Fundamenta Informaticae XXI (2001) 1001–1016 1001

IOS Press

Learning Rule Ensembles for Ordinal Classification with Monotonicity Constraints

Krzysztof Dembczyński

Institute of Computing Science, Poznań University of Technology, 60-965 Poznań, Poland

kdembczynski@cs.put.poznan.pl

Wojciech Kotłowski

Institute of Computing Science, Poznań University of Technology, 60-965 Poznań, Poland

wkotlowski@cs.put.poznan.pl

Roman Słowiński

Institute of Computing Science, Poznań University of Technology, 60-965 Poznań, Poland

Systems Research Institute, Polish Academy of Sciences, 01-447 Warsaw, Poland

rslowinski@cs.put.poznan.pl

Abstract. Ordinal classification problems with monotonicity constraints (also referred to as multi-criteria classification problems) often appear in real-life applications, however, they are considered relatively less frequently in theoretical studies than regular classification problems. We introduce a rule induction algorithm based on the statistical learning approach that is tailored for this type of problems. The algorithm first monotonizes the dataset (excludes strongly inconsistent objects), using Stochastic Dominance-based Rough Set Approach, and then uses forward stagewise additive modeling framework for generating a monotone rule ensemble. Experimental results indicate that taking into account knowledge about order and monotonicity constraints in the classifier can improve the prediction accuracy.

Keywords: ordinal classification, monotonicity constraints, rule ensembles, forward stagewise additive modeling, boosting, dominance-based rough set approach.

Address for correspondence: Wojciech Kotłowski, Institute of Computing Science, Poznań University of Technology, 60-965 Poznań, Poland

1. Introduction

In the problem of *ordinal classification with monotonicity constraints* the purpose is to predict for a given object one of K ordered class labels (*ranks*). Objects are described by attributes with ordered value sets and *monotonicity constraints* are present in the data: a higher value of an object on an attribute, with other values being fixed, should not decrease its class assignment. Depending on interpretation of attributes, also inverse monotonicity constraints may be relevant: a lower value of an object on an attribute with other attributes being fixed, should not decrease the class assignment. In this paper, we assume, without loss of generality, that only the first kind of monotonicity constraints holds.

As an example, consider the problem of house pricing, i.e., classification of houses with respect to their prices, into one of the following classes: "cheap", "moderate", "expensive", "very expensive". The classification is based on the following attributes: lot size, number of bedrooms, garages, whether a house contains air conditioning, basement, etc. [29]. It is obvious that the price of house A should not be less than that of house B if, for instance, house A has greater number of bedrooms and garages than B, and opposite to B, has a basement, and, moreover, house A is as good as B on other attributes.

Problems of ordinal classification with monotonicity constraints are commonly encountered in real-life applications. A typical representative is multiple-criteria classification (or sorting) considered within multiple-criteria decision analysis (MCDA) [39, 24]. Moreover, in many other domains, ordinal and monotone properties follow from the domain knowledge about the problem and should not be neglected. They are encountered in such problems as bankruptcy risk prediction [43, 21, 41], breast cancer diagnosis [40], house pricing [36], Internet content filtering [28], credit rating [14], liver disorder diagnosis [42], credit approval [17], surveys data [6] and many others.

In order to solve ordinal classification problem with monotonicity constraints, one can apply two steps for improving the accuracy of the classifier. The first one consists in "monotonization" of the dataset, i.e., exclusion of objects strongly violating the monotone relationships in order to make the dataset monotone. The second one consists in imposing the constraints such that only monotone classification functions are taken into account.

Dominance-based Rough Set Approach (DRSA) [22, 23, 24] is one of the first approaches introduced to deal with this type of problems. Replacing indiscernibility relation, considered in classical rough sets [35], by a dominance relation, DRSA is able to handle inconsistencies following from violation of monotone relationships. In this context, several specialized decision rule induction algorithms were proposed that were able to capture the ordinal nature of data and handle domain knowledge in the form of monotonicity constraints [26, 13] (we will refer to rules consistent with monotonicity constraints as *monotone* rules). Among them, DOMLEM [25] seems to be the most popular one. It aims at finding a minimal set of monotone rules covering the dataset, using the well-known sequential covering procedure as a search heuristic.

In this paper, we follow a different methodology for monotone rule induction that is based on *forward stagewise additive modeling* (FSAM) [19], i.e., greedy procedure for minimizing a loss function on the dataset. The algorithm presented in this paper, called MORE (from MOnotone Rule Ensembles), treats a single rule as a subsidiary base classifier in the ensemble. The rules are added to the ensemble iteratively, one by one. Each rule is fitted by concentrating on the examples which were hardest to classify correctly by rules that have already been generated. The advantage of this approach is that we use a single measure only (value of the empirical risk) at all stages of the learning procedure: setting the best cuts (conditions), stopping the rule's growth and determining the weight of the rule; no additional features (e.g., impurity

measures, pruning procedures) are considered. Such an approach was already considered in ordinary classification problems and algorithms such as RuleFit [20], SLIPPER [7], LRI [45], MLRules [12] or ENDER [11] exist. MORE can be seen as an extension of the last one from among the methods mentioned above. It monotonizes the dataset (excludes strongly inconsistent objects) using Stochastic DRSA [30, 9] and then generates monotone rules.

The main idea of MORE has been introduced in [10]. In this paper, we extend the algorithm, give formal properties of this approach and present results of an extensive computational experiment.

2. Problem Statement

In the *classification* problem, the aim is to predict the unknown class label $y \in Y = \{1, \ldots, K\}$ (decision value) of an object $\mathbf{x}$ using the description of the object in terms of p (condition) attributes, $\mathbf{x} = (x_1, x_2, \ldots, x_p) \in X$, where X is the *attribute space*. Here, we assume without loss of generality that value set of each attribute is a subset of $\mathbb{R}$, so that $X \subseteq \mathbb{R}^p$. In the *ordinal classification*, it is assumed that there is a meaningful order between classes which corresponds to the natural order between class labels. We also assume the presence of *monotonicity constraints* in the data.

In order to formalize the concept of monotonicity, we define the *dominance* relation as a binary relation on X in the following way: for any $\mathbf{x}, \mathbf{x}' \in X$ we say that $\mathbf{x}$ *dominates* $\mathbf{x}'$, denoted $\mathbf{x} \succeq \mathbf{x}'$, if on every attribute, $\mathbf{x}$ has value not smaller than $\mathbf{x}'$, $x_j \geq x'_j$, for all $j = 1, \ldots, p$. The dominance relation is a partial pre-order on X, i.e., it is reflexive and transitive. Having defined the dominance relation, we define the *monotone function* to be any function $h \colon X \to Y$ satisfying the monotonicity constraints:

$$\mathbf{x} \succeq \mathbf{x}' \to h(\mathbf{x}) \geq h(\mathbf{x}'), \tag{1}$$

for any $\mathbf{x}, \mathbf{x}' \in X$.

