease the classification performance of the induced classifier [7]. Thus, only the most relevant attributes should be used. The aim is to find the smallest subset of attributes leading to a higher classification accuracy than the set of all attributes.

As it is discussed in literature, the feature selection can be reduced to a search problem, where one selects a good feature subset based on a selected evaluation measure [7, 8]. Each state in the search space represents a subset of possible features. It is a partially ordered space, where each state has a child differing on exactly one feature. Following the idea presented by Kohavi [7], to design a feature selection algorithm one has to define three following components: *search algorithm* (technique that looks through the space of feature subsets), *evaluation function* (used to evaluate examined subsets of features), and *classifier* (a learning algorithm that uses the final subset of features). These elements can be integrated in two ways. They are called *filter* and *wrapper* approaches, respectively [7]. In the filter approach, features are selected as a pre-processing step before a classifier is used. Features are selected (i.e. filtered) basing on properties of data itself independently of the learning algorithm used as a classifier. In the wrapper approach, the search algorithm conducts a search for a good subset of features using the classifier itself as the evaluation function. An evaluation is usually done by estimating a classification accuracy obtained by this classifier.

Each filter and wrapper approach have its own strong and week points. Some authors claim [7] that the wrapper model is superior because it takes into account the bias of the learning algorithm in order to get a feature subset. It should result in a better estimate of an accuracy on unseen data than the evaluation function used in the filter model whose, bias differs from that of the classifier. However, the major limitation of the wrapper approach is the additional computational cost resulting from using many times the cross-validation technique evaluating learning algorithm to check each feature subset. On the other hand, the filter approach is less demanding. The typical evaluation measures, e.g. Info-gain entropy, Chi-squared statistics or correlation based measures, need less computations than the wrapper.

As to the wrapper approach one should remember that the classification accuracy for the final classifier induced using the selected subset of attributes should be evaluated on extra verification examples, which are not used in the learning phase to select attributes. It is necessary to avoid the phenomena of the overfitting the learning algorithm to the examples [8].

The next issue concerns constructing the search algorithm. Since the exhaustive search is of exponential complexity, it is more efficient to perform the heuristic search. Commonly employed algorithms are *backward elimination* and *forward selection*. Former starts with all attributes and successively removes the one that its elimination improves performance. The second starts with an empty set of attributes and successively adds the one with the best performance. There are also known more sophisticated variants of both approaches that combine performing either adding or removing operations as to get the best subset see, e.g.[6].

In our study we will use wrapper approach studying separately these two forward and backward search strategies.

3. Bagging approach to select learning examples

The bagging approach is one of the methods for creating multiple classifiers. According to this idea the same learning algorithm is runned several times, each time using a different distribution of the training examples. The generated classifiers are, then, combined to create a final classifier that is used to classify new objects.

The *Bagging* approach (**B**ootstrap **agg**regat**ing**) was introduced by Breiman [2]. It aggregates by voting classifiers generated from different bootstrap samples. The *bootstrap sample* is obtained by uniformly sampling objects from the training set with replacement. Each sample has the same size as the original set, however, some examples do not appear in it, while others may appear more than once. For a training set with m examples, the probability of an example being selected at least once is $1-(1-1/m)^m$. For a large m, this is about 1 - $1/e$. Each bootstrap sample contains, on the average, 63.2% unique examples from the training set.

Given the parameter T which is the number of repetitions, T bootstrap samples $S_1, S_2, \ldots, S_T$ are generated. From each sample S_i a classifier C_i is induced by the same algorithm and the final classifier C^* is formed by aggregating T classifiers. A final classification of object x is built by a uniform voting scheme on $C_1, C_2, \ldots, C_T$, i.e. is assigned to the class predicted most often by these sub-classifiers, with ties broken arbitrarily. The approach is presented briefly below. For more details see [2].

(**input** LS learning set; T number of bootstrap samples;
LA learning algorithm
output C^* classifier)
begin
  for $i = 1$ **to** T **do**
  begin
    S_i := bootstrap sample from LS; {sample with replacement}
    $C_i := LA(S_i)$; { generate a sub-classifier }
  end; { end for }
  $C^*(x) = \arg\max_{y \in K_j} \sum_{i=1}^{T} (C_i(x) = y)$
    {the most often predicted class}
end

Experimental results presented in [2, 3] show a significant improvement of classification accuracy while using decision tree classifiers. For more theoretical discussion on the justification of bagging the reader is referred to [2].

