Poznań University of Technology
Institute of Computing Science

Wojciech Kotłowski

Statistical Approach to Ordinal Classification with Monotonicity Constraints

Doctoral Dissertation

Submitted to the Council of the Faculty of Computer Science and Management
of Poznań University of Technology

Supervisor: Ph.D., Dr. Habil., Professor Roman Słowiński

Poznań, Poland
2008

Institute of Computing Science
Poznan University of Technology

Typeset using LaTeX in Computer Modern.

BibTeX:

```
@phdthesis{ key,
    author = "Wojciech Kot{\l}owski",
    title = "{Statistical Approach to Ordinal Classification with Monotonicity Constraints}",
    school = "Poznan University of Technology",
    address = "Pozna{\'n}, Poland",
    year = "2008",
}
```

Contents

Preface

The role of machine learning is to generalize from a data sample. In order to generalize, every learning algorithm must be biased. Experience indicates that when this bias comes from the domain knowledge, the algorithm has a chance to generalize better. A typical domain knowledge encountered in applications of machine learning are the statements about orders and about relationships among these orders in data. Indeed, although experts may fail in providing quantitative relationships between attributes of a system under study, they can usually describe their qualitative characteristics in terms of orders of value sets of these attributes, and in terms of monotone dependencies between these orders (e.g., "the higher, the better"), which happen to be the relations that are the easiest to express.

In the problem of *ordinal classification* (also referred to as *ordinal regression*), the purpose is to predict for a given object one of the ordered class labels. In this thesis, we require a stronger assumption: a meaningful ordering exists not only between class labels, but also on the value set (domain) of each attribute. Moreover, we assume that we are given a domain knowledge about the problem expressed by the *monotonicity constraints*: a higher value of an object on an attribute, with other values being fixed, should not decrease its class assignment. Thus, we focus on the problem of *ordinal classification with monotonicity constraints*. The monotonicity constraints are commonly encountered in real-life applications, but rarely taken into account in the theoretical considerations in machine learning. Neglecting the constraints may lead, however, to worse accuracy of the classifiers and to inconsistencies in the model. On the other hand, taking the monotonicity constraints into account requires a dedicated, specific approach. Such an approach, along with in-depth analysis of the problem, is proposed in this thesis.

The success of the data analysis was possible due to a continuous improvement of computing resources, but also due to the invention of efficient and accurate algorithms. Those algorithms mainly come from two different research fields – statistics and artificial intelligence, subfield of computer science. Although these two fields have very different origins, it soon have become clear that they heavily overlap. Statistics has a great impact on the development of artificial intelligence, but also artificial intelligence made an influence on the way of thinking in statistical community.

Having in mind the above, we aim in this thesis at providing a statistical framework for ordinal classification with monotonicity constraints. Up to our knowledge, such a framework has not been introduced before. We explain existing nonparametric approaches within this framework and propose a generalization of these approaches. We also explain the Dominance-based Rough Set Approach (DRSA) from statistical point of view. DRSA has its roots in logic and was the first theory of ordinal classification problems with monotonicity constraints. We provide a natural probabilistic extension of this theory, Stochastic DRSA, and show its connections with statistical estimation and Bayesian decision theory.

Apart from theoretical foundations of the considered problem, we also propose efficient methods for solving it. The methods are based on decision rules, combined into an ensemble of classifiers. This corresponds to a modern approach to rule induction, which employs *boosting* strategy to

learning. We provide a theoretical analysis which shows that the monotonicity assumptions allow us to bound the difference between the performance of a rule ensemble and the performance of an optimal classifier in terms of the empirically measurable value of the so called margin. High performance and good scalability of the proposed rule ensemble methods are verified in an extensive computational experiment.

Acknowledgments. Numerous people contributed to my research in different ways. I would like to deeply thank to my supervisor Professor Roman Słowiński for his tremendous support, invaluable help, inspiration and insightful remarks, which made this work definitely much better. I also express my gratitude to Krzysztof Dembczyński and Salvatore Greco for a very friendly and fruitful cooperation and inspiring ideas during countless discussions.

Finally, I wish to thank to my parents, my brothers and Ania for their love, warm support and endless patience. Without them, I could not have brought this work to an end.

Chapter 1

Introduction

1.1 Problem Setting

Classification. Machine learning originated in the field of artificial intelligence but also has deep connections with probability theory and statistics (Duda et al., 2000; Friedman et al., 2003; Vapnik, 1999; Koronacki and Ćwik, 2005). It concerns designing algorithms which have ability to learn. The learning is done by providing the algorithm a set of previously observed *training* examples (also called *objects* or *observations*), described by a set of *attributes*. Here we deal with a *classification problem*, in which for all the training objects we know a priori the values of a particular attribute with finite domain, called *class* (or *decision*) *attribute*. The goal of the learning process is to predict the value of the class attribute (*class label*) on unseen objects as accurate as possible, using the known values of other attributes.

Domain knowledge. A very important issue in the learning process is the utilization of the domain (expert) knowledge. In order to generalize, every learning algorithm must be somehow biased (prefer some hypothesis over the others although their similar behavior on the data sample), and when it is biased by the domain knowledge it has a better chance to be biased in the right way (Wolpert, 1996). This results in the improvement of prediction accuracy, but it is not the only advantage: the model consistent with domain knowledge is easier to interpret and easier to be accepted by the domain experts. Notice that although interpretability issues cannot be expressed in a quantitative way, they often play much more important role in real-life applications than a small gain in accuracy.

A typical knowledge encountered in real-life applications of machine learning is the knowledge about order and monotonicity. Exploiting this kind of knowledge in the learning process is the main purpose of this thesis.

Ordinal classification. An *ordinal classification* problem consists in assignment of objects to classes, for which a meaningful order exists. A good example is the classification of documents into three groups "irrelevant", "partly relevant", "relevant" (Herbrich et al., 1999). As another example, consider classifying patients receiving the chemotherapy with respect to the severity of nausea into the classes "none", "mild", "moderate" and "severe" (Anderson, 1984). The distances between class labels are usually meaningless, since the scale is assumed to be purely ordinal.

The ordinal classification problem shares some characteristics of multi-class classification and regression. However, on the contrary to the classification, the order between class labels cannot be neglected, and on the contrary to the regression, the scale on the output attribute is not cardinal.

Monotonicity constraints. In the *ordinal classification with monotonicity constraints* we require a meaningful ordering not only between class labels, but also on the value set of each attribute. Moreover, we assume that *monotonicity constraints* are present in the data: a higher value of an object on an attribute, with values on other attributes being fixed, should not decrease its class assignment; in other words, the expected class label should not decrease with increasing values on condition attributes.

As an example, consider the customer satisfaction analysis (Greco et al., 2006), which aims at determining customer preferences in order to optimize decisions about strategies for launching new products, or about improving the image of existing products. The monotonicity constraints are of fundamental importance here. Indeed, consider two customers, A and B, and suppose that the evaluations of a product by customer A on a set of attributes are better than the evaluations by customer B. In this case, it is reasonable to expect that also the comprehensive evaluation of this product (its class label) by customer A is better (or at least not worse) than the comprehensive evaluation made by customer B.

As another 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, bathrooms, garages, whether house contains air conditioning, basement, etc. (Koop, 2000). It is apparent 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 bathrooms than B, and opposite to B, has basement, and is as good as B on the other attributes.

Problems of ordinal classification in the presence of monotonicity constraints are commonly encountered in real-life applications, where ordinal and monotone properties follow from the domain knowledge about the problem. They are encountered in such problems as bankruptcy risk prediction (Słowiński and Zopounidis, 1995; Greco et al., 1998; Ryu and Yue, 2005), breast cancer diagnosis (Ryu et al., 2007), house pricing (Potharst and Feelders, 2002), Internet content filtering (Jacob et al., 2007), credit rating (Doumpos and Pasiouras, 2005), liver disorder diagnosis (Sill and Abu-Mostafa, 1997), credit approval (Feelders and Pardoel, 2003), surveys data (Cao-Van, 2003) and many others. The problem of ordinal classification with monotonicity constraints is widely considered under the name *multicriteria sorting* within multicriteria decision analysis (MCDA) (Roy, 1996; Grabisch, 1996; Greco et al., 2001a, 2002a; Słowiński et al., 2005; Greco et al., 2008). Thus, we should add to the above list numerous applications of multicriteria sorting methods to real-life decision problems.

Nevertheless, ordinal classification with monotonicity constraints is rarely considered in the machine learning community. There are only few methods which take the monotone nature of data into account. What is even worse, no comprehensive theoretical approach has been established. This thesis aims at creating such approach from the statistical point of view.

1.2 Problem Statement

In this section, we overview the statistical theory of machine learning and introduce the concept of ordinal classification and dominance relation. This will serve as a basis for the definition of ordinal classification with monotonicity constraints in Chapter 2.

1.2.1 Statistical Theory for Machine Learning

Classification problem. In the classification problem (Friedman et al., 2003), the aim is to predict the unknown class label $y \in Y$ for a given object (assign object to a class), where $Y =$

$\{1, \ldots, K\}$ is the set of K class labels. This is usually done by using the available knowledge about the object, expressed in terms of a vector of attributes $\mathbf{x} = (x_1, \ldots, x_m) \in X$, where X is called *attribute space*. Here, we assume without loss of generality that the value set of each attribute is a subset of $\mathbb{R}$, so that $X \subseteq \mathbb{R}^m$. It is assumed that objects are generated independently according to some fixed but unknown probability distribution $P(\mathbf{x}, y)$.

Classification is performed by constructing a function $h(\mathbf{x})$ (called *classifier*) which predicts accurately the value of y. The accuracy of the single prediction is measured in terms of the *loss function* $L(y, h(\mathbf{x}))$, which reflects the cost of predicting the class label $h(\mathbf{x})$ when the actual (observed) value is y. The overall accuracy of the classifier $h(\mathbf{x})$ is measured by the expected loss (*risk*) according the data distribution $P(\mathbf{x}, y)$:

$$R(h) = \mathbb{E}[L(y, h(\mathbf{x}))]. \tag{1.1}$$

Bayes classifier. Thus, the optimal prediction function is the function minimizing the risk (1.1):

$$h^* = \arg\min_h R(h), \tag{1.2}$$

which is called *Bayes classifier* (Berger, 1993), and $R^* := R(h^*)$ is called *Bayes risk*. From the definition it follows that Bayes risk is the minimal possible risk achievable by any prediction function in a given problem.

It follows from statistical decision theory (Berger, 1993) that Bayes classifier is obtained by minimizing the risk according to so called *posterior* distribution $P(y|\mathbf{x})$. The derivation of this fact is the following. First, decompose $P(\mathbf{x}, y)$ as $P(y|\mathbf{x})P(\mathbf{x})$. Then:

$$R(h) = \int L(y, h(\mathbf{x}))dP(\mathbf{x}, y) = \int \left(\int L(y, h(\mathbf{x}))dP(y|\mathbf{x}) \right) dP(\mathbf{x}) \tag{1.3}$$

which is a decomposition of the form $\mathbb{E}[\cdot] = \mathbb{E}\left[\mathbb{E}[\cdot|\mathbf{x}]\right]$. From (1.3) it follows that the Bayes classifier must minimize the term in parenthesis for any $\mathbf{x}$:

$$h^*(\mathbf{x}) = \arg\min_z \int L(y, z)dP(y|\mathbf{x}) = \arg\min_z \mathbb{E}[L(y, z)|\mathbf{x}] \tag{1.4}$$

Empirical risk minimization. Since the probability distribution $P(\mathbf{x}, y)$ is unknown, we cannot obtain h^*. The learning procedure uses a *training set* $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset X$ only to construct h to be a good approximation of h^*. Usually, it is performed by minimization of the *empirical risk*:

$$R_{\text{emp}}(h) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, h(\mathbf{x}_i)), \tag{1.5}$$

where function h is chosen from a restricted, predefined class of functions $\mathcal{H}$ (linear functions, classification trees, etc.), for which we believe that $h^* \in \mathcal{H}$ or at least that h^* may be approximated by some function from $\mathcal{H}$. This procedure is usually called empirical risk minimization (ERM) (Vapnik, 1999).