Now, the problem of ordinal classification with monotonicity constraints can be stated as a problem of finding the *monotone classification function* $h(\mathbf{x})$ that predicts accurately values of y. The accuracy is measured in terms of the *loss function* $L(y, h(\mathbf{x}))$, which is the penalty for predicting $h(\mathbf{x})$ when the actual value is y. The overall accuracy of function $h(\mathbf{x})$ is defined as the expected loss (*risk*) according to the probability distribution of data to be predicted:

$$R(h) = E[L(y, h(\mathbf{x}))]. \tag{2}$$

A *Bayes classification function* is a function h^* minimizing the risk (2). We assume that h^* is monotone, which justifies restricting to the class of monotone classification functions. Since $P(x, y)$ is unknown, h^* is unknown and the classification function is learned from a set of n training examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ (*training set*). In order to minimize the value of risk (2), the learning procedure usually performs minimization of the *empirical risk*:

$$R_{\text{emp}}(h) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, h(\mathbf{x}_i)), \tag{3}$$

which is the value of a loss function on the training set (i.e., training error). It is possible to use a variety of loss functions for measuring accuracy; here, we assume the loss function to be the *absolute error loss*,

$$L_{\text{abs}}(y, h(\mathbf{x})) = |y - h(\mathbf{x})|. \tag{4}$$

Notice that the proposed model shares the spirit of [31, 32] while being different from the previous rank loss formulation found in [27].

Although in classification, a 0-1 loss is often considered (defined as $L_{0-1}(y, h(\mathbf{x})) = 1$ if $y \neq h(\mathbf{x})$, 0 otherwise), absolute error loss has the advantage over 0-1 loss of being sensitive to the difference between predicted and actual class labels, therefore, taking the order between classes into account. Moreover, absolute error loss has another very important property: its minimizer (Bayes classification function) does not depend on a particular encoding of class labels. Indeed, it is known [4], that for a given $\mathbf{x}$, the absolute error loss is minimized by the median of the conditional distribution $P(y|\mathbf{x})$. The median in the ordered set does not depend on the particular values of the elements in the set and takes only the order into account.

Thus, although it seems that absolute error imposes a distance measure between the class labels (equal to the difference between their indices), its minimization is invariant under arbitrary monotone (order-preserving) transformation of the labels, so it is actually scale-free.

3. Decomposition into Binary Subproblems

One can solve the ordinal classification problem with loss function (4) by reducing the problem to $K-1$ binary problems. Let us define for a given class label y, $K-1$ auxiliary class labels y_k equal to 1 if $y \geq k$, otherwise -1, for each $k = 2, \ldots, K$, i.e. $y_k = \text{sgn}\,(y - k)$.[1] Therefore, $y_k = 1$ corresponds to the class union "at least k", while $y_k = -1$ corresponds to the class union "at most $k-1$". We also say that $y_k = 1$ indicates a "positive" class, and $y_k = -1$ indicates a "negative" class.

Suppose we have an access to the learning algorithm for binary classification problems, which, by using auxiliary labels y_k, can produce classifier $h(\mathbf{x}) \in \{-1, 1\}$ with small absolute error loss (or 0-1 loss, as they are equivalent in the binary case). Moreover, let us assume that each binary classifier has the form $h_k(\mathbf{x}) = \text{sgn}\,(f_k(\mathbf{x}))$, where $f_k(\mathbf{x})$ is a real-valued function such that the magnitude $|f_k(\mathbf{x})|$ corresponds to the confidence of prediction – the higher the magnitude, the more we are certain about predicting the class $\text{sgn}\,(f_k(\mathbf{x}))$. Since $f_k(\mathbf{x})$ is an increasing function of the confidence of classification to the class union $\{k, \ldots, K\}$, then it should hold:

$$f_{k+1}(\mathbf{x}) \leq f_k(\mathbf{x}), \tag{5}$$

because we are always more certain about classifying an object to the set $\{k, \ldots, K\}$ than to its subset $\{k+1, \ldots, K\}$. If (5) holds, then the final classification procedure combining $K-1$ binary classifiers into a single K-class ordinal classifier $h(\mathbf{x})$ is straightforward: we seek for the first k in the sequence of $k = 2, \ldots, K$, where $f_k(\mathbf{x})$ changes the sign, and we classify $\mathbf{x}$ to the class labeled by this k. This is equivalent to comprehensively writing $h(\mathbf{x}) = 1 + \sum_{k=2}^{K} 1_{f_k(\mathbf{x}) \geq 0}$.[2]

However, binary problems are solved independently, so we cannot guarantee that such constraints hold in each case. We deal with violation of constraints (5) using the *isotonic regression* in the following way. Fix $\mathbf{x}$ and notice that from (5) it follows that $f_k(\mathbf{x})$ must be a monotonically decreasing function of k. If $f_k(\mathbf{x})$ is not monotonically decreasing, we search for another function $g_k(\mathbf{x})$ which is monotonically

[1] We define the *sign function* as $\text{sgn}\,(x) = 1$ if $x \geq 0$, and -1 otherwise.

[2] We define the *indicator function* as $1_A = 1$ if predicate A holds, and 0 otherwise.

decreasing and is as close as possible to $f_k(\mathbf{x})$ in the sense of squared error:

$$\min \sum_{k=2}^{K} (f_k(\mathbf{x}) - g_k(\mathbf{x}))^2.$$

This is the problem of *isotonic regression* [5, 38]. It can be thought of as "monotonizing" function $f_k(\mathbf{x})$ which, possibly, violates constraints (5).

What is surprising, we do not even need to solve the isotonic regression. Let us consider the following algorithm of combining $K-1$ classifiers $f_k(\mathbf{x})$, $k = 2, \ldots, K$, to obtain a class label $h(\mathbf{x}) \in \{1, \ldots, K\}$. The algorithm calculates votes for each class and the class with the highest vote is chosen as the prediction $h(\mathbf{x})$. Let us denote the vote for class k as $vote_k$. The vote is calculated in the following way:

$$vote_k(\mathbf{x}) = \sum_{l=2}^{k} f_l(\mathbf{x}). \tag{6}$$

Note, that according to (6) $vote_1(\mathbf{x}) = 0$. The following theorem holds:

Theorem 3.1. Consider the classifier:

$$b(\mathbf{x}) = 1 + \sum_{k=2}^{K} 1_{g_k(\mathbf{x}) \geq 0},$$

where for each $k = 2, \ldots, K$, $g_k(\mathbf{x})$ is the isotonic regression of $f_k(\mathbf{x})$. Let $vote_k = \sum_{l=2}^{k} f_l(\mathbf{x})$ and let us define the classifier $h(\mathbf{x}) = \arg\max_k vote_k$ (in case of ties, we choose the highest label). Then, $h(\mathbf{x}) = b(\mathbf{x})$.