4. Rule induction by the MODLEM algorithm

The MODLEM algorithm has been introduced in [14], see also [15]. It generates heuristicaly a minimal set of rules describing succeeding decision classes (or their rough approximations). Its extra specificity is handling directly numerical attributes during rule induction when elementary conditions of rules are created, without any preliminary discretization phase. Below we present its basic concept. First we need to introduce necessary notation. Learning examples are described by a set of *condition attributes* A, where V_a is a domain of $a \in A$ and $a(x)$ denotes the value of attribute $a \in A$ taken by an example x; $d \notin A$ is a decision attribute that partitions examples into a set of decision classes (concepts) $\{K_j : j = 1, \ldots, k\}$.

A decision rule r describing class K_j is represented in the following form: *if* P *then* Q, where $P = w_1 \wedge w_2 \wedge \ldots w_p$ is a condition part of the rule and Q is decision part of the rule indicating that example satisfying P should be assigned to class K_j. The elementary condition of the rule r is defined as $(a_i(x)\ rel\ v_{a_i})$, where rel is a relational operator from the set $\{=, <, \leq, >, \geq\}$ and v_{a_i} is a constant being a value of attribute a_i. $[P]$ is a *cover* of the condition part of rule r in DT, i.e. it is a set of examples, which description satisfy elementary conditions in P. Let B be a set of examples belonging to class K_j. For a certain decision rule r we require that $[P] = \bigcap [w_i] \subseteq B$. The general schema of the MODLEM algorithm is given below. It is iteratively repeated for each set of examples B from a succeeding decision class K_j.

Procedure MODLEM (**input** B set of examples; *criterion* - evaluation measure; **output P** single local covering of B)
begin
  $G := B$; {examples not covered by conjunction from **P**}
  P $:= \emptyset$;
  while $G \neq \emptyset$ **do begin**
    $P := \emptyset$; {candidate for condition part of the rule}
    $S := U$; {set of objects covered by P}
    while $(P = \emptyset)$ or (not($[P] \subseteq B$)) **do begin**
      $w := \emptyset$; {candidate for elementary condition}
      for each attribute $a \in C$ **do begin**
        $new\_p := Find\_best\_condition(a, S)$;
        if $Better(new\_p, w, criterion)$ **then** $w := new\_p$;
        { evaluate if new condition new_p is better than previous w}
      end;
      $P := P \cup \{w\}$; {add to the condition part }
      $S := S \cap [w]$;
    end; { while not($[P] \subseteq B$)}
    for each elementary condition $w \in P$ **do**
      if $[P - w] \subseteq B$ then $P := P - \{w\}$;
      { Test minimality of the rule }
    P $:=$ **P** $\cup \{P\}$; { Add P to the local covering }
    $G := B - \bigcup_{P \in \mathbf{P}} [P]$;
    { Remove examples covered by the induced rule}
  end; { while $G \neq \emptyset$ }
  for each $P \in$ **P do** **if** $\bigcup_{P' \in \mathbf{P} - P} [P'] = B$ **then** **P** $:=$ **P** $- P$
end{procedure}

Table 1. Data sets used in the experiments

Data set	Number of examples	Number of classes	Number of attributes
bank	66	2	5
buses	76	2	8
zoo	101	7	8
hsv	122	4	11
iris	150	3	4
hepat	147	2	19
glass	214	6	9
bricks	216	2	10
vote	300	5	16
bupa	345	2	6
pima	768	2	8

Let us comment how function *Find_best_condition* works – for more details see [5, 14]. Elementary conditions are represented as either $(a(x) < v_a)$ or $(a(x) \geq v_a)$. For nominal attributes, these conditions are $(a(x) = v_a)$. We will shortly present how best conditions are chosen for numerical attributes. First, for a given set of objects their attribute values are sorted in an increasing order. The candidates for the cut-point v_a are computed as mid-points between successive values in the sorted order, taking into account decision class assignment of objects. They are evaluated according to a chosen evaluation measure either *class entropy* or *Laplace accuracy* - in the experiment we used the entropy. The best point among all tested ones (function *Better*) is chosen to be further compared against other attributes. The best condition w for all compared attributes is chosen to be added to the condition part of the rule.