Loss functions In the most general case, the loss function takes the matrix form $[l_{yk}]_{K \times K}$, where $l_{yk} = L(y, k)$. An obvious assumption is that $l_{kk} = 0$ for every $k = 1, \ldots, K$ (the correct prediction is not penalized), and $l_{yk} > 0$ for each $y \neq k$ (any incorrect prediction is penalized).

The most popular loss function for classification is *zero-one loss*, defined as:

$$L_{0-1}(y, h(\mathbf{x})) = \begin{cases} 0 & \text{if } y = h(\mathbf{x}) \\ 1 & \text{if } y \neq h(\mathbf{x}) \end{cases}, \tag{1.6}$$

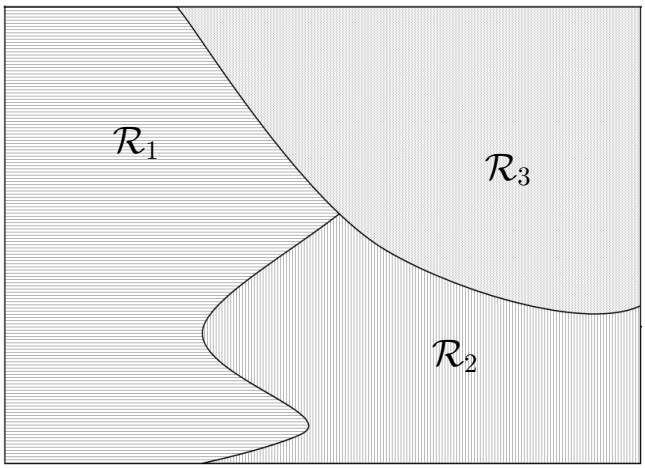

Figure 1.1: An example of Bayes decision boundary between three classes on $\mathbb{R}^2$

so that the misclassification is always penalized with unit loss, and for correct classification no penalty is imposed. This corresponds to the matrix with zeros on the diagonal and ones anywhere else, $l_{yk} = 1_{y \neq k}$[1]. The Bayes classifier has the form:

$$h^*_{0-1}(\mathbf{x}) = \arg\max_{y \in \{1,\ldots,m\}} P(y|\mathbf{x}) \tag{1.7}$$

so it indicates the most probable class at $\mathbf{x}$. The Bayes classifier divides the attribute space X into K separate regions $\mathcal{R}_1, \ldots, \mathcal{R}_K$, such that $h^*(\mathbf{x}) = k$ if and only if $\mathbf{x} \in \mathcal{R}_k$. The boundary between those regions is called *Bayes decision boundary.* It consists of points, for which there are more than one class with the highest probability (mode of the distribution is not unique). A simple example is shown in Figure 1.1, with $X = \mathbb{R}^2$. Although in each region there might be nonzero probability for any class, in the region $\mathcal{R}_k$ class k is always the most probable.

1.2.2 Ordinal Classification and Monotonicity Constraints

Ordinal loss matrix. In the ordinal classification setting, the loss matrix should be "consistent" with the order between class labels in the sense that the loss should not decrease, as the predicted value "moves away" from the true value. Thus, for any loss matrix $[l_{yk}]_{K \times K}$ we assume that each row of the loss matrix is V-shaped (Lin and Li, 2007):

$$\begin{aligned} l_{y,k-1} &\geqslant l_{yk} \qquad \text{if } k \leqslant y, \\ l_{yk} &\leqslant l_{y,k+1} \qquad \text{if } k \geqslant y, \end{aligned} \tag{1.8}$$

for each $y, k = 1, \ldots, K$. Such matrix will be called *ordinal loss matrix.* It is straightforward to see that 0-1 loss (1.6) is a proper ordinal loss matrix, however it does not take order into account, because every misclassification is given the same penalty. This is different for other two popular ordinal loss matrices: absolute and squared error loss. *Absolute error loss* is defined as:

$$L_{\text{abs}}(y, h(\mathbf{x})) = |y - h(\mathbf{x})|, \tag{1.9}$$

[1] 1_c is the indicator function, i.e. $1_c = 1$ if c is true, otherwise $1_c = 0$

so that the penalty for misclassification grows linearly with the difference between real and predicted class labels. One can show (Berger, 1993; Friedman et al., 2003) that the Bayes classifier for absolute error loss has the form:

$$h^*_{\text{abs}}(\mathbf{x}) = \text{med}(y|\mathbf{x}), \tag{1.10}$$

the median of the conditional distribution $P(y|\mathbf{x})$. Notice that the median is a purely ordinal operation: if class labels were encoded by arbitrary real numbers, the Bayes classifier would remain the same. This is not the case of another ordinal loss matrix, *squared error loss*:

$$L_{\text{sqr}}(y, h(\mathbf{x})) = (y - h(\mathbf{x}))^2, \tag{1.11}$$

The Bayes classifier is then (Berger, 1993; Friedman et al., 2003):

$$h^*_{\text{sqr}}(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}], \tag{1.12}$$

the expected value over the conditional distribution $P(y|\mathbf{x})$. Taking the expected value is not an ordinal operation, because it depends on the particular encoding of class labels and usually does not belong to $\{1, \ldots, K\}$. Thus, we state that absolute loss is the canonical loss for ordinal classification.

There is another approach for incorporating the order between classes, in which the ordinal classification is treated as a special case of a *ranking problem*. It is described in the Section 1.2.3.

Dominance relation and monotone functions. A *dominance* relation $\succeq$ is a binary relation on X, defined in the following way: for any $\mathbf{x}, \mathbf{x}' \in X$ we say that $\mathbf{x}$ *dominates* $\mathbf{x}'$, $\mathbf{x} \succeq \mathbf{x}'$, if on every attribute, $\mathbf{x}$ has evaluation not worse than $\mathbf{x}'$, $x_j \geqslant x'_j$, for all $j = 1, \ldots, m$:

$$\mathbf{x} \succeq \mathbf{x}' \iff \forall_{j=1,\ldots,m}\, x_j \geqslant x'_j.$$

The dominance relation $\succeq$ is a partial preorder on X, i.e. it is reflexive and transitive. A function $h\colon X \to Y$ is called *monotone* if the following condition holds for any $\mathbf{x}, \mathbf{x}' \in X$:

$$\mathbf{x} \succeq \mathbf{x}' \to h(\mathbf{x}) \geqslant h(\mathbf{x}').$$

The vector $\mathbf{v} = (v_1, \ldots, v_n)$ is called *monotone* if for every $i, j = 1, \ldots, n$:

$$\mathbf{x}_i \succeq \mathbf{x}_j \to v_i \geqslant v_j.$$

The difference between monotone functions and vectors is that the latter corresponds to the training set D only, while the former – to the whole space X.

Monotonicity constraints. The concept of monotone function is the core of what we intuitively understand by monotonicity constraints. Therefore, the problem of ordinal classification with monotonicity constraints is often referred to as the problem of classification with monotone functions, i.e. the problem of finding the accurate classifier within the class of monotone functions.

In order to justify imposing monotonicity constraints, one usually assumes that the process generating the data has monotone nature and the aim is then to "discover" (approximate) the process using the available training data D. Such definition is stated, more or less explicitly, in most of the papers dealing with ordinal classification with monotonicity constraints (Ben-David, 1995; Greco et al., 1998, 1999a,b, 2001a; Potharst and Feelders, 2002; Cao-Van, 2003; Popova, 2004; Chandrasekaran et al., 2005; Velikova, 2006). In our probabilistic setting, this corresponds to the statement that the Bayes classifier h^* is a monotone function. An example of decision boundaries with monotone Bayes classifier is shown in Figure 1.2. The problem with such a formulation is that the Bayes classifier is not a primitive concept in classification problems, it rather follows from the form of the loss function and from the data distribution. Therefore, we postpone the formal definition to the Chapter 2, in which we introduce a probabilistic model for monotonicity constraints.

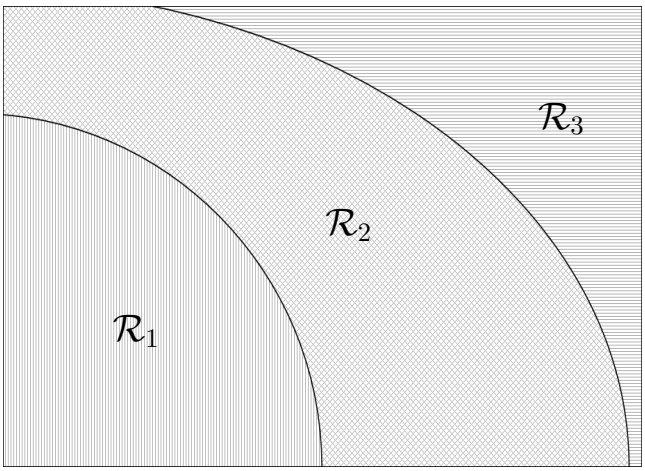

Figure 1.2: An example of monotone Bayes decision boundary between three classes on $\mathbb{R}^2$. Compare with non-monotone case in Figure 1.1

Inconsistencies. The monotonicity constraints imply that it should hold:

$$\mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow y_i \geqslant y_j, \tag{1.13}$$

for $i, j = 1, \ldots, n$, i.e. vector $\mathbf{y} = (y_1, \ldots, y_n)$ should be monotone. If such property holds, we say that the dataset D is *consistent* (or *monotone*). However, for many real datasets, (1.13) is not satisfied. We say that an object $\mathbf{x}_i$ is *inconsistent* if there exists another object $\mathbf{x}_j$, such that $\mathbf{x}_i, \mathbf{x}_j$ violate (1.13). In other words, there is $\mathbf{x}_j$ such that $\mathbf{x}_i$ dominates $\mathbf{x}_j$ but $y_i < y_j$, or such that $\mathbf{x}_i$ is dominated by $\mathbf{x}_j$ but $y_i > y_j$. We say that object $\mathbf{x}_i$ is *consistent* if it is not inconsistent.

Inconsistencies are usually avoided, because none monotone function can approximate accurately inconsistent objects. Some methods for ordinal classification with monotonicity constraints work only with consistent data. Others try to remove inconsistencies and operate only on the consistent part of the data.