Proof:
Let us fix $\mathbf{x}$; we will omit the dependency on $\mathbf{x}$ and write f_k, g_k, h, b, etc. Notice that h is such that for any $k > h$ it holds $vote_h > vote_k$, while for any $k \leq h$ it holds $vote_h \geq vote_k$. To show that $b = h$, it is enough to show that if $k > b$ then $vote_b > vote_k$, and if $k \leq b$ then $vote_b \geq vote_k$.

Let us define $G_+ = \{k\colon g_k \geq 0\}$ and similarly $G_- = \{k\colon g_k < 0\}$. Notice that $G_+ = \{2, \ldots, b\}$ and $G_- = \{b+1, \ldots, K\}$. Moreover, let us also denote $f(A) = \sum_{k \in A} f_k$. We will use Theorem 1.4.3 from [16], which states that for every $k = 2, \ldots, K$ it holds:

$$f(\{2, \ldots, k\} \cap G_-) < 0, \qquad f(\{k+1, \ldots, K\} \cap G_+) \geq 0.$$

Suppose $k \leq b$. Then:

$$vote_b - vote_k = \sum_{l=k+1}^{b} f_l = f(\{2, \ldots, b\} \cap \{k+1, \ldots, K\}) = f(\{k+1, \ldots, K\} \cap G_+) \geq 0,$$

so that $vote_b \geq vote_k$. Similarly, if $k > b$ then:

$$vote_k - vote_b = \sum_{l=b+1}^{k} f_l = f(\{2, \ldots, k\} \cap \{b+1, \ldots, K\}) = f(\{2, \ldots, k\} \cap G_-) < 0,$$

so that $vote_b > vote_k$, which ends the proof. □

Thus, we end up with a simple procedure of combining binary classifiers by summing their votes for each class union and predicting the class with the highest vote.

We would like to have the monotonicity property of the classifier created according to Theorem 3.1, i.e., if $\mathbf{x} \succeq \mathbf{x}'$ then $h(\mathbf{x}) \geq h(\mathbf{x}')$. The following theorem gives sufficient conditions for monotonicity:

Theorem 3.2. For each $k = 2, \ldots, K$, let $f_k(\mathbf{x})$ be a monotone function, i.e.: $\mathbf{x} \succeq \mathbf{x}' \rightarrow f_k(\mathbf{x}) \geq f_k(\mathbf{x}')$. Then, the classifier $h(\mathbf{x})$, obtained by choosing the class label with the highest vote (6), is a monotone function.

Proof:
Choose any $r, s \in \{1, \ldots, K\}$ such that $r \geq s$ and let $\mathbf{x} \succeq \mathbf{x}'$. Then:

$$vote_r(\mathbf{x}) - vote_s(\mathbf{x}) = \sum_{k=s+1}^{r} f_k(\mathbf{x}) \geq \sum_{k=s+1}^{r} f_k(\mathbf{x}') = vote_r(\mathbf{x}') - vote_s(\mathbf{x}'). \tag{7}$$

In other words, the difference in votes for any two classes r and s (where $r \geq s$) is a monotone function, according to definition (1).

Now, suppose the contrary, that $h(\mathbf{x}) < h(\mathbf{x}')$. From the definition of $h(\mathbf{x})$, we have that for all $k > h(\mathbf{x})$, $vote_{h(\mathbf{x})}(\mathbf{x}) > vote_k(\mathbf{x})$ (since $h(\mathbf{x})$ is the highest label with the greatest vote). In particular, it holds for $k = h(\mathbf{x}')$, so $vote_{h(\mathbf{x}')}(\mathbf{x}) - vote_{h(\mathbf{x})}(\mathbf{x}) < 0$. But then, using (7), we also have $vote_{h(\mathbf{x}')}(\mathbf{x}') - vote_{h(\mathbf{x})}(\mathbf{x}') < 0$, which means, that $h(\mathbf{x}')$ is not the label with the highest vote for $\mathbf{x}'$, a contradiction. □

Thus, if the binary functions $f_k(\mathbf{x})$ are all monotone, then so is $h(\mathbf{x})$.

4. Data Monotonization

From the monotonicity assumption the *dominance principle* follows: for any two objects $\mathbf{x}_i, \mathbf{x}_j$ from the dataset, such that $\mathbf{x}_i \succeq \mathbf{x}_j$, it should hold $y_i \geq y_j$. However, it still may happen that in the dataset there exists an object $\mathbf{x}_i$, dominating another object $\mathbf{x}_j$, while it holds $y_i < y_j$. This is possible since the monotonicity constraints hold only in the probabilistic sense due to inherent uncertainty of the data generation stochastic process. Such a situation violates the monotonicity assumption, so we shall call objects $\mathbf{x}_i$ and $\mathbf{x}_j$ *inconsistent*. Notice that no monotone function can approximate accurately inconsistent objects. Therefore, stochastic extension of DRSA [30, 9] is applied in order to monotonize the data.

Stochastic DRSA estimates the conditional probabilities $P(y \geq k|\mathbf{x}_i)$, or equivalently $P(y_k = 1|\mathbf{x}_i)$. This is done in a nonparametric way, using the multiple isotonic regression estimator [30]. Let us consider the k-th binary problem. All the objects for which $P(y_k = 1|\mathbf{x}_i) \geq \alpha$ form the so called *lower approximation* of class $y_k = 1$, while all the objects for which $P(y_k = -1|\mathbf{x}_i) \geq \alpha$ – the lower approximation of class $y_k = -1$. The rest of the objects is discarded.

The idea of DRSA is to use in the k-th binary problem lower approximation of positive ($y_k = 1$) and negative ($y_k = -1$) classes, instead of the original classes. This requires that inconsistent objects are effectively relabeled. Indeed, suppose the label of object $\mathbf{x}_i$ in the k-th binary problem is $y_{ik} = -1$. Then, if such an object enters lower approximation of class $y_k = 1$, then we can think of it as giving to the object a new label $y^*_{ik} = 1$.