Finally, the unordered set of induced rules is applied to classify examples using the classification strategy introduced by Grzymala in LERS system [4].

5. Experiments

5.1. Conditions of experiments

The aim of the experimental study is to check how much two different techniques of modifying the representations of learning data, discussed in this paper, could increase classification accuracy of the rule classifier induced by MODLEM algorithm. More precisely, we compare performance of:

1. The single classifier obtained by MODLEM used for all attributes (without any selection).

2. The single classifier induced for selected subsets of attributes and all learning examples. Two search strategies inside wrapper model are independently studied: forward selection and backward elimination (denotated FS and BE, repectively).

3. The bagging classifier composed of single classifiers induced by MODLEM with the complete set of features.

The MODLEM algorithm has been used with the entropy measure to choose elementary conditions. Moreover, if there are any inconsistencies in the learning sets, then the rough approximations were calculated and MODLEM produced only certain decision rules.

Table 2. Comparison of classification accuracies [%] for classifiers induced by using all attributes and attribute subsets obtained by Forward Selection and Backward Elimination search strategies

Dataset	All attributes	FS	BE
bank	93.81 ± 0.94	95.46 ± 1.1	92.15* ± 1.14
buses	97.20 ± 0.94	98.05* ± 1.15	97.38* ± 1.12
zoo	94.64 ± 0.67	95.01* ± 0.59	93.71* ± 0.59
hsv	54.52 ± 1.05	65.94 ± 0.69	58.41 ± 1.28
hepatitis	78.62 ± 0.93	83.91 ± 0.49	80.57 ± 0.79
iris	94.93 ± 0.5	94.67* ± 0.58	94.93* ± 0.5
glass	72.41 ± 1.23	71.42* ± 1.1	73.69* ± 1.04
bricks	90.32 ± 0.82	89.82* ± 0.37	89.89* ± 0.5
vote	92.67 ± 0.38	88.67 ± 0.82	93.91 ± 0.48
bupa	65.77 ± 0.6	62.28 ± 0.79	65.15* ± 0.55
pima	73.57 ± 0.67	74.92 ± 0.47	75.82 ± 0.51

Table 3. Comparison of classification accuracies [%] obtained by the single MODLEM classifier and the bagging approach

Dataset	Single classifier	Bagging
bank	93.81 ± 0.94	95.22 ±1.02
buses	97.20 ± 0.94	97.45* ± 1.13
zoo	94.64 ± 0.67	93.68 ± 0.70
hsv	54.52 ± 1.05	65.94 ± 0.69
hepatitis	78.62 ± 0.93	84.0 ± 0.49
iris	94.93 ± 0.5	94.33* ± 0.59
glass	72.41 ± 1.23	76.09 ± 0.68
bricks	90.32* ± 0.82	90.77* ± 0.72
vote	92.67 ± 0.38	96.01 ± 0.29
bupa	65.77 ± 0.6	75.69 ± 0.7
pima	73.57 ± 0.67	77.87 ± 0.39

All experiments have been performed on the benchmark data sets, which are coming from Machine Learning Repository at the University of California at Irvine [1] or from author's case studies (data *buses, hsv, bricks*, see [15]). Their characteristics is given in Table 1. The classification accuracy was estimated by stratified version of 10-fold cross-validation technique, i.e. the training examples were partitioned into 10 equal-sized blocks with similar class distributions as in the original set.

5.2. Attribute selection

For all data sets we used the wrapper approach with two search strategies forward selection (FS) or backward elimination (BE) . The obtained classification accuracies are presented in Table 2. They are calculated as average classification accuracy with standard deviation. An asterisk indicates that differences for compared classifiers and the given data set are not statistically significant.