1.2.3 Rank Loss Function

Ranking and scoring. *Ranking* is defined on a pair of objects: we consider a *ranking function* $r(\mathbf{x}, \mathbf{x}')$ such that $r(\mathbf{x}, \mathbf{x}') > 0$ if $\mathbf{x}$ is ranked higher than $\mathbf{x}'$. The ranking loss (Clémençon et al., 2006) is defined as:

$$L_{\text{rank}}(r(\mathbf{x}, \mathbf{x}'), y - y') = 1_{r(\mathbf{x}, \mathbf{x}')(y - y') < 0}, \tag{1.14}$$

where y, y' are ranks (class labels). Thus, ranking function is penalized with a unit loss if it makes a ranking error: $\mathbf{x}$ is ranked lower (higher) than $\mathbf{x}'$, while $y > y'$ ($y < y'$). The aim is to find a prediction function minimizing the risk:

$$R_{\text{rank}}(r) = \int L_{\text{rank}}(r(\mathbf{x}, \mathbf{x}'), y - y') dP(\mathbf{x}, y) dP(\mathbf{x}', y'). \tag{1.15}$$

The Bayes classifier has the simple form:

$$r^*(\mathbf{x}, \mathbf{x}') = \text{sign}\Big(P(y > y' | \mathbf{x}, \mathbf{x}') - P(y' > y | \mathbf{x}, \mathbf{x}') \Big), \tag{1.16}$$

In the core of rank loss approach lies the assumption of the existence of the optimal *scoring function* (Clémençon et al., 2006), i.e. single-argument function $s^*(\mathbf{x})$ such that:

$$r^*(\mathbf{x}, \mathbf{x}') > 0 \iff s^*(\mathbf{x}) > s^*(\mathbf{x}'), \tag{1.17}$$

and then the rank loss takes the form:

$$L_{\text{rank}}(s(\mathbf{x}) - s(\mathbf{x}'), y - y') = 1_{(s(\mathbf{x})-s(\mathbf{x}'))(y-y')<0}. \tag{1.18}$$

The assumption (1.17) must be satisfied, otherwise there would be no function of a single argument $s^*(\mathbf{x})$ achieving Bayes risk. Herbrich et al. (1999) considered the problem of ordinal classification as finding the scoring function (called "function inducing ordering" in their work) which minimizes the risk (1.15) with the SVM classification method. Freund et al. (2003) proposed the extension of AdaBoost in the rank loss formulation (*RankBoost*).

Analysis. The rank loss approach has nonparametric form and (seemingly) avoids imposing any metric on the ordinal scale of ranks, as the error is measured by a number of rank inverses. However, there are two drawbacks. The first one is a computational issue: the complexity of the problem grows quadratically with n, making the method computationally infeasible for larger datasets. The second drawback is more fundamental. The scoring function is real-valued, so the output of the ranking procedure is the real number, rather than class label. Therefore, one must somehow estimate the position of the thresholds on the scale of scoring function to change the real values into the class labels. In (Herbrich et al., 1999) the positions of the thresholds were obtained by a margin-based procedure. Generally, the threshold can be obtained only by minimizing (implicitly or explicitly) some loss function. Thus, at the end we anyway do the thing which we tried to avoid: impose some loss function on the ranks' scale.

We believe the ordinal classification is *different* to ranking, and our belief is supported by achievements in multicriteria decision analysis (MCDA) (Roy, 1996; Greco et al., 2002a): in the ordinal classification (multicriteria sorting in this context) one does not compare objects with each other (as in ranking), one rather assigns them to ordered classes on the basis of their values on considered attributes. The assignment is made by comparing objects with the "class profiles", which are artificial objects associated with each class: if the object is better than the k-th class profile, its class must be at least k. This approach can be translated to a modification of the ranking problem by stating that each object is compared only with the "rank of the class" and considering the "ranking" function $r(\mathbf{x}, k)$. Then, the class label predicted for $\mathbf{x}$ is the smallest k for which $\mathbf{x}$ is still ranked lower: $h(\mathbf{x}) = \min\{k\colon \text{sign}(r(\mathbf{x}, k)) < 0\}$ ($h(\mathbf{x}) = K$ is no such k exists). But if for a given $\mathbf{x}$, we sum the rank loss for each class label, we obtain:

$$\sum_{k=1}^{K} L_{\text{rank}}(r(\mathbf{x}, k), y - k) = L_{\text{abs}}(h(\mathbf{x}), y),$$

the absolute error loss. Thus, we conjecture that the absolute error is the canonical loss function for ordinal classification.

1.3 Existing Approaches to Ordinal Classification with Monotonicity Constraints

We overview the existing approaches to ordinal classification with monotonicity constraints in the fields of rough sets, multicriteria decision analysis (MCDA), statistics and machine learning/knowledge discovery. In description of fuzzy integrals, Ordinal Learning Model and monotone decision trees we base on the survey of monotone classification methods in (Cao-Van, 2003).

OBJECT	a_1	a_2	y	OBJECT	a_1	a_2	y
$\mathbf{x}_1$	0.15	0.3	1	$\mathbf{x}_4$	0.6	0.5	3
$\mathbf{x}_2$	0.3	0.15	1	$\mathbf{x}_5$	0.7	0.7	2
$\mathbf{x}_3$	0.4	0.8	2	$\mathbf{x}_6$	0.9	0.8	3

Table 1.1: A set of objects described by 2 attributes a_1, a_2 and class label $y \in \{1, 2, 3\}$.

1.3.1 Dominance-based Rough Set Approach

Dominance-based Rough Set Approach (DRSA) proposed by Greco, Matarazzo, and Słowiński (1999a,b, 2001a) is an extension of the classical rough set approach (Pawlak, 1982) for dealing with multicriteria sorting. By substituting the indiscernibility relation with the dominance relation, it handles inconsistencies coming from violation of the monotonicity constraints. DRSA originated from the fusion of rough set theory and MCDA. It was the first approach, which proposed the comprehensive theory for ordinal classification problems with monotonicity constraints in the domain of knowledge discovery. In this section, we only sketch the idea of DRSA, while a deeper insight is postponed to Chapter 4. Extensive surveys of DRSA can be found in (Greco et al., 2001a, 2004b,c; Słowiński et al., 2005).

Class unions and approximations. DRSA divides the monotone K-class problem into $K-1$ binary subproblems. This is done by considering the *upward* and *downward unions of classes* which are subsets of data for which class label is at least (or at most) $k = 1, \ldots, K$:

$$Cl_k^{\geqslant} = \{\mathbf{x}_i \colon y_i \geqslant k, i = 1, \ldots, n\}$$
$$Cl_k^{\leqslant} = \{\mathbf{x}_i \colon y_i \leqslant k, i = 1, \ldots, n\}.$$

Then, the k-th problem, $k = 2, \ldots, K$ is defined as the problem of discrimination between objects from class union $Cl_k^{\geqslant}$ and $Cl_{k-1}^{\leqslant}$, i.e. the problem of binary classification with two classes $Cl_k^{\geqslant}$ and $Cl_{k-1}^{\leqslant}$. The core idea of DRSA (and other rough set approaches) is the following: instead of using the whole class unions, which might contain inconsistent objects, use subsets of unions, containing only consistent objects for a given binary problem. Those subsets are called *lower approximations*, and corresponds to the region of *certain* knowledge[2].

Let us define first the *dominating* and *dominated sets* for a given $\mathbf{x}$ as the subsets of objects which dominate (or are being dominated by) $\mathbf{x}$:

$$D^+(\mathbf{x}) = \{\mathbf{x}_i \colon \mathbf{x}_i \succeq \mathbf{x}, i = 1, \ldots, n\}$$
$$D^-(\mathbf{x}) = \{\mathbf{x}_i \colon \mathbf{x}_i \preceq \mathbf{x}, i = 1, \ldots, n\}.$$

We remind (see Section 1.2.2) that object $\mathbf{x}_i$ is inconsistent if there exists another object $\mathbf{x}_j$ which belongs to a lower class but $\mathbf{x}_j \succeq \mathbf{x}_i$, or which belongs to a higher class but $\mathbf{x}_j \preceq \mathbf{x}_i$. However, in the k-th binary problem we associate objects from $Cl_k^{\geqslant}$ with one ("positive") class, while objects from $Cl_{k-1}^{\leqslant}$ with another ("negative") class. Thus, in the k-th problem object $\mathbf{x}_i \in Cl_k^{\geqslant}$ is inconsistent if there exists $\mathbf{x}_j \in Cl_{k-1}^{\leqslant}$ such that $\mathbf{x}_j \succeq \mathbf{x}_i$ (and vice-versa). But this is equivalent to $D^+(\mathbf{x}_i) \nsubseteq Cl_k^{\geqslant}$ and $D^-(\mathbf{x}_j) \nsubseteq Cl_{k-1}^{\leqslant}$.

Having above in mind, we are ready to define *lower approximations* of upward and downward

[2]Regions of possible knowledge, *upper approximations*, are introduced in Chapter 4.

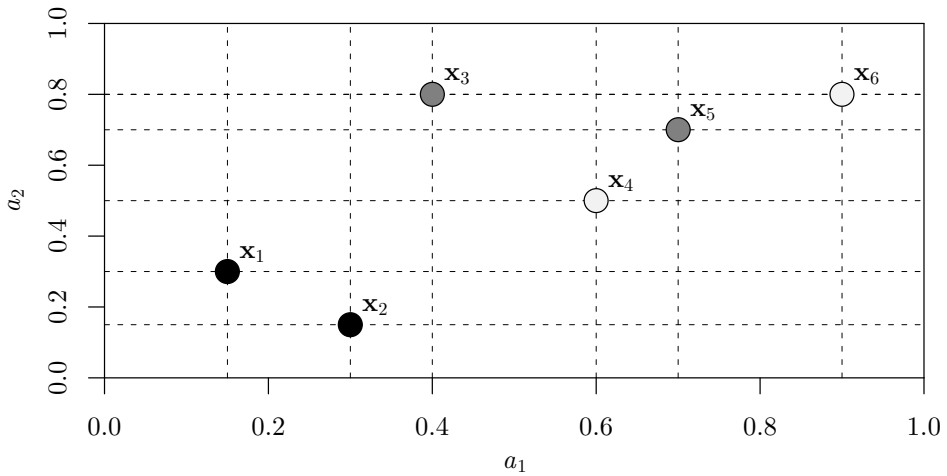

Figure 1.3: Example of three-class problem described in Table 1.1. Black points are objects from class 1, dark gray points – from class 2, light gray – from class 3.

unions of classes as:

$$\underline{Cl}_k^{\geqslant} = \{\mathbf{x}_i \colon D^+(\mathbf{x}_i) \subseteq Cl_k^{\geqslant}, i = 1, \ldots, n\},$$
$$\underline{Cl}_k^{\leqslant} = \{\mathbf{x}_i \colon D^-(\mathbf{x}_i) \subseteq Cl_k^{\leqslant}, i = 1, \ldots, n\}.$$

In other words, lower approximations of class unions contain consistent objects, thus can be regarded as certain regions of X for a given binary problem.

Example. A simple example of training set is shown in Table 1.1 and in Figure 1.3. Notice that object $\mathbf{x}_4$ is inconsistent with $\mathbf{x}_5$. We have $\underline{Cl}_2^{\geqslant} = \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$, $\underline{Cl}_3^{\geqslant} = \{\mathbf{x}_6\}$, $\underline{Cl}_1^{\leqslant} = \{\mathbf{x}_1, \mathbf{x}_2\}$, $\underline{Cl}_2^{\leqslant} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$. Observe that $\underline{Cl}_1^{\geqslant}$ and $\underline{Cl}_3^{\leqslant}$ are always trivial (contain all objects).