It follows from the properties of isotonic regression that probability $P(y_k = 1|\mathbf{x})$ is a monotone function of $\mathbf{x}$. Therefore, the dataset with new labels y_k^* will be always consistent. From the computational point of view, it can be shown that for $\alpha = 0.5$ the new labels (assignments to the lower approximations) are obtained from the solution of the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} 1_{y_{ik} \neq y_{ik}^*} \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \rightarrow y_{ik}^* \geq y_{jk}^* \qquad i,j = 1,\ldots,n, \\ & y_{ik}^* \in \{-1,1\} \qquad i = 1,\ldots,n. \end{aligned}$$

Although the variables are integer, the problem can be effectively solved as a simple linear program [30]. Note that this is the problem of reassignment – the objective is to reassign the smallest number of objects in order to make the dataset monotone. That is why Stochastic DRSA is a monotonization process.

Thus, we enter the phase of rule induction with a consistent dataset.

5. Ensemble of Decision Rules

This section describes the general scheme for decision rule induction. It follows from the previous sections that we can focus only on the binary classification case, since the multi-class case is reduced to the sequence of $K-1$ binary problems. Thus, assume that $Y = \{-1, 1\}$, where a "positive" class is ranked higher (in the order) to a "negative" class.

Decision rule is a logical statement of the form: $if\ [condition], then\ [decision]$. Let X_j be the set of all possible values for the j-th attribute. Condition part of the rule consists of elementary expressions of the form $x_j \geq s_j$ or $x_j \leq s_j$ for some $s_j \in X_j$. Let Φ denote the set of elementary expressions constituting the condition part of the rule, and let $\Phi(\mathbf{x})$ be an indicator function equal to 1 if an object $\mathbf{x}$ satisfies the condition part of the rule (we also say that a rule *covers* an object), otherwise $\Phi(\mathbf{x}) = 0$. The decision is a single real value, denoted by α. Therefore, we define a decision rule as:

$$r(\mathbf{x}) = \alpha\Phi(\mathbf{x}). \tag{8}$$

Notice that the decision rule takes only two values, $r(\mathbf{x}) \in \{\alpha, 0\}$, depending whether $\mathbf{x}$ satisfies the condition or not. In this paper, we assume the classification function is a linear combination of M decision rules:

$$f(\mathbf{x}) = \alpha_0 + \sum_{m=1}^{M} r_m(\mathbf{x}), \tag{9}$$

where α_0 is a constant value, which can be interpreted as a default rule, covering the whole X. Object $\mathbf{x}$ is classified to the class indicated by the sign of $f(\mathbf{x})$. The combination (9) has very simple interpretation as a voting procedure: rules with positive α vote for the positive class, while rules with negative α – for the negative class. Object $\mathbf{x}$ is classified to the class with a higher vote (which is equivalent to the sign of $f(\mathbf{x})$). Notice that in order to maintain monotonicity of $f(\mathbf{x})$, it is necessary and sufficient that for the m-th rule, α_m is positive when all elementary expressions in Φ_m are of the form $x_j \geq s_j$; similarly, for negative α_m all the conditions must be of the form $x_j \leq s_j$.

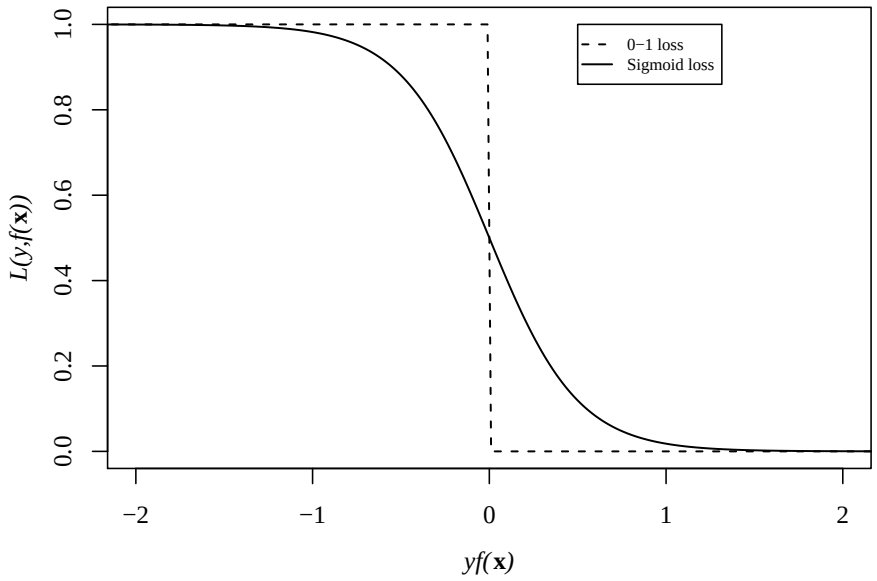

Figure 1. Sigmoid approximation of the 0-1 loss.

Rule induction is performed by minimizing the 0-1 loss function (classification error) on the set of n training examples (empirical risk). Notice that this loss function, is neither smooth nor differentiable. Therefore, we approximate it with the sigmoid function (see Figure 1):

$$\sigma(x) = \frac{1}{1+e^x}. \tag{10}$$

Thus, we minimize the following empirical risk:

$$R_{\text{emp}}(f) = \sum_{i=1}^{n} \sigma(y_i f(\mathbf{x}_i)). \tag{11}$$

However, finding a set of rules minimizing (11) is computationally hard, that is why we follow here FSAM, i.e., the rules are added one by one, greedily minimizing (11). We start with the default rule defined as:

$$\alpha_0 = \arg\min_{\alpha} R_{\text{emp}}(\alpha) = \arg\min_{\alpha} \sum_{i=1}^{n} \sigma(\alpha y_i). \tag{12}$$

The default rule is a real value but can be thought of as a rule covering the whole space X. In each subsequent iteration, a new rule is added by taking into account previously generated rules. Let $f_{m-1}(\mathbf{x})$ be a classification function after $m-1$ iterations, consisting of first $m-1$ rules and the default rule. The m-th decision rule $r_m(\mathbf{x}) = \alpha_m \Phi_m(\mathbf{x})$ should be obtained from $r_m = \arg\min_r R_{\text{emp}}(f_{m-1} + r)$, but this is still computationally hard, because we need to determine both the optimal $\Phi_m(\mathbf{x})$ and a_m simultaneously. Therefore, we restrict our algorithm to the case, in which all rules have decisions of the

Algorithm 1: Monotone Rule Ensemble (MORE).

input : set of n training examples $\{(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)\}$,
M – number of decision rules to be generated.
output: monotone rule ensemble $\{\alpha_0, r_1(\mathbf{x}), \ldots, r_M(\mathbf{x})\}$.