Let us remind that the classifier obtained for a selected attribute subset should be evaluated on the verification examples, which are not used inside the wrapper evaluation. Therefore, we employed two level "k-fold cross validation" technique. First, in the outside cross-validation, the examples are randomly divided into the learning and verification sets. Then, for the wrapper search strategy, the other "inner"10-fold cross validation is used only in the learning set to find the selected subset. In our experiments for both cross validations, k has been equal to 10.

We have also analysed the structure of attribute subsets selected by both methods FS and BE. We have noticed that these subsets were not the same for any data set, except *glass* and *bupa*. Usually, the forward strategy selected much smaller number of attributes than the backward elimination.

5.3. Bagging and modification of the learning set

While creating the multiple classifier, the parameter T being the number of sub-classifiers inside bagging was set at the following values: 3, 5, 7 and 10. Choosing these small values of T was inspired by good results obtained by Quinlan for studying C4.5 decision trees with the bagging [13].

The results of these experiments are given in Table 3. For each data set, the first column shows the average classification accuracy obtained by a single classifier over the ten cross-validations. Standard deviation is also given. The next column contains results for composite bagging classifier. An asterisk indicates that differences for compared classifiers and given data sets are not statistically significant. For nearly all data set we present results obtained for T=10 which was the best value, except: *bank* T=7, *hsv* T=5.

6. Discussion of results and final remarks

Firstly, let us discuss results of the comparative experiments performed on 11 data sets.

The rule classifier built with attributes determined by the forward selection strategy was better than the classifier using all attributes on 4 data sets (*bank, hsv, hepatitis, pima*) and worse on 2 data sets (*vote, bupa*). The difference between both classifiers was not significant on the remaining 5 data sets.

Using backward elimination search strategy, an improved classification accuracy was observed on 4 data sets (*hsv, hepatitis, vote, pima*). For other data sets the difference between classifiers were not significant.

Comparing the number of attributes selected by each search strategy, we noticed that the forward selection usually identified smaller subsets than the backward elimination. Moreover, the time of computations for the forward selection was usually shorter.

The other approach to modify the number of examples, i.e. bagging, outperformed the single classifier on 7 of 11 data sets. The difference between classifiers was non-significant on 3 data sets (*buses, iris, bricks*) and the single classifier was better only for 1 data sets *zoo*. In fact, the slightly worse performance the bagging was observed for quite small data sets (e.g. *buses* - which could be difficult for representative sampling) or data sets rather easy to be learned by standard single classifier (as e.g. *iris*). Bagging substantially improves the classification accuracy for data sets contating higher number of examples. Let us notice, that for some of these data bagging allowed to achieve the highest improvement of classification (see e.g. *glass, vote, bupa* or *pima*).

Comparing the experimental results between the two approaches to change the data representations, we can conclude that the bagging approach is more efficient for improving classification accuracy. It could be, somehow, explained by the specific properties of the used rule induction. In general, the

attribute selection should be useful for such learning algorithms, which are particularly sensitive to the presence of irrelevant attributes [8]. For instance, this could be the k–nearest neighbor algorithm, where all attributes influence the way of calculating the distance or similarity between examples. We have examined it in our previous studies [6]. However, the rule induction algorithm chooses the best elementary condition (referring to attributes) just inside a rule generation. Quite often, only part of the original attributes may be used in the finally induced rules. Therefore, this kind of learning algorithm has a kind of "internal ability" to select the most relevant attributes. Although the usefulness of an additional attributes selection in the wrapper approach is limited, we observed an improvement for a few data sets.

On the other hand, the main aim of introducing the bagging approach was just to improve the classification performance [2]. Our experimental results have confirmed it also for using of the MODLEM algorithm. Let us remind that according to some authors [2, 3] the bagging should work especially well for, the so called, unstable learning algorithms, i.e. ones whose output classifier undergoes major changes in response to small changes in learning data. Indeed the algorithm MODLEM is the unstable algorithm in the sense of this postulate.

Finally, let us comment on the computational costs. As the bagging approach needs the number T of sub-classiers, it requires around T more computational efforts than single learning algorithm. While costs of the wrapper depend on the number of attributes subsets to be examined during the search process. Moroever, for each examined subset it is necessary to perform the cross-validation evaluation. Therefore, it is more demanding approach than the bagging.