Rule induction (DOMLEM). Having obtained the lower approximations of class unions, we use them to learn a classifier to distinguish between $\underline{Cl}_k^{\geqslant}$ and $\underline{Cl}_{k-1}^{\leqslant}$, for each $k = 2, \ldots, K$. The classifier has the form of the set of decision rules, which generalize description of the information contained in the dataset D and serves as a basis for further classification of unseen objects. By *decision rule* we mean a simple expression of the form "if *condition*, then *decision*". In DRSA, only ordinal types of rules are considered, of the following form:

- if $x_{j_1} \geqslant s_{j_1}$ and … and $x_{j_q} \geqslant s_{j_q}$, then $y \geqslant k$,
- if $x_{j_1} \leqslant s_{j_1}$ and … and $x_{j_q} \leqslant s_{j_q}$, then $y \leqslant k$,
- if $x_{j_1} \geqslant s_{j_1}$ and … and $x_{j_q} \geqslant s_{j_q}$ and $x_{j_1} \leqslant s_{j_1}$ and … and $x_{j_q} \leqslant s_{j_q}$, then $y \in \{k', k' + 1, \ldots, k-1\}$

where $j_1, \ldots, j_p$ are indices of attributes which appear in the condition part, and $s_{j_1}, \ldots, s_{j_p}$ are some values from the domains of respective attributes. The first two types of rules are called *certain* while the rules of the third type are called *approximate*[3]. Certain rules are induced from lower approximations of the appropriate class unions (e.g. rules with decision part $y \geqslant k$ are induced from $\underline{Cl}_k^{\geqslant}$). Approximate rules are induced from the sets of the form: $B_{k',k} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \setminus \left(\underline{Cl}_{k'-1}^{\leqslant} \cup \underline{Cl}_k^{\geqslant}\right)$, $k \geqslant k'$, the boundaries between certain regions.

[3]There are in fact five types of rules, since first two types can also be induced from upper approximations and are called *possible* then.

Algorithm 1.1: DOMLEM

input : Family of sets $\mathcal{L} = \left\{\underline{Cl}_k^{\geqslant}, \underline{Cl}_{k-1}^{\leqslant}, B_{k',k} \colon 2 \leqslant k' \leqslant k \leqslant K\right\}$.
output: Set $\mathcal{R}$ of decision rules.

```
R := ∅;
for each L ∈ 𝓛 do
    R_L := ∅;          (rule set covering L)
    while L ≠ ∅ do
        Start with the rule with empty condition part, Φ = ∅;
        while Φ covers objects outside L do
            Evaluate each φ ∉ Φ by |cov(Φ∪φ)∩L| / |L|, where cov(Φ ∪ φ) is the set of objects
            covered by Φ ∪ φ;
            Choose a condition φ ∉ Φ with the highest evaluation;
            Φ := Φ ∪ φ;
        end
        Add rule with condition part Φ to R_L;
        Remove from L objects covered by Φ;
    end
    for each rule Φ in R_L do
        if Φ is minimal then
            R := R ∪ Φ;
        end
    end
end
return R;
```

There have been several algorithms proposed to induce decision rules within DRSA (Greco et al., 2001c; Pindur and Susmaga, 2003; Stefanowski and Żurawski, 2003; Błaszczyński and Słowiński, 2003; Dembczyński et al., 2003; Pindur et al., 2004). The most popular one, DOMLEM (Greco et al., 2001c), is based on the sequential covering procedure. It tends to generate the so called minimal set of rules (i.e. the non-redundant set of rules covering all objects) with the smallest number of rules. DOMLEM consists of a covering procedure run for each lower approximation and boundary in each subproblem. In the procedure, a set of rules is induced until it covers all objects from the respective approximation (or boundary). Rules are induced one by one, and in each rule the elementary conditions are added one by one (ordered by a specific evaluation function), until the rule covers only examples from the approximation. The scheme of DOMLEM is shown as Algorithm 1.1.

When classifying unseen object, each rule is tested whether it covers the object. Then, some procedure is used to combine the results of all covering rules and classify the object to a class (see e.g. Błaszczyński et al. (2007)).

1.3.2 Utility Functions in MCDA

Since the multicriteria sorting, considered within MCDA, coincides with ordinal classification with monotonicity constraints, potentially any method dealing with sorting can be regarded as a method for solving ordinal classification. However, many methods from MCDA do not suit to the approach used in this thesis, since:

- they construct the preference model using different forms of prior information (weights of

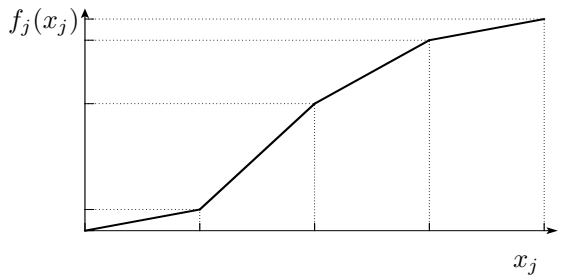

Figure 1.4: An example of piecewise linear function on attribute j with 3 break-points (and 2 boundary points).

criteria, preference thresholds, etc.) than the set of training examples, or

- they are interactive and demand the presence of decision maker during the learning process, or
- they are designed for smaller problems and scale badly with the problem size (both with n and m).

In this section we briefly describe methods based on the utility function models, such as the additive functions, Choquet or Sugeno integrals.

UTA methods. Originally, the UTA (UTilités Additives) method was first proposed by Jacquet-Lagrèze and Siskos (1982) for dealing with ranking problem. The UTADIS method (Jacquet-Lagrèze, 1995; Zopounidis and Doumpos, 1997; Siskos et al., 2005) is a variant of UTA for solving the multicriteria sorting. The values of the object on each attribute (criterion) are aggregated into an additive utility function:

$$f(\mathbf{x}) = \sum_{j=1}^{m} f_j(x_j),$$

where $f_j(x_j)$ are marginal utility functions. They are assumed to be piecewise linear consisting of the fixed number of break-points (see Figure 1.4). There are also $K+1$ thresholds $= -\infty = \theta_0 < \theta_1 < \ldots < \theta_{K-1} < \theta_K = \infty$ and it is assumed that if $\mathbf{x}$ is classified to the class k, i.e. $h(\mathbf{x}) = k$, if $\theta_{k-1} < f(\mathbf{x}) \leqslant \theta_k$. The construction of all of the marginal utility functions and thresholds is performed by solving a single linear programming problem.

For the purpose of robustness in the approach to ranking problems, Greco, Mousseau, and Słowiński (2008) proposed UTA^{GMS} method, which fixes a break-point for every value taken by any of the objects on each attribute. They showed that using such arepresentation, one can model every additive monotone function compatible with training examples. Dembczyński, Kotłowski, and Słowiński (2006b) extended this idea to classification and combine the approach with DRSA. They also introduced a specific penalty term which made the method similar to support vector machines (Vapnik, 1999).

Choquet and Sugeno integrals methods. Choquet integral (Choquet, 1953) and its ordinal counterpart, Sugeno integral (Sugeno, 1974), are much more powerful aggregation operators than additive value functions, because they can model interaction between the attributes (Grabisch, 1996). However, they have $2^m - 2$ parameters in general, therefore one usually restricts only to the interaction of the j-th order (for $j = 1$ we have an additive model). The number of parameters to estimate makes them rather impractical: for small datasets they are overparametrized and thus

tend to overfit, while for larger datasets the parameters' estimation becomes computationally very expensive.

Choquet integral has been applied in (Verkeyn et al., 2002) to survey data. It constitutes a main part of the sorting procedure implementation TOMASO (Tool for Ordinal Multi-Attribute Sorting and Ordering) (Marichal et al., 2005).

1.3.3 Isotonic Regression and Monotone Approximation

The statistical estimation and hypothesis testing in the presence of monotonicity constraints dates back to early 1950s and has been considered much before any other approach mentioned in this survey. The important contribution of this field is the algorithm of *isotonic regression*[4] (Brunk, 1955; Ayer et al., 1955).

Isotonic regression. Let $\succeq$ be a preorder relation, i.e. reflexive and transitive, on X (dominance relation in our case) and let $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a set of points in X. Let $\mathbf{y} = (y_1, \ldots, y_n)$ be a real-valued vector. A vector $\mathbf{p}^* = (p_1^*, \ldots, p_n^*)$ is an isotonic regression of $\mathbf{y}$ with weight vector $\mathbf{w} = (w_1, \ldots, w_n)$ if and only if $\mathbf{p}^*$ is the solution of the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} w_i (y_i - p_i)^2 \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow p_i \geqslant p_j \qquad \forall\, 1 \leqslant i, j \leqslant n, \end{aligned} \tag{1.19}$$

so that $\mathbf{p}^*$ minimizes the squared error in the space of all monotone vectors $\mathbf{p} = (p_1, \ldots, p_n)$. Isotonic regression is a solution to many order restricted statistical inference problems (Robertson et al., 1998). When $\succeq$ is a simple order (e.g. order between real numbers), an efficient (with complexity $O(n \log n)$) algorithm for solving (1.19) exists (Ayer et al., 1955), PAV (Pool Adjacent Violators). In a general case of any preorder, the problem can be solved in $O(n^4)$ (Maxwell and Muchstadt, 1985), which is impractical for larger datasets. However, several heuristics exist, in particular an effective $O(n^2)$ algorithm, which very often achieves the results close to optimal, was introduced by Burdakov et al. (2006).

Monotone approximation. Let us consider classification case with $y \in \{1, \ldots, K\}$. We state a problem similar to isotonic regression, but based on the minimization of arbitrary loss matrix:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} L(y_i, d_i) \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow d_i \geqslant d_j \qquad \forall\, 1 \leqslant i, j \leqslant n \\ & d_i \in \{1, \ldots, K\} \qquad \forall\, 1 \leqslant i \leqslant n. \end{aligned} \tag{1.20}$$

In other words, we would like to minimize a loss matrix within the class of all monotone vectors taking integer values. This is a nonparametric approach to ordinal classification, since we do not impose any additional condition, apart from the monotonicity. The problem has the following interpretation: relabel the objects to make the dataset monotone, such that new class labels are as close as possible to the original class labels, where the closeness is measured in terms of the loss function. The new labels in (1.20) are the values of variables d_i. The set of new labels will be called *monotone approximation* and we will refer to the problem (1.20) as *monotone approximation problem.*

[4]Sometimes *monotone regression* term is used; the word *isotonic* means monotone and increasing.

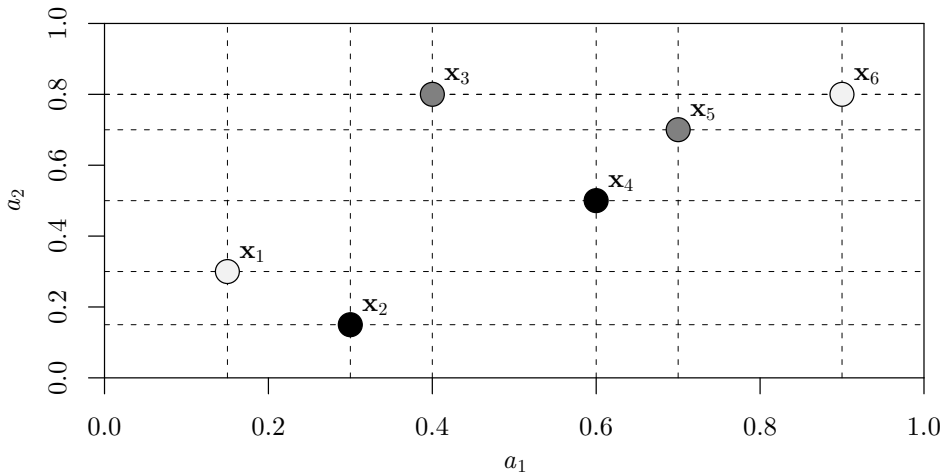

Figure 1.5: Example of three-class problem.

Dykstra et al. (1999) solved (1.20) for two particular loss functions: absolute error loss $L(y, d) = |y - d|$ and squared error loss $L(y, d) = (y - d)^2$. It was shown that the latter problem can be solved by isotonic regression, while the former can be solved by a sequence of $K - 1$ isotonic regression problems.

The problem for $K = 2$ and 0-1 loss appeared in the logical analysis of data, as a problem of finding monotone Boolean function approximating the data (Boros et al., 1995). It was solved in $O(n^3)$ by transformation to the maximum network flow problem, which is an improvement over (Dykstra et al., 1999).