$\alpha_0 = \arg\min_{\alpha=\pm\beta} \sum_{i=1}^{n} \sigma(\alpha y_i)$;
$f_0(\mathbf{x}) = \alpha_0$;
for $m = 1$ *to* M **do**
  $\Phi_m^+(\mathbf{x}) = \arg\min_{\Phi} \mathcal{L}_m^+(\Phi)$;
  $\Phi_m^-(\mathbf{x}) = \arg\min_{\Phi} \mathcal{L}_m^-(\Phi)$;
  if $\mathcal{L}_m^+(\Phi_m^+) \leq \mathcal{L}_m^-(\Phi_m^-)$ **then**
    $\Phi_m = \Phi^+(m)$; $\alpha_m = \beta$;
  else
    $\Phi_m = \Phi^-(m)$; $\alpha_m = -\beta$;
  end
  $r_m(\mathbf{x}) = \alpha_m \Phi_m(\mathbf{x})$;
  $f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + r_m(\mathbf{x})$;
end

same magnitude, equal to β, i.e., we only allow $\alpha_m = \pm\beta$, where β is a parameter of the problem. With such a restriction, the optimal rule in the m-th iteration is defined as:

$$\Phi_m(\mathbf{x}) = \arg \min_{\Phi, \alpha=\pm\beta} \sum_{i=1}^{n} \sigma[y_i(f_{m-1}(\mathbf{x}_i) \pm \alpha\Phi(\mathbf{x}))], \tag{13}$$

and it requires calculating the values of sigmoid loss only at three points: $f_{m-1}(\mathbf{x}_i)$, $f_{m-1}(\mathbf{x}_i) + \beta$ and $f_{m-1}(\mathbf{x}_i) - \beta$, for every object $\mathbf{x}_i$. In each subsequent iteration, problem (13) can be solved via a heuristic procedure for rule generation, described below.

Let us define:

$$\mathcal{L}_m^+(\Phi) = \sum_{i=1}^{n} \sigma[y_i(f_{m-1}(\mathbf{x}_i) + \beta\Phi(\mathbf{x}))], \qquad \mathcal{L}_m^-(\Phi) = \sum_{i=1}^{n} \sigma[y_i(f_{m-1}(\mathbf{x}_i) - \beta\Phi(\mathbf{x}))].$$

Then, using (13), we can comprehensively write $\Phi_m(\mathbf{x}) = \arg\min \left\{ \min_{\Phi}\{\mathcal{L}_m^+(\Phi)\}, \min_{\Phi}\{\mathcal{L}_m^-(\Phi)\} \right\}$.

The general idea of the algorithm for finding Φ_m is the following: first we search for Φ_m^+ by minimizing $\mathcal{L}_m^+(\Phi)$, which corresponds to searching the best rule voting for the positive class. Next, we search for Φ_m^- by minimizing $\mathcal{L}_m^-(\Phi)$, which gives us the best rule voting for the negative class. Then, both rules are compared and the one with lower empirical risk is chosen, while another one is discarded.

The procedures for finding both rules Φ_m^+ and Φ_m^- are analogous, therefore, they are presented simultaneously:

- At the beginning, $\Phi_m^{\pm}$ is empty (no elementary condition is specified), i.e., $\Phi_m^{\pm}(\mathbf{x}) \equiv 1$.
- In each step, an elementary condition $x_j \geq s_j$ (for a rule voting for the positive class) or $x_j \leq s_j$ (for a rule voting for the negative class) is added to $\Phi_m^{\pm}$ that minimizes $\mathcal{L}_m^{\pm}(\Phi)$. Such expression is searched by consecutive testing of elementary conditions, attribute by attribute. Let

$x_j^{(1)}, x_j^{(2)}, \ldots, x_j^{(N)}$ be a sequence of ordered values of the j-th attribute, such that $x_j^{(i-1)} \geq x_j^{(i)}$, for $i = 2, \ldots, n$. Each elementary condition of the form $x_j \geq s_j$ (for rules voting for the positive class) or $x_j \leq s_j$ (for rules voting for the negative class) for each $s_j = \frac{x_j^{(i-1)} + x_j^{(i)}}{2}$ is tested.

- The previous step is repeated until $\mathcal{L}_m^{\pm}(\Phi)$ cannot be decreased.

The above procedure is very fast and proved to be efficient in computational experiments. The attributes can be sorted once before generating any rule. Notice that the procedure resembles the way the decision trees are generated. Here, however, we look for only one path from the root to the leaf instead of the whole decision tree. Let us underline that a minimal value of $\mathcal{L}_m^{\pm}(\Phi)$ is a natural stop criterion, what differentiates this procedure from those used for generation of decision trees.

The whole procedure is presented as Algorithm 1.

6. Analysis of the Step Length

We now analyze the behavior of the rule induction algorithm depending on the value of the parameter β, i.e., the magnitude of the decision for each rule. Notice that this parameter corresponds to the scale of the loss function, since:

$$f(\mathbf{x}) = \alpha_0 + \sum_{m=1}^{M} \alpha_m \Phi_m(\mathbf{x}) = \beta \left(\pm 1 + \sum_{m=1}^{M} \pm \Phi_m(\mathbf{x}) \right) = \beta \tilde{f}(\mathbf{x}),$$

where $\tilde{f}(\mathbf{x}) = \pm 1 + \sum_{m=1}^{M} \pm \Phi_m(\mathbf{x})$. Then:

$$\sigma(yf(\mathbf{x})) = \frac{1}{1 + \exp(\beta y \tilde{f}(\mathbf{x}))}.$$

Thus, small values of β cause the loss function to broaden and the changes on the slope of the function are smaller (the loss becomes similar to the linear function). On the other hand, large values of β cause the sigmoid loss to become similar to the 0-1 loss. In general, large values of β correspond to a more complex model [33], because we are able to decrease the error significantly in a smaller number of steps.

For a better insight into the problem, we state the following general theorem:

Theorem 6.1. Assume α_m is fixed and equal to $\pm\beta$. Minimization of (13) for any twice differentiable loss function $L(yf(\mathbf{x}))$ and for any β is equivalent to the minimization of:

$$\mathcal{L}(\Phi) = \sum_{i \in R_-} w_i^m + \frac{1}{2} \sum_{\Phi(\mathbf{x}_i)=0} \left(w_i^m - \beta v_i^m \right), \tag{14}$$

where:

$$R_- = \{i\colon\ \alpha_m y_i < 0\}, \tag{15}$$

$$w_i^m = -\frac{\partial}{\partial u} L(u) \Big|_{u = y_i f_{m-1}(\mathbf{x}_i)}, \tag{16}$$

$$v_i^m = \frac{1}{2} \frac{\partial^2}{\partial u^2} L(u) \Big|_{u = y_i f_{m-1}(\mathbf{x}_i) + \gamma_i \alpha_m y_i}, \tag{17}$$

for some $\gamma_i \in [0,1]$.