However, there is another disadvantage of the bagging approach - loosing a simple and easy interpretable structure of knowledge represented in a form decision rules. While, the attribute selection maintains a typical form of the rule set. For future research, it could be interesting to consider yet another way of improving the original representation of attributes - i.e. generating new attributes, which may be function of the original ones. It is consistent with the postulate of the constructive induction [10].

Acknowledgment: The research has been supported from the BW grant.

References

[1] C. Blake, E .Koegh , C.J. Mertz , Repository of Machine Learning, University of California at Irvine 1999 [URL: http://www.ics.uci.edu/ mlearn/MLRepositoru.html].

[2] L. Breiman, Bagging predictors. *Machine Learning*, **24** (2), (1996), 123–140

[3] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning* **36** (1/2), (1999) , 105–139.

[4] J.W. Grzymala-Busse, Managing uncertainty in machine learning from examples. In: *Proc. 3rd Int. Symp. in Intelligent Systems*, Wigry, Poland, IPI PAN Press, 1994, 70–84.

[5] J.W. Grzymala-Busse, J. Stefanowski, Three approaches to numerical attribute discretization for rule induction. *International Journal of Intelligent Systems*, **16** (1), (2001), 29–38.

[6] K. Krawiec , J. Jelonek, J. Stefanowski, Comparative study of feature subset selection techniques for machine learning tasks. In *Proceedings of VIIth Intelligent Information Systems* IIS'98 Malbork, 15-19 June 1998, IPI PAN Press Warszawa 1998, 68-77.

[7] G. John, R. Kohavi, K. Pfleger, Irrelevant features and the subset selection problem. *Proceedings of the Eleventh International Machine Learning Conference*, New Brunswick NJ, Morgan Kaufmann, 1994, 121-129.

[8] R. Kohavi , D. Sommerfield , Feature Subset Selection Using the Wrapper Method: Overfitting and Dynamic Search Space Topology. *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, Montreal, AAAI Press, 1995, 192-197.

[9] P. Langley, H.A. Simon, Fielded applications of machine learning, In: R.S. Michalski, I. Bratko, M. Kubat (eds.), *Machine learning and data mining*, John Wiley & Sons, 1998, 113-129.

[10] R.S. Michalski, A theory and methodology of inductive learning. In: R.S. Michalski, J.G. Carbonell and T.M. Mitchell, (eds.), *Machine Learning: An Artificial Intelligence Approach*, vol. 1, Morgan Kaufman, 1983 83–134.

[11] R.S. Michalski, I. Bratko, M. Kubat (eds.), *Machine learning and data mining*, John Wiley & Sons, 1998.

[12] W. Moczulski, Inductive learning in design: A method and case study concerning design of antifriction bearing systems. In: R.S. Michalski, I. Bratko, M. Kubat (eds.), *Machine learning and data mining*, John Wiley & Sons, 1998, 203-220.

[13] J.R. Quinlan, Bagging, boosting and C4.5. In: *Proceedings of the 13th National Conference on Artificial Intelligence*, 1996, 725–730.

[14] J. Stefanowski, The rough set based rule induction technique for classification problems. In: *Proceedings of 6th European Conference on Intelligent Techniques and Soft Computing* EUFIT'98, Aaachen 7-10 Sept. 1998, 109-113.

[15] J. Stefanowski, *Algorithims of rule induction for knowledge discovery.* (In Polish), Habilitation Thesis published as Series Rozprawy no. 361, Poznan Univeristy of Technology Press, Poznan, 2001.

[16] J. Stefanowski, Bagging and induction of decision rules. In: Klopotek M., Wierzchon S. Michalewicz M. (eds.), *Int. Symposium on Intelligent Systems*; Post-Proceedings of the IIS'2002. Series "Advances of Soft Computing", Physica Verlag, Heidelberg, 2002, 121-130.

[17] S.M. Weiss, C.A. Kulikowski, *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems*, Morgan Kaufmann, San Francisco, 1991.

[18] W. Klosgen, J.M. Żytkow, *Handbook of Data Mining and Knowledge Discovery.* Oxford Press, 2002.