Chandrasekaran et al. (2005) considered (1.20) in the most general form, for any K and any loss function, as a part of the *isotonic separation* procedure (see Section 1.3.8). It was shown that it can be solved either by linear programming or by maximum network flow in $O(n^3)$.

We were considering the problem of monotone approximation within the context of DRSA (Dembczyński et al., 2006a, 2007a,b).

The problem of monotone approximation has a drawback: in most cases the optimal solution is not unique. Thus, the particular solution found by the algorithms mentioned above is accidental (e.g. depends on the parameters of the solver used). We show in Chapter 3 how to cope with this issue.

Example. Consider the dataset from Table 1.1, shown in Figure 1.3. Assume the absolute error loss function. Then, there are two optimal solutions: either object $\mathbf{x}_4$ will be reassigned to class 2 or object $\mathbf{x}_5$ will be reassigned to class 3.

Consider one more example presented in Figure 1.5. This example is similar to the previous one, with exception that now $y_4 = 1$ and $y_1 = 3$. There is a unique monotone approximation problem with absolute error loss, which reassigns only a single object $\mathbf{x}_1$ to class 1.

1.3.4 Ordinal Learning Model

The *Ordinal Learning Model* (OLM) (BenDavid et al., 1989; Ben-David, 1992) was the first method for ordinal classification with monotonicity constraints proposed in the machine learning community. It is very simple and consists in choosing the subset $D' \subseteq D$ of training objects (so called "rule base"). Then, classification of new objects is done simply by a function of the form:

$$f_{\mathrm{OLM}}(\mathbf{x}) = \max\{y_i \colon \mathbf{x}_i \in D', \mathbf{x}_i \preceq \mathbf{x}\} \tag{1.21}$$

If the set over which the maximum is chosen is empty (i.e. there is no object from D' which is dominated by $\mathbf{x}$), then a class label is assigned by a nearest neighbor procedure using the Euclidean distance.

The rule base D' is chosen to be consistent and not to contain redundant objects. An object $\mathbf{x}_i$ is redundant in D' if there exists another object $\mathbf{x}_j$ such that $\mathbf{x}_i \succeq \mathbf{x}_j$ and $y_i = y_j$, i.e. $\mathbf{x}_i$ does not have any influence $f_{\mathrm{OLM}}(\mathbf{x})$. Set D' is constructed in the following way. First, all objects with the same attribute vector are replaced by one vector. The class label of new vector is the average of the class labels of replaced vectors. Then, we start with D' empty and objects are added to D' in the order of decreasing class labels. The particular order of objects within a given class is random. An object is added to D' only if it is consistent with objects already present in D'; otherwise it is rejected. Moreover, redundancy is checked each time: if the new object is redundant, then it is rejected; if the new object makes other objects in D' redundant, then those objects are rejected. The process is repeated until all the objects are examined.

The algorithm has several drawbacks. Firstly, the procedure of building D' relies on the random order of processing the objects, therefore can return different D' for each run. Moreover, the nearest neighbor rule is not an ordinal and can produce non-monotone results. The averaging of labels is also not ordinal operation. There are procedures for building the consistent subset which are deterministic and much better formally grounded, such as DRSA or monotone approximations, described in previous sections.

1.3.5 Monotone Neural Networks

Neural networks have received a great attention and popularity in the machine learning community during the late 1980s and the 1990s. Therefore, it is not surprising that early approaches to ordinal classification with monotonicity constraints were also focused on neural network models. Utilizing the domain knowledge about the monotonicity was done in one of two ways:

- by adding a second term measuring the "monotonicity error" (the extent to which the model violates monotonicity constraints) to the typical error measure (Sill and Abu-Mostafa, 1997),
- by enforcing some constraints on the weights and architecture of the network, which makes the network monotone by construction (Wang, 1994; Wang and Archer, 1994; Sill, 1998; Daniels, 1999).

Neural network models have been successfully applied to e.g. bond rating, house pricing (Daniels, 1999) or liver disorders diagnosis (Sill and Abu-Mostafa, 1997). However, their main drawback is the lack of interpretability, which impede the explanation of the model behavior. Moreover, neural networks work only with the cardinal scale on the attributes, so that they may lead to wrong or meaningless results if the attribute scale is purely ordinal.

1.3.6 Monotone Trees

Overview. Decision tree algorithms, such as CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993), are very popular in machine learning. They were also considered in the context of ordinal classification with monotonicity constraints. The first algorithm for tree induction in this context was proposed by Ben-David (1995) and called MID (Monotone Induction of Decision trees). The main idea was to modify the conditional entropy, by adding a term called order-ambiguity-score, which pushed the splitting strategy towards monotonicity. However, this did not guarantee the tree to be a monotone function.

A different approach was overtaken by Makino et al. (1999), which presented two algorithms: P-DT (Positive Decision Trees) and QP-DT (Quasi-Positive Decision Trees)[5]. The former constructs a monotone decision tree (i.e. a tree, which is a monotone function), while the latter – a so called quasi-monotone tree. Both algorithms work only with binary-class problems. They were adapted to handle K-class problems and extended in (Potharst et al., 1997; Potharst, 1999; Potharst and Bioch, 2000; Potharst and Feelders, 2002; Bioch and Popova, 2002), under the names MDT (Monotone Decision Trees) and QMDT (Quasi-Monotone Decision Trees).

Decision tree models have also been considered within the DRSA (Giove et al., 2002). In this approach, a cut minimizing the training error for a given class union is chosen for splitting. Three types of trees were considered: single-class (discriminating a single class union only), progressively-ordered (using a single class union, which is, however, progressively changing as the tree is growing) and full range (using all class unions simultaneously).

Cao-Van and De Baets (2003) and Cao-Van (2003) considered RT (Ranking Tree) algorithm for growing monotone decision trees. It differs from MDT in using an impurity measure based on the ranking error (number of reversed ranks) and in using a specific procedure for maintaining the monotonicity of the tree. It can handle inconsistent datasets (while MDT in its original version cannot).

MDT. MDT (Potharst and Bioch, 2000) worked originally with consistent (monotone) datasets only, however a modified version was then proposed to deal with inconsistent data (Bioch and Popova, 2002). It can use any of the popular impurity measures such as entropy or Gini index and the growth of the tree is done in a typical way. The main difference between MDT and ordinary tree induction method is the so called "cornering technique" for maintaining the monotonicity of a tree. It consists of adding artificial objects to each node, one in the lower left corner of the node, and another in the upper right corner (notice that each node in the tree represents a subset of X and has the form of hyperrectangle). The lower left object obtains the highest possible class label which does not violate consistency of the dataset, while the upper left object – lowest possible label, respectively. The tree is induced until every node corresponds to the objects from the same class (i.e. until each node is pure). It was shown that a tree generated in such s way is monotone.

1.3.7 Ordinal Stochastic Dominance Learner

Ordinal Stochastic Dominance Learner (OSDL) (Cao-Van, 2003; Cao-Van and De Baets, 2004) is an instance-based method for ordinal classification with monotonicity constraints. It is based on the dominance relation only. It constructs the estimate of the probability distribution function, $\hat{F}(\mathbf{x}) = \left(\hat{F}(\mathbf{x},1),\ldots,\hat{F}(\mathbf{x},K)\right) \in \mathbb{R}^K$, where $\hat{F}(\mathbf{x},k)$ is the estimate of $P(y \leqslant k|\mathbf{x})$. First, to each object $\mathbf{x}_i$ a distribution $\hat{f}(\mathbf{x}_i) = (\hat{f}(\mathbf{x}_i,1),\ldots,\hat{f}(\mathbf{x}_i,K))$ is assigned:

$$\hat{f}(\mathbf{x}_i,k) = \frac{|\{\mathbf{x}_j\colon \mathbf{x}_j = \mathbf{x}_i \wedge y_j \leqslant k, j = 1,\ldots,n\}|}{|\{\mathbf{x}_j\colon \mathbf{x}_j = \mathbf{x}_i, j = 1,\ldots,n\}|},$$

i.e. by counting objects with the same values of attributes. Notice that if each object lies in a different point in X, each distribution $\hat{f}(\mathbf{x}_i)$ consists of y_i ones and $K - y_i$ zeros. Next, for each $\mathbf{x} \in X$, we define two distribution estimators $\hat{F}_m(\mathbf{x})$ and $\hat{F}_M(\mathbf{x})$ as follows:

$$\begin{aligned}\hat{F}_m(\mathbf{x},k) &= \min\{\hat{f}(\mathbf{x}_i,k)\colon \mathbf{x}_i \preceq \mathbf{x}, i = 1,\ldots,n\}\\ \hat{F}_M(\mathbf{x},k) &= \max\{\hat{f}(\mathbf{x}_i,k)\colon \mathbf{x}_i \succeq \mathbf{x}, i = 1,\ldots,n\}.\end{aligned}$$

[5]In boolean reasoning, the term "positive" is sometimes used instead of "monotone".

If the dataset is consistent then $F_m(\mathbf{x}, k) \geqslant F_M(\mathbf{x}, k)$ for each $\mathbf{x} \in X$, $k = 1, \ldots, K$. In order to handle the inconsistencies, one also defines:

$$N_m(\mathbf{x}, k) = |\{\mathbf{x}_i \colon \mathbf{x}_i \preceq \mathbf{x} \wedge y_i > k\}|$$
$$N_M(\mathbf{x}, k) = |\{\mathbf{x}_i \colon \mathbf{x}_i \succeq \mathbf{x} \wedge y_i \leqslant k\}|.$$

The final estimate $\hat{F}(\mathbf{x})$ of the probability distribution depends on whether the situation at $\mathbf{x}$ is consistent or not, and is defined as:

$$\hat{F}(\mathbf{x}, k) = \begin{cases} (1-s)\hat{F}_m(\mathbf{x}, k) + s\hat{F}_M(\mathbf{x}, k) & \text{if } F_m(\mathbf{x}, k) \geqslant F_M(\mathbf{x}, k) \\ \frac{(1-s')N_m(\mathbf{x},k)\hat{F}_m(\mathbf{x},k) + s'N_M(\mathbf{x},k)\hat{F}_M(\mathbf{x},k)}{(1-s')N_m(\mathbf{x},i) + s'N_M(\mathbf{x},k)} & \text{otherwise.} \end{cases} \tag{1.22}$$

where $s, s' \in [0, 1]$ are the parameter of the algorithm (chosen by e.g. cross-validation). Having obtained the estimate of the probability distribution $\hat{F}(\mathbf{x})$, one can classify the object according to the form of the Bayes classifier for a given loss function, as described in Section 1.2 (for instance, for absolute error loss (1.9) we should choose the median). In (Cao-Van, 2003) the expectation value rounded to the closest integer is suggested (so, squared error loss (1.11) is implicitly used), although it is not a purely ordinal operation.

In (1.22), the upper expression is related to the consistent situation. The lower expression is more robust against the inconsistencies (it resembles the variable consistency model used in DRSA, see Chapter 4). For example, suppose there is an object $\mathbf{x}_i$ from the highest class, $y_i = K$, dominated by every other object, i.e. $\mathbf{x}_i \preceq \mathbf{x}$ for each $\mathbf{x} \in X$. Then, the upper expression in (1.22) would give $\hat{F}_m(\mathbf{x}, k) = 1$ for every k and for every $\mathbf{x}$ just because of the single troublesome object $\mathbf{x}_i$. The lower expression includes weighting by the number of objects $N_m(\mathbf{x}, k)$ and $N_M(\mathbf{x}, k)$ which reduces the influence of single, inconsistent objects (see Cao-Van (2003) for more details). Notice that the estimate (1.22) is the extension of the "max" formula (1.21) used in OLM.