Proof:
From Taylor expansion it follows that for a twice differentiable loss function, we have:

$$L(u+z) = L(u) + z\frac{\partial}{\partial u}L(u) + \frac{z^2}{2}\frac{\partial^2}{\partial u^2}L(u+\gamma z)$$

for some $\gamma \in [0,1]$. By denoting $L_i = L(y_i(f_{m-1}(\mathbf{x}_i)))$, using (16)-(17) and substituting $u = y_i f_{m-1}(\mathbf{x}_i)$ and $z = \alpha_m y_i$, we have for every $\mathbf{x}_i$ such that $\Phi(\mathbf{x}_i) = 1$:

$$L(y_i(f_{m-1}(\mathbf{x}_i) + \alpha_m)) = L_i - \alpha_m y_i w_i^m + \beta^2 v_i^m.$$

Thus, the empirical risk becomes:

$$\mathcal{L}(\Phi) = \sum_{\Phi(\mathbf{x})_i=1} \left(L_i - \alpha_m y_i w_i^m + \beta^2 v_i^m\right) + \sum_{\Phi(\mathbf{x})_i=0} L_i.$$

The term $\sum_{i=1}^n L_i$ is constant, so it can be dropped from the optimization process. Thus, we equivalently minimize:

$$\mathcal{L}(\Phi) = \sum_{\Phi(\mathbf{x})_i=1} -\alpha_m y_i w_i^m + \beta^2 v_i^m = \sum_{i\in R_-} \beta w_i^m - \sum_{i\in R_+} \beta w_i^m + \sum_{\Phi(\mathbf{x})_i=1} \beta^2 v_i^m,$$

where $R_+ = \{i\colon\ \alpha_m y_i > 0\}$ and R_- is defined by (15). We now use the fact that $\sum_{i\in R_+} = \sum_{i=1}^n - \sum_{i\in R_-} - \sum_{\Phi(\mathbf{x}_i)=0}$ and that $\sum_{\Phi(\mathbf{x}_i)=1} = \sum_{i=1}^n - \sum_{\Phi(\mathbf{x}_i)=0}$ to obtain:

$$\mathcal{L}(\Phi) = \sum_{i=1}^n (\beta^2 v_i^m - \beta w_i^m) + 2\beta \sum_{i\in R_-} w_i^m + \beta \sum_{\Phi(\mathbf{x}_i)=0} (w_i^m - \beta v_i^m),$$

and by dropping the first constant term and dividing by constant value 2β, we prove the theorem. □

Thus, β establishes a trade-off between misclassified and unclassified examples. Values v_i^m are always positive, because the loss function is decreasing. Sigmoid loss is convex for $yf(\mathbf{x}) > 0$, and concave for $yf(\mathbf{x}) < 0$, therefore, as β increases, uncovered examples satisfying $y_i f_{m-1}(\mathbf{x}_i) > 0$ ("correctly classified") are penalized less, while the penalty for uncovered "misclassified" examples ($y_i f_{m-1}(\mathbf{x}_i) < 0$) increases. This leads to the following conclusion: although the rule covers only a part of the examples, with respect to uncovered examples it still tries to make a small error; remark that the weights of the uncovered examples depend on the curvature of the function (second derivative) rather than on the slope (first derivative).

7. Experimental Results

We found 12 datasets, for which it is known that monotonicity constraints should hold. We did not search for monotonicity directions by calculating any particular statistics, rather we obtained the directions using the domain knowledge about the problem. Four datasets, which are the results of surveys, were taken

Table 1. Description of datasets used in experiments.

DATA SET	#ATTRIBUTES	#OBJECTS	#CLASSES
ESL	4	488	8
SWD	10	1000	4
LEV	4	1000	5
ERA	4	1000	8
HOUSING	8	506	4
WINDSOR	11	546	4
DENBOSCH	9	119	2
WISCONSIN	9	699	2
LJUBLJANA	8	286	2
CAR	6	1728	4
CPU	6	209	4
BALANCE	4	625	3

from [2, 3]: *employee selection* (ESL), *social workers decisions* (SWD), *lecturers evaluation* (LEV) and *employee rejection/acceptance* (ERA). Three datasets are related to the problem of house pricing: *Boston housing* from the UCI repository [1], *Windsor housing* [29] and *DenBosch housing* [8]. From those three house pricing datasets only DenBosch dataset contained a discrete output variable (price discretized into two levels). We decided to discretize the price variable in (Boston) Housing and Windsor into four levels containing equal numbers of objects (i.e., quartiles of the price distribution), similarly as in [17].

There are also five other datasets taken from the UCI repository: *Wisconsin breast cancer*, *Ljubljana breast cancer*, *Car evaluation*, *CPU performance* (for which the class attribute was also discretized into four levels) and *Balance scale*.

For all datasets, objects with missing values were removed, since not every method is able to deal with missing values (rule ensembles have a very natural way of handling the missing values, which is included in our implementation, however, its description is beyond the scope of this paper). The quantitative characteristics of the datasets are shown in Table 1.

Absolute error loss was the measure of accuracy. For each dataset we tested three regular classifiers which do not take monotonicity constraints into account: support vector machines (SVM) with linear kernel [44], j48 decision trees [37] and nearest neighbors [15] (kNN). We used their implementations from Weka package [46]. Unfortunately, both SVM and j48 are designed to minimize 0-1 error and do not handle the order between class labels; using it directly on multi-class problems led to very poor results in terms of the mean absolute error. Therefore, we decided to improve its performance and use it in the ordinal setting by combining it with a simple approach to ordinal classification proposed in [18]. Since kNN estimates the conditional probabilities, we simply used the median of the distribution (minimizer of the absolute error). For SVM and j48, typical parameters from Weka were chosen; for kNN we increased the number of neighbors (k) to 5 in order to better estimate the conditional distribution.

For MORE we have chosen the number of rules $M = 50$ and the step length $\beta = 0.5$. *Those values were not optimized on the described datasets in any way*. For each dataset and for each algorithm, 10-fold cross validation was used and repeated 10 times to decrease the variance of the results. The measured

error rate is the mean absolute error, which is the value of the absolute error loss on the testing set. We calculated the average error along with the standard deviation (to avoid underestimation, standard deviation was calculated by taking into account the dependence between the subsequent testing samples in the repeated cross-validation, as described in [34]).

Table 2. Mean absolute error ± standard error. For each dataset, the best method and all methods within one standard error below the best are marked in bold.