Although the estimate (1.22) looks sensible, it has a drawback of being based only on the dominance relation. If there are many attributes, the dominance relation becomes sparse. Then, sets $\{\mathbf{x}_i \colon \mathbf{x}_i \succeq \mathbf{x}\}$ and $\{\mathbf{x}_i \colon \mathbf{x}_i \preceq \mathbf{x}\}$ become small, which leads to very poor estimates of probability distribution.

1.3.8 Isotonic Separation

Isotonic separation (Chandrasekaran et al., 2005) is a strong, novel tool for solving the problem of ordinal classification with monotonicity constraints. It has already been successfully applied to the firm bankruptcy prediction (Ryu and Yue, 2005), breast cancer diagnosis (Ryu et al., 2007), and Internet content filtering (Jacob et al., 2007).

The algorithm works with any loss function and consists of two stages. In the first stage, the dataset is "monotonized" (inconsistencies are removed) using the monotone approximation (1.20)[6]. The second stage is related to the classification of unseen objects. It is similar to the nearest-neighbors method, however it maintains monotonicity, i.e. the classifier is monotone. Let $(u)_+$ be a function which returns u if $u > 0$ and 0 otherwise, i.e. $(u)_+ = u1_{u>0}$. We define the "distance":

$$d(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{m} (x_j - x'_j)_+$$

Notice that if $\mathbf{x} \preceq \mathbf{x}'$ then $d(\mathbf{x}, \mathbf{x}') = 0$. Moreover, if $\mathbf{x} \succeq \mathbf{x}'$ then d is equivalent to the familiar L_1 metric. Let us denote $\theta(u) = 1_{u \geqslant 0}$ and let l_{yk} be a loss of classifying an object as k if the observed

[6] Of course, Chandrasekaran et al. (2005) do not use the term "monotonize" nor "monotone approximation" in their paper.

value is y. The classifier has the following form:

$$h_{\mathrm{IS}}(\mathbf{x}) = 1 + \sum_{k=2}^{K} \theta\Big(l_{k,k-1} \min\{d(\mathbf{x}, \mathbf{x}_i) \colon y'_i < k, i = 1, \ldots, n\} - l_{k-1,k} \min\{d(\mathbf{x}_i, \mathbf{x}) \colon y'_i \geqslant k, i = 1, \ldots, n\}\Big), \tag{1.23}$$

where y'_i are the new, consistent labels of objects (monotone approximation). First, notice that if $\mathbf{x}_i \succeq \mathbf{x}$ then $h(\mathbf{x}) \leqslant y'_i$, since then $d(\mathbf{x}, \mathbf{x}_i) = 0$ and in each term of the sum with $k > y'_i$ the argument of θ is always negative. Similarly, if $\mathbf{x}_i \preceq \mathbf{x}$ then $h(\mathbf{x}) \geqslant y'_i$. The particular class label is assigned by comparing the "distances" d to the "nearest neighbors" of $\mathbf{x}$.

The function (1.23) has a severe (and very strange) drawback of depending only on the values of the loss function above and below the diagonal, i.e. $l_{k,k-1}$ and $l_{k-1,k}$. Notice that, for 0-1 loss, absolute error loss and squared error loss, the expression (1.23) remains the same, which shows a very undesirable behavior in the context of Bayesian analysis considered in Section 1.2. Nevertheless, isotonic separation proved to be very effective in real-life applications.

The paper (Chandrasekaran et al., 2005) gives much broader analysis of the problem, including reduction of the problem size, speeding up the classification procedure, separation with "doubt regions" (where the classifier can abstain from the answer or indicate more than one class) and continuous outcome (regression) case.

1.4 Goal and Scope of the Thesis

This thesis is devoted to the ordinal classification with monotonicity constraints. The general goal of the thesis is the following:

> Provide a comprehensive statistical theory for the problem of ordinal classification with monotonicity constraints, as well as an efficient and accurate method for solving the problem.

In particular, there are four major objectives associated with this goal. These four objectives are achieved in separate chapters of the thesis. They are characterized briefly below.

Probabilistic model for ordinal classification with monotonicity constraints. Although ordinal classification with monotonicity constraints has been considered in multicriteria decision analysis, rough set approach and machine learning, there is no comprehensive theory which defines the problem from statistical point of view. We meet these needs in Chapter 2. We show, how monotonicity constraints can be expressed by making general assumptions about the probability distribution. Moreover, we formulate the necessary and sufficient conditions for the structure of the loss function which ensures the monotonicity of the optimal Bayes classifier.

Nonparametric methods. Having defined the model, we consider the probability estimation. We propose a nonparametric method, based on isotonic regression. Although isotonic regression has already been used to this end in binary-class case with linear preorder relation, our approach for any K and partial preorder is novel. Next, we analyze the problem of monotone approximation from statistical point of view. We also give a general method for reduction of the problem size. We thoroughly analyze the binary-class case and the case of a linear loss function. We show that the monotone approximation can be used as a general method for incorporating monotonicity constraints into the learning process.

Stochastic framework for Dominance-based Rough Set Approach. DRSA is the approach having its roots in logic, which is probably the reason why no statistical explanation of the approach has ever been proposed. In Chapter 4 we provide such an explanation basing on the statistical theory of ordinal classification with monotonicity constraints. The explanation leads us to a natural, stochastic extension of DRSA, which is found to be very useful for data in the presence of noticeable inconsistency. We also show that stochastic DRSA implicitly aims at minimizing a specific interval loss function. This makes the rough set methods perfectly tailored to the problems where abstaining from classification is allowed in some cases.

Monotone rule ensembles. Although the statistical theory for ordinal classification with monotonicity constraints provides an explanation for many concepts and approaches, it does not directly lead to the learning algorithm, which can be used for prediction purposes. In Chapter 5 we introduce two such algorithms, which both have the form of ensembles of decision rules. They possess a good prediction performance, low computational costs and maintain simplicity and interpretability. We provide a theoretical analysis which shows that the monotonicity assumptions allow us to bound the difference between the performance of rule ensemble and the performance of optimal classifier in terms of the empirically measurable value of the so called margin.

Computational experiments. The most of the thesis is theoretical. However, the learning algorithms demand empirical evaluation to test how they perform on real data. We address this issue in Chapter 6. We compare our algorithm based on monotone approximation, isotonic regression and rule ensembles with popular existing approaches to ordinal classification with monotonicity constraints. To our knowledge, there has not been such an extensive comparison of so many methods on so many datasets for classification problem with monotonicity constraints before.

Chapter 2

Probabilistic Model for Ordinal Classification with Monotonicity Constraints

Ordinal classification with monotonicity constraints is often referred to as the problem of finding an accurate classifier within the class of monotone functions. Restricting to such class of functions is justified either by the assumption that the "target" function is monotone, or by requirement that the constructed model should maintain monotonicity.

None of these statements, however, is applicable in the probabilistic setting, in which objects are generated according to some distribution and we do not take into account any semantic information about the data. Therefore, in this section we introduce a general assumption about the probability distribution, expressed in terms of the stochastic dominance in order to capture the concept of monotonicity constraints. Then we show how the monotonicity of the Bayes classifier follows for a specific class of loss function (Kotłowski and Słowiński, 2008).

2.1 Stochastic Dominance

We will formulate the most general assumption about the probability distribution $P(\mathbf{x}, y)$ when the monotonicity constraints are present in the ordinal classification. The monotonicity constraints require that if $\mathbf{x} \succeq \mathbf{x}'$ then $\mathbf{x}$ should be assigned a class not lower than $\mathbf{x}'$. In practice, these constraints are not always satisfied, leading to the situations referred to as inconsistencies. This suggests that the dominance relation $\succeq$ does not impose "hard" constraints and the constraints should rather be defined in a probabilistic setting.

Consider two points $\mathbf{x}, \mathbf{x}' \in X$, such that $\mathbf{x} \succeq \mathbf{x}'$. We believe that the core of the monotonicity concept consist in the observation that the probability that $\mathbf{x}$ will get higher class label then $\mathbf{x}'$, should not be smaller than the probability of the opposite event, i.e.:

$$P(y > y'|\mathbf{x}, \mathbf{x}') \geqslant P(y < y'|\mathbf{x}, \mathbf{x}') \tag{2.1}$$

In other words, the event that $\mathbf{x}$ gets a higher label than $\mathbf{x}'$ is more probable than the even that $\mathbf{x}'$ gets a higher label than $\mathbf{x}$. Moreover, this property should still hold if we merge some of the contiguous classes. For instance, suppose we have four classes in a house pricing dataset: "cheap", "moderate", "expensive" and "very expensive". If we merge classes "cheap" and "moderate" into a single class "not expensive", obviously, (2.1) should still hold. The intuition behind is that the

problem with merged contiguous classes maintains monotone properties (both, with respect to order on class labels and monotone relationships) and thus can still be regarded as ordinal classification with monotonicity constraints.

More formally, let $Y = \{1, \ldots, K\}$ be the original set of class labels and let $\tilde{Y} = \{1, \ldots, \tilde{K}\}$, with $\tilde{K} \leqslant K$, be a set of class labels which results from merging some of the contiguous classes, i.e. $\tilde{y} = 1 \iff y \in \{1, \ldots, k_1\}$, $\tilde{y} = 2 \iff y \in \{k_1 + 1, \ldots, k_2\}$, etc. We expect that for every $\mathbf{x} \succeq \mathbf{x}'$ and for every $\tilde{Y}$, (2.1) holds. We believe this is the minimal requirement for the probability distribution in the ordinal classification with monotonicity constraints.

It would be convenient, however, to have more comprehensive conditions than those described above. In particular, we would like to formulate the conditions expressed only by means of the original set of labels Y. Such conditions are introduced by the following theorem:

Theorem 2.1. *Let $\mathbf{x}, \mathbf{x}' \in X$ be such that $\mathbf{x} \succeq \mathbf{x}'$. Then, (2.1) holds for original set of class labels Y and for every set of class labels $\tilde{Y}$ which results from merging some of the contiguous classes if and only if for the original set Y it holds:*

$$P(y \leqslant k|\mathbf{x}) \leqslant P(y \leqslant k|\mathbf{x}') \tag{2.2}$$

for every $k = 1, \ldots, K$.

The proof of the theorem is quite technical and is given in the Appendix at the end of this chapter. Notice that (2.2) is a relation between two probability distributions, conditioned at $\mathbf{x}$ and $\mathbf{x}'$, respectively. This relation is known as *(first order) stochastic dominance* (Levy, 1998). Therefore, we define the following property of the probability distribution as the *principle of stochastic dominance* (Dembczyński et al., 2007b):

$$\mathbf{x} \succeq \mathbf{x}' \Longrightarrow P(y \leqslant k|\mathbf{x}) \leqslant P(y \leqslant k|\mathbf{x}') \qquad \forall \mathbf{x}, \mathbf{x}' \in X, k = 1, \ldots, K. \tag{2.3}$$

The principle can also be expressed in the following, equivalent way:

$$\mathbf{x} \succeq \mathbf{x}' \Longrightarrow P(y \geqslant k|\mathbf{x}) \geqslant P(y \geqslant k|\mathbf{x}') \qquad \forall \mathbf{x}, \mathbf{x}' \in X, k = 1, \ldots, K.$$

We will call a probability distribution to be *monotonically constrained* if it satisfies (2.3). The principle of stochastic dominance is the core of what we understand by monotonicity constraints. Notice that in (Cao-Van, 2003; Cao-Van and De Baets, 2004) stochastic dominance was also used, but to define the properties of the estimator, not the properties of the probability distribution.