DATASET	J48	SVM	KNN	MORE
DENBOSCH	0.172 ±0.031	0.202 ±0.034	0.199 ±0.034	**0.133** ±0.029
WISCONSIN	0.046 ±0.009	**0.03** ±0.007	**0.027** ±0.007	**0.031** ±0.006
ESL	0.369 ±0.021	**0.355** ±0.022	**0.345** ±0.023	**0.344** ±0.022
SWD	**0.442** ±0.015	**0.435** ±0.015	**0.433** ±0.016	**0.441** ±0.016
LEV	0.415 ±0.017	0.444 ±0.015	**0.398** ±0.017	**0.413** ±0.015
ERA	**1.217** ±0.031	1.271 ±0.028	1.278 ±0.031	1.269 ±0.029
HOUSING	0.332 ±0.022	0.314 ±0.024	0.326 ±0.023	**0.288** ±0.022
CPU	0.1 ±0.018	0.371 ±0.029	0.142 ±0.025	**0.065** ±0.016
BALANCE	0.271 ±0.021	**0.137** ±0.017	0.169 ±0.015	**0.126** ±0.014
LJUBLJANA	**0.259** ±0.02	0.299 ±0.023	**0.249** ±0.016	**0.251** ±0.021
WINDSOR	0.565 ±0.024	**0.491** ±0.025	0.604 ±0.025	0.538 ±0.024
CAR	0.09 ±0.007	0.078 ±0.007	0.086 ±0.004	**0.061** ±0.005

The results are shown in Table 2. For each dataset, the best method, and all methods within one standard error below the best, are marked in bold. Judging by the number of times MORE is among the best classifiers, we can conclude that an improvement in accuracy was obtained when using monotone rule ensembles over the regular classifiers. Notice, however, that improvement in prediction ability is not the only advantage: the model build consistently with the domain knowledge is more likely to be accepted and trusted by the domain experts, and, moreover, the decision rules entering the ensemble represent interesting patterns which have straightforward interpretation. For example, in the case of the "housing" dataset, we got the rules like:

if *nitric oxides concentration* ≥ 0.66
  and *% lower status of the population* ≥ 13.0
  and *average number of rooms per dwelling* ≤ 7.35
  and *crime rate* ≥ 2.15
$\Longrightarrow$ *price* is at most "low"

8. Conclusions

We introduced a novel rule induction algorithm, called MORE, for ordinal classification problem in the presence of monotonicity constraints. The algorithm uses forward stagewise additive modeling scheme

for generating an ensemble of decision rules for binary problems. We showed how to solve the ordinal classification problem with absolute error by solving binary subproblems with zero-one error. Due to specific nature of the problem, a syntax typical to monotone rules was used to find the statistically best ensemble. Moreover, we showed how to use stochastic DRSA to monotonize the dataset before the rules are generated. The main advantages of our algorithm is its comprehensibility (decision rules are probably the easiest model to interpret) and consistency with the domain knowledge. Moreover, the experimental results show that incorporating the domain knowledge about monotonicity in the classifier can significantly improve the prediction accuracy.

References

[1] Asuncion, A., Newman, D.: UCI Machine Learning Repository, 2007.

[2] Ben-David, A.: Automatic generation of symbolic multiattribute ordinal knowledge-based DSS: Methodology and applications, *Decision Sciences*, **23**, 1992, 1357–1372.

[3] Ben-David, A.: Monotonicity Maintenance in Information-Theoretic Machine Learning Algorithms, *Machine Learning*, **19**(1), 1995, 29–43.

[4] Berger, J. O.: *Statistical Decision Theory and Bayesian Analysis*, 2nd edition, Springer, 1993.

[5] Brunk, H. D.: Maximum likelihood estimates of monotone parameters, *Annals of Mathematical Statistics*, **26**(4), 1955, 607–616.

[6] Cao-Van, K.: *Supervised ranking: from semantics to algorithms*, Ph.D. dissertation, Ghent University, 2003.

[7] Cohen, W. W., Singer, Y.: A simple, fast, and effective rule learner, in: *Proc. of National Conference on Artificial Intelligence*, AAAI Press / The MIT Press, Orlando, USA, 1999, 335–342.

[8] Daniels, H., K. B.: Applications of MLP networks to bond rating and house pricing, *Neural Computation and Applications*, **8**, 1999, 226–234.

[9] Dembczyński, K., Greco, S., Kotłowski, W., Słowiński, R.: Statistical Model for Rough Set Approach to Multicriteria Classification, in: *Knowledge Discovery in Databases: PKDD 2007* (J. N. Kok, J. Koronacki, R. L. de Mántaras, S. Matwin, D. Mladenič, A. Skowron, Eds.), vol. 4702 of *Lecture Notes in Artificial Intelligence*, Springer, Warsaw, Poland, 2007, 164–175.

[10] Dembczyński, K., Kotłowski, W., Słowiński, R.: Ensemble of Decision Rules for Ordinal Classification with Monotonicity Constraints, in: *Rough Sets and Knowledge Technology 2008* (G. Wang, T. Li, J. W. Grzymała-Busse, D. Miao, A. Skowron, Y. Yao, Eds.), vol. 5009 of *Lecture Notes in Artificial Intelligence*, Springer, Chengdu, China, 2008, 260–267.

[11] Dembczyński, K., Kotłowski, W., Słowiński, R.: A General Framework for Learning an Ensemble of Decision Rules, in: *LeGo '08: From Local Patterns to Global Models, ECML/PKDD 2008 Workshop*, Antwerp, Belgium, 2008, 17–36.

[12] Dembczyński, K., Kotłowski, W., Słowiński, R.: Maximum likelihood rule ensembles, in: *Proc. of International Conference on Machine Learning* (W. W. Cohen, A. McCallum, S. T. Roweis, Eds.), Omnipress, Helsinki, Finland, 2008, 224–231.

[13] Dembczyński, K., Pindur, R., Susmaga, R.: Dominance-based Rough Set Classifier without Induction of Decision Rules, *Electronic Notes in Theoretical Computer Science*, **82**(4), 2003.

[14] Doumpos, M., Pasiouras, F.: Developing and Testing Models for Replicating Credit Ratings: A Multicriteria Approach, *Computational Economics*, **25**(4), 2005, 327–341.

[15] Duda, R. O., Hart, P. E., Stork, D. G.: *Pattern Classification*, 2nd edition, Wiley-Interscience, 2000.

[16] Dykstra, R., Hewett, J., Robertson, T.: Nonparametric, isotonic discriminant procedures, *Biometrica*, **86**(2), 1999, 429–438.