Now we investigate the consequences of (2.3) for the Bayes classifier.

2.2 Monotonicity of the Bayes classifier

2.2.1 Loss functions and monotonicity of the Bayes classifier.

In the classification problem, we aim at finding the classifier which is as close as possible to the Bayes classifier, the best possible classifier. In other words, the Bayes classifier is our "target function" which we try to approximate. Therefore, no surprisingly, we require that in the ordinal classification with monotonicity constraints, the Bayes classifier must be monotone.

The Bayes classifier may not be unique. First of all, it is defined only up to a zero measure set. To solve this problem, we assume that for *every* $\mathbf{x} \in X$, the Bayes classifier returns the class label k with the smallest conditional risk $\mathbb{E}[L(y, k)|\mathbf{x}]$. Secondly, there can be ties between class labels on the conditional risk. Therefore, we assume the lowest label is always chosen. This makes the Bayes classifier unique.

Suppose that the probability distribution is monotonically constrained. We will investigate under what assumptions about the loss function, the Bayes classifier is a monotone function. Let $l_{yk} = L(y, k)$ be the loss for predicting class k when the actual class is y, and assume $l_{kk} = 0$ for each k, and $l_{yk} > 0$ for each $y \neq k$. We remind that the ordinal loss matrix was defined in Section 1.2.2 as:

$$\begin{aligned} l_{y,k-1} &\geqslant l_{yk} \qquad \text{if } k \leqslant y, \\ l_{yk} &\leqslant l_{y,k+1} \qquad \text{if } k \geqslant y. \end{aligned}$$

Those properties of the loss matrix are not sufficient to ensure monotonicity of the Bayes classifier when the probability distribution is monotonically constrained. We will prove this claim by a counter-example. Let us consider two popular loss matrices, 0-1 loss $l_{yk} = 1_{y \neq k}$ and absolute error loss $l_{yk} = |y - k|$. Both are ordinal (see Section 1.2.2), however the Bayes classifier is monotone only for absolute error loss. Indeed, the Bayes classifier for absolute error loss is the median of the distribution. It will be later shown that the median is a monotone function under stochastic dominance assumption. On the other hand, the Bayes classifier for 0-1 loss is the mode of the distribution (most probable class). Consider the following 3-class counter-example: assume that $\mathbf{x} \succeq \mathbf{x}'$ and that the probability distribution at $\mathbf{x}'$ is $(0.3, 0.3, 0.4)$ (e.g. $P(y = 3|\mathbf{x}') = 0.4$), while the probability distribution at $\mathbf{x}$ is $(0.1, 0.5, 0.4)$. Although (2.3) is satisfied (distribution is monotonically constrained), the Bayes classifier for $\mathbf{x}'$ indicates class 3, while for $\mathbf{x}$ it indicates class 2, which contradicts the monotonicity.

Thus, additional constraints must be imposed on the ordinal loss matrix in order to ensure the monotonicity of the Bayes classifier. This issue is solved by the following theorem:

Theorem 2.2. *Suppose $[l_{yk}]_{K \times K}$ is an ordinal loss matrix. Then the Bayes classifier is a monotone function for every monotonically constrained probability distribution $P(\mathbf{x}, y)$ if and only if the loss matrix satisfies the following constraints:*

$$\begin{aligned} l_{y,k+1} - l_{yk} &\geqslant l_{y+1,k+1} - l_{y+1,k} \qquad \textit{if } k > y, \\ l_{y,k-1} - l_{yk} &\geqslant l_{y-1,k-1} - l_{y-1,k} \qquad \textit{if } k < y. \end{aligned} \tag{2.4}$$

We give the proof in the Appendix due to its technical content. From Theorem 2.2 it follows that conditions (2.4) are necessary and sufficient for monotonicity of the Bayes classifier. The monotonicity property is required in the ordinal classification with monotonicity constraints, because otherwise there would be no point in restricting to the class of monotone functions. Hence, we will call the ordinal loss matrix satisfying (2.4) *monotone loss matrix*. Notice that for $K = 2$, every loss matrix is monotone.

Summarizing, we can define the ordinal classification problem with monotonicity constraints as the problem of minimizing the risk with the monotone loss matrix where the data are generated according to the monotonically constrained distribution.

2.2.2 Convex loss functions and monotonicity.

We will investigate the conditions (2.4) for a popular subclass of the loss matrices.

Let $c\colon \mathbb{Z} \to \mathbb{R}$ be a function which assigns to each integer[1] a real number. We say that the function $c(k)$ is *convex* if for each $k \in \mathbb{Z}$ it holds:

$$c(k) \leqslant \frac{c(k-1) + c(k+1)}{2}. \tag{2.5}$$

[1] We denote the set of integers by $\mathbb{Z}$.

This definitions is equivalent to the more familiar definition of the form: function $c(k)$ is convex if for every $i, j \in \mathbb{Z}$ and for every $\lambda \in [0,1]$ such that $\lambda i + (1-\lambda)j \in \mathbb{Z}$, we have:

$$c(i\lambda + (1-\lambda)j) \leqslant \lambda c(i) + (1-\lambda)c(j). \tag{2.6}$$

Here we use (2.5), due to its simplicity. The equivalence of definitions (2.5) and (2.6) is proved in the Appendix as Lemma 2.2. In multi-class problems, the loss matrix is very often expressed in the following form:

$$l_{yk} = c(y-k), \tag{2.7}$$

with $c(0) = 0$ and $c(k) > 0$ if $k \neq 0$. The matrices of this form are, for instance, 0-1 loss (with function $c(k) = 1_{k\neq 0}$), mean absolute error loss ($c(k) = |k|$) or squared error loss ($c(k) = k^2$). Moreover, every binary loss matrix is of that form, since it is determined by setting only two parameters, $l_{12} = c(-1)$ and $l_{21} = c(1)$. With such a representation of the loss matrices, we can prove the following theorem:

Theorem 2.3. *Suppose $[l_{yk}]_{K\times K}$ is an ordinal loss matrix of the form $l_{yk} = c(y-k)$ for some function $c\colon \mathbb{Z} \to \mathbb{R}$, such that $c(0) = 0$ and $c(k) > 0$ for every $k \neq 0$. Then, the Bayes classifier is monotone if and only if $c(k)$ is convex.*

Proof. The conditions (2.4) can now be expressed as:

$$\begin{aligned} c(y-k-1) - c(y-k) &\geqslant c(y-k) - c(y-k+1) \qquad \text{if } k > y, \\ c(y-k+1) - c(y-k) &\geqslant c(y-k) - c(y-k-1) \qquad \text{if } k < y, \end{aligned}$$

which are equivalent (along with condition $c(0) \leqslant \frac{c(1)+c(k+1)}{2}$ holding for every loss matrix) to the condition (2.5). □

Corollary 2.4. *Consider loss function of the form $l_{yk} = |y-k|^p$, for $p \geqslant 0$ and $K > 2$. Then the Bayes classifier is monotone if and only if $p \geqslant 1$.*

Proof. For any k, expression (2.5) is satisfied if and only if $p \geqslant 1$. This follows from the well known fact that functions of the form $c(x) = |x|^p$ are convex if and only if $p \geqslant 1$. □

Corollary 2.4 explains why 0-1 loss ($p \to 0$) does not lead to the monotone Bayes classifier under the stochastic dominance assumption, while absolute error loss ($p = 1$) and squared-error loss ($p = 2$) do ensure monotonicity. These results suggest that 0-1 is not a proper loss for ordinal classification with monotonicity constraints (apart from binary-class case $K = 2$), since it is not a monotone loss function, according to the definition (2.4). The absolute error loss is the "boundary" function – it just satisfies the minimal requirements for the loss function to be monotone.

2.3 Linear Loss

Definition. Let us consider a specific loss function known as *the linear loss* (Berger, 1993):

$$l_{yk} = \begin{cases} \alpha(k-y) & \text{if } k > y \\ (1-\alpha)(y-k) & \text{if } k \leqslant y, \end{cases} \tag{2.8}$$

where $0 < \alpha < 1$. For $\alpha = \frac{1}{2}$ we have absolute error loss $l_{yk} = |k-y|$ (up to proportional constant). The purpose of introducing (2.8) is to model asymmetric costs of misclassification: for $\alpha > \frac{1}{2}$ the cost of predicting higher class then the actual class y is more penalized than predicting the lower class; for $\alpha < \frac{1}{2}$ we have the opposite case. Such a loss function can be useful e.g. in medicine: consider

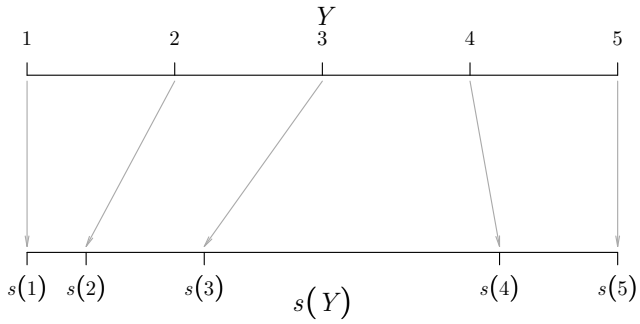

Figure 2.1: Example of extended linear loss with $K = 5$. The function $s(k)$ changes the position of each class label on the scale.

classifying patients into the classes according to their health condition: "good", "moderate", "bad", "very bad". Then, classifying the patient's condition to be better than it really is, is probably more dangerous to her/his health than regarding the patient to be in a worse condition than the real one.

As a corollary from Theorem 2.3, we immediately have:

Corollary 2.5. *The linear loss is a monotone loss matrix.*

It is also known (Berger, 1993) that such a loss function is minimized by the $(1-\alpha)$-quantile of the conditional distribution[2] i.e. by such $y_{1-\alpha}$ that $P(y \leqslant y_{1-\alpha}) \geqslant 1-\alpha$ and $P(y \geqslant y_{1-\alpha}) \geqslant \alpha$. For $\alpha = \frac{1}{2}$ we have the of median.

Extended linear loss. One can even extend the definition of linear loss by introducing the following definition:

$$l_{yk} = \begin{cases} \alpha(s(k) - s(y)) & \text{if } k > y \\ (1-\alpha)(s(y) - s(k)) & \text{if } k \leqslant y, \end{cases} \tag{2.9}$$

where $s(k)\colon Y \to \mathbb{R}$ is a strictly increasing function. It can be interpreted as a "scale changing" function, which positions the class labels on the real axis, thus changing the distances between them. Introducing arbitrary scale may at first look like a strong generalization of (2.8). However, it is not hard to show that the arbitrary scale will not change anything on the population level:

Theorem 2.6. *The Bayes classifier for the extended linear loss (2.9) does not depend on the function $s(k)$. In particular, this implies that the Bayes classifier for extended linear loss is the $(1-\alpha)$-quantile of the conditional distribution.*

Proof. First, notice that since $s(k)$ is strictly increasing, it has an inverse $s^{-1}\colon s(Y) \to Y$. Let $y \in Y$ be a random variable according to distribution $P(y|\mathbf{x})$. Let us define the random variable $y' = s(y)$. Moreover, for each $k \in Y$, let $k' = h(k)$. Then,

$$\arg\min_k \mathbb{E}\big[L(s(y), h(k))|\mathbf{x}\big] = s^{-1}\left(\arg\min_{k'} \mathbb{E}\big[L(y', k')|\mathbf{x}\big]\right), \tag{2.10}$$

i.e. minimizing the extended loss (2.9) is equivalent to first minimizing the expected loss (2.8) with random variable y' and then taking the s^{-1}-inverse of the obtained minimizer. The Bayes classifier for (2.8) is $y'_{1-\alpha}$, the $(1-\alpha)$-quantile of distribution $P(y'|\mathbf{x})$. Since $P(y' = k|\mathbf{x}) = P(y = s^{-1}(k)|\mathbf{x})$ for every $k = 1, \ldots, K$, then we must have $y'_{1-\alpha} = s(y_{1-\alpha})$. According to (2.10), the Bayes classifier for (2.9) is $s^{-1}(y'_{1-\alpha}) = s^{-1}(s(y_{1-\alpha})) = y_{1-\alpha}$. □

[2]We remind that in general, p-quantile of probability distribution $P(x)$ is defined as a value x_p such that $P(x \leqslant x_p) \geqslant p$ and $P(x \geqslant x_p) \geqslant 1-p$.