[17] Feelders, A., Pardoel, M.: Pruning for Monotone Classification Trees, in: *Advances in Intelligent Data Analysis V* (M. R. Berthold, H.-J. Lenz, E. Bradley, R. Kruse, C. Borgelt, Eds.), vol. 2810 of *Lecture Notes in Computer Science*, Springer, 2003, 1–12.

[18] Frank, E., Hall, M.: A simple approach to ordinal classification, in: *Proc. of European Conference on Machine Learning* (L. De Raedt, P. A. Flach, Eds.), vol. 2167 of *Lecture Notes in Artificial Intelligence*, Springer, Freiburg, Germany, 2001, 145–157.

[19] Friedman, J. H., Hastie, T., Tibshirani, R.: *Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2003.

[20] Friedman, J. H., Popescu, B. E.: *Predictive Learning via Rule Ensembles*, Technical report, Dept. of Statistics, Stanford University, 2005.

[21] Greco, S., Matarazzo, B., Słowiński, R.: A new rough set approach to evaluation of bankruptcy risk, in: *Operational Tools in the Management of Financial Risks* (C. Zopounidis, Ed.), Kluwer Academic Publishers, Dordrecht, 1998, 121–136.

[22] Greco, S., Matarazzo, B., Słowiński, R.: Rough approximation of a preference relation by dominance relations, *European Journal of Operational Research*, **117**, 1999, 63–83.

[23] Greco, S., Matarazzo, B., Słowiński, R.: The use of rough sets and fuzzy sets in Multiple-Criteria Decision Making, in: *Advances in Multiple Criteria Decision Making* (T.Gal, T.Stewart, T.Hanne, Eds.), chapter 14, Kluwer Academic Publishers, 1999, 14.1–14.59.

[24] Greco, S., Matarazzo, B., Słowiński, R.: Rough sets theory for multicriteria decision analysis, *European Journal of Operational Research*, **129**, 2001, 1–47.

[25] Greco, S., Matarazzo, B., Słowiński, R., Stefanowski, J.: An Algorithm for Induction of Decision Rules Consistent with the Dominance Principle, in: *Rough Sets and Current Trends in Computing* (W. Ziarko, Y. Yao, Eds.), vol. 2005 of *Lecture Notes in Artificial Intelligence*, Springer, Berlin, 2001, 304–313.

[26] Greco, S., Matarazzo, B., Słowiński, R., Stefanowski, J.: Variable consistency model of dominance-based rough set approach, in: *Rough Sets and Current Trends in Computing* (W. Ziarko, Y. Yao, Eds.), vol. 2005 of *Lecture Notes in Artificial Intelligence*, Springer, Berlin, 2001, 170–181.

[27] Herbrich, R., Graepel, T., Obermayer, K.: *Regression models for ordinal data: A machine learning approach*, Technical report TR-99/03, Technical University of Berlin, 1999.

[28] Jacob, V., Krishnan, R., Ryu, Y. U.: Internet Content Filtering Using Isotonic Separation on the PICS Specification, *ACM Transactions on Internet Technology*, **7**(1), 2007.

[29] Koop, G.: *Analysis of Economic Data*, John Wiley and Sons, 2000.

[30] Kotłowski, W., Dembczyński, K., Greco, S., Słowiński, R.: Stochastic Dominance-based Rough Set Model for Ordinal Classification, *Information Sciences*, **178**(21), 2008, 4019–4037.

[31] Lin, H.-T., Li, L.: Large-Margin Thresholded Ensembles for Ordinal Regression: Theory and Practice, in: *Algorithmic Learning Theory* (J. L. Balcázar, P. M. Long, F. Stephan, Eds.), vol. 4264 of *Lecture Notes in Artificial Intelligence*, Springer, 2006, 319–333.

[32] Lin, H.-T., Li, L.: Ordinal Regression by Extended Binary Classifications, in: *Proc. of Advances in Neural Information Processing Systems* (B. Schölkopf, J. C. Platt, T. Hoffman, Eds.), vol. 19, MIT Press, 2007, 865–872.

[33] Mason, L., Baxter, J., Bartlett, P., Frean, M.: Functional Gradient Techniques for Combining Hypotheses, in: *Advances in Large Margin Classifiers* (P. Bartlett, B. Schölkopf, D. Schuurmans, A. J. Smola, Eds.), MIT Press, 1999, 33–58.

[34] Nadeau, C., Bengio, Y.: Inference for the Generalization Error, *Machine Learning*, **52**(3), 2003, 239–281.

[35] Pawlak, Z.: Rough sets, *International Journal of Information & Computer Sciences*, **11**, 1982, 341–356.

[36] Potharst, R., Feelders, A. J.: Classification trees for problems with monotonicity constraints, *SIGKDD Explorations*, **4**(1), 2002, 1–10.

[37] Quinlan, J. R.: *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

[38] Robertson, T., Wright, F. T., Dykstra, R. L.: *Order Restricted Statistical Inference*, John Wiley & Sons, 1998.

[39] Roy, B.: *Multicriteria Methodology for Decision Aiding*, Kluwer Academic Publishers, Dordrecht, 1996.

[40] Ryu, Y. U., Chandrasekaran, R., Jacob, V.: Data Classification Using the Isotonic Seeparation Technique: Application to Breast Cancer Prediction, *Eurpoean Journal of Operational Research*, **181**(2), 2007, 842–854.

[41] Ryu, Y. U., Yue, W. T.: Firm Bankruptcy Prediction: Experimental Comparison of Isotonic Separation and Other Classification Approaches, *IEEE Transactions on Systems, Man and Cybernetics*, **35**(5), 2005, 727–737.

[42] Sill, J., Abu-Mostafa: Monotonicity Hints, in: *Advances in Neural Information Processing Systems* (M. C. Mozer, M. I. Jordan, T. Petsche, Eds.), vol. 9, The MIT Press, Denver, USA, 1997, 634–640.

[43] Słowiński, R., Zopounidis, C.: Application of the rough set approach to evaluation of bankruptcy risk, *International Journal of Intelligent Systems in Accounting, Finance and Management*, **4**(1), 1995, 27–41.

[44] Vapnik, V.: *Statistical Learning Theory*, Wiley, 1998.

[45] Weiss, S. M., Indurkhya, N.: Lightweight Rule Induction, in: *Proc. of International Conference on Machine Learning* (P. Langley, Ed.), Morgan Kaufmann, Stanford, USA, 2000, 1135–1142.

[46] Witten, I. H., Frank, E.: *Data Mining: Practical machine learning tools and techniques, 2nd Edition*, Morgan Kaufmann, San Francisco, 2005.