We conclude that there is no point in using complex scale changing functions and we can stay with the ordinary linear loss. Although parametrized by only a single value α, linear loss is sufficient to model most of real-life ordinal classification problems. Hence, for the rest of this thesis we will focus on the linear loss function.

Appendix: Proofs of the Theorems

Proof of Theorem 2.1

Theorem 2.1. *Consider* $\mathbf{x}$ *and* $\mathbf{x}'$ *such that* $\mathbf{x} \succeq \mathbf{x}'$. *Then, (2.1) holds for the original set of class labels* Y *and for every set of class labels* $\tilde{Y}$ *which results from merging some of the contiguous classes if and only if for the original set* Y *it holds:*

$$P(y \leqslant k|\mathbf{x}) \leqslant P(y \leqslant k|\mathbf{x}') \tag{2.11}$$

for every $k = 1, \ldots, K$.

Proof. First we prove the "if" part. Assume (2.11) holds for Y. But then (2.11) also holds for every merged set $\tilde{Y}$, since the event $\{\tilde{y} \leqslant k\}$ is equivalent to the event $\{y \leqslant k'\}$ for some k'. Choose any such set $\tilde{Y}$ and let us denote $\Delta = P(\tilde{y} \geqslant \tilde{y}'|\mathbf{x}, \mathbf{x}') - P(\tilde{y} \leqslant \tilde{y}'|\mathbf{x}, \mathbf{x}')$. We shall prove that $\Delta \geqslant 0$. Let us denote $p_k = P(\tilde{y} = k|\mathbf{x})$ and $q_k = P(\tilde{y} = k|\mathbf{x}')$ for each $k = 1, \ldots, \tilde{K}$. Due to the independence of y and y', we have:

$$P(\tilde{y} \geqslant \tilde{y}'|\mathbf{x}, \mathbf{x}') = \sum_{k=2}^{\tilde{K}} \sum_{l=1}^{k-1} P(\tilde{y} = k \wedge \tilde{y}' = l|\mathbf{x}, \mathbf{x}') = \sum_{k=2}^{\tilde{K}} \sum_{l=1}^{k-1} p_k q_l,$$

and, similarly, $P(\tilde{y} \leqslant \tilde{y}'|\mathbf{x}, \mathbf{x}') = \sum_{k=2}^{\tilde{K}} \sum_{l=1}^{k-1} q_k p_l$, so that:

$$\Delta = \sum_{k=2}^{\tilde{K}} \sum_{l=1}^{k-1} (p_k q_l - q_k p_l). \tag{2.12}$$

We will prove $\Delta \geqslant 0$ by induction on $\tilde{K}$. For $\tilde{K} = 1$ obviously $\Delta = 0$. Now, fix some $\tilde{K}$ and assume the theorem holds for $\tilde{K} - 1$. From (2.11) we have $p_{\tilde{K}} \geqslant q_{\tilde{K}}$, because:

$$p_{\tilde{K}} = 1 - \sum_{k=1}^{\tilde{K}-1} p_k \geqslant 1 - \sum_{k=1}^{\tilde{K}-1} q_k = q_{\tilde{K}}.$$

Notice that if we keep $p_{\tilde{K}} + p_{\tilde{K}-1}$ constant, but decrease $p_{\tilde{K}}$ and increase $p_{\tilde{K}-1}$ by the same amount ϵ, Δ decreases; one can easily check by simple analysis of the sums in (2.12) that the total decrease is $\epsilon(q_{\tilde{K}-1} + q_{\tilde{K}})$. We can decrease $p_{\tilde{K}}$ without violating (2.11) as long as $p_{\tilde{K}} \geqslant q_{\tilde{K}}$. Thus, it is enough to show that $\Delta \geqslant 0$ for $p_{\tilde{K}} = q_{\tilde{K}}$.

We transform the expression (2.12) to obtain:

$$\Delta = \sum_{k=2}^{\tilde{K}-1} \sum_{l=1}^{k-1} (p_k q_l - q_k p_l) + \sum_{l=1}^{\tilde{K}-1} (p_{\tilde{K}} q_l - q_{\tilde{K}} p_l). \tag{2.13}$$

The second term can be transformed to:

$$\sum_{l=1}^{\tilde{K}-1} (p_{\tilde{K}} q_l - q_{\tilde{K}} p_l) = p_{\tilde{K}}(1 - q_{\tilde{K}}) - q_{\tilde{K}}(1 - p_{\tilde{K}}) = p_{\tilde{K}} - q_{\tilde{K}} \geqslant 0.$$

Thus, it is enough to show that the first term is nonnegative. But the first term looks almost like the case with $\tilde{K}-1$ classes – the only difference is that probabilities, both q_k and p_k do not sum to 1. But they both sum to the same number, $1-p_{\tilde{K}}$, since $p_{\tilde{K}}=q_{\tilde{K}}$. So, if we divide (rescale) the first term by $1-p_{\tilde{K}}$ (which will neither change the sign of Δ nor violate (2.11)), we have the case with $\tilde{K}-1$, so we can use the induction assumption and finally we obtain $\Delta \geqslant 0$.

Now, we prove the "only if" part. Assume (2.1) holds for every merged set $\tilde{Y}$. In particular, it holds when $\tilde{Y}=\{1,2\}$, such that $\tilde{y}=1 \iff y \leqslant k$ and $\tilde{y}=2 \iff y>k$; in other words, we merge classes $1,\ldots,k$ into a single class and merge classes $k+1,\ldots,K$ into another class. Let us denote $p=P(y\leqslant k|\mathbf{x})$ and $q=P(y\leqslant k|\mathbf{x}')$. Then, we have

$$0 \leqslant P(\tilde{y}\geqslant \tilde{y}'|\mathbf{x},\mathbf{x}')-P(\tilde{y}\leqslant \tilde{y}'|\mathbf{x},\mathbf{x}')=q(1-p)-p(1-q)=q-p,$$

which means:

$$P(y\leqslant k|\mathbf{x}) \leqslant P(y\leqslant k|\mathbf{x}').$$

Since k was chosen arbitrarily, this ends the proof. □

Proof of Theorem 2.2

Before we prove the theorem, we first need a simple lemma related to the stochastic dominance. The lemma is a basic result in decision theory, but we give the proof for clarity and completeness.

Lemma 2.1. *Let $\mathbf{x}\succeq\mathbf{x}'$ so that $P(y|\mathbf{x})$ stochastically dominates $P(y|\mathbf{x}')$. Let $z\colon Y\to\mathbb{R}$ be a non-increasing random variable. Then it holds:*

$$\mathbb{E}[z|\mathbf{x}] \leqslant \mathbb{E}[z|\mathbf{x}'], \tag{2.14}$$

i.e. the expected value of z according to distribution $P(y|\mathbf{x})$ is always smaller then the expected value of z according to $P(y|\mathbf{x}')$.

Proof. Let us denote $p_k=P(y=k|\mathbf{x})$, $q_k=P(y_k|\mathbf{x}')$ and $z_k=z(k)$. Then (2.14) can be rewritten in the following way:

$$\sum_{k=1}^{K} p_k z_k \leqslant \sum_{k=1}^{K} q_k z_k.$$

Let us denote the cumulative distribution by $P_k=\sum_{l=1}^{k} p_l$ and $Q_k=\sum_{l=1}^{k} q_l$ (we assume $P_0=Q_0=0$). From the stochastic dominance it follows that $P_k\leqslant Q_k$ for each k. Then,

$$\begin{aligned}\sum_{k=1}^{K} p_k z_k &= \sum_{k=1}^{K}(P_k-P_{k-1})z_k = \sum_{k=1}^{K} P_k z_k - \sum_{k=0}^{K-1} P_k z_{k+1} = \\ &= z_K + \sum_{k=1}^{K-1} P_k(z_k-z_{k+1}) \leqslant z_K + \sum_{k=1}^{K-1} Q_k(z_k-z_{k+1}) = \sum_{k=1}^{K} q_k z_k,\end{aligned} \tag{2.15}$$

where the inequality follows from the fact that $z_k-z_{k+1}\geqslant 0$ for each k. □

Now, we prove the main theorem:

Theorem 2.2. *Suppose $[l_{yk}]_{K\times K}$ is an ordinal loss matrix. Then the Bayes classifier is a monotone function for every monotonically constrained probability distribution $P(\mathbf{x},y)$ if and only if the loss matrix satisfies the following constraints:*

$$\begin{aligned} l_{y,k+1}-l_{yk} &\geqslant l_{y+1,k+1}-l_{y+1,k} \qquad \text{if } k>y \\ l_{y,k-1}-l_{yk} &\geqslant l_{y-1,k-1}-l_{y-1,k} \qquad \text{if } k<y \end{aligned} \tag{2.16}$$

Proof. First we prove the "if" part. Suppose conditions (2.16) hold. Let us define δ_{yk} in the following way:

$$\delta_{yk} = \begin{cases} l_{yk} - l_{y,k-1} & \text{for } k > y, \\ l_{yk} - l_{y,k+1} & \text{for } k < y. \end{cases}$$

Let $P(\mathbf{x}, y)$ be any monotonically constrained probability distribution and let $\mathbf{x}, \mathbf{x}' \in X$ be any two points such that $\mathbf{x} \succeq \mathbf{x}'$. Let us denote $p_k = P(y = k|\mathbf{x})$ and $q_k = P(y = k|\mathbf{x}')$. Let u be a predicted class label. The expected loss for u over the distribution $P(y|\mathbf{x})$ is as follows:

$$\mathbb{E}[L(y,u)|\mathbf{x}] = \sum_{y=1}^{K} p_y l_{yu} = \sum_{y=1}^{u-1} p_y \sum_{k=y+1}^{u} (l_{yk} - l_{y,k-1}) + \sum_{y=u+1}^{K} p_y \sum_{k=u}^{y-1} (l_{yk} - l_{y,k+1}),$$

or by using δ_{yk}:

$$\mathbb{E}[L(y,u)|\mathbf{x}] = \sum_{y=1}^{u-1} p_y \sum_{k=y+1}^{u} \delta_{yk} + \sum_{y=u+1}^{K} p_y \sum_{k=u}^{y-1} \delta_{yk}.$$

Consider $\Delta(u|\mathbf{x}) = \mathbb{E}[L(y,u+1)|\mathbf{x}] - \mathbb{E}[L(y,u)|\mathbf{x}]$, the difference between the expected losses for $u+1$ and u:

$$\begin{aligned} \Delta(u|\mathbf{x}) &= \sum_{y=1}^{u} p_y \sum_{k=y+1}^{u+1} \delta_{yk} + \sum_{y=u+2}^{K} p_y \sum_{k=u+1}^{y-1} \delta_{yk} - \sum_{y=1}^{u-1} p_y \sum_{k=y+1}^{u} \delta_{yk} - \sum_{y=u+1}^{K} p_y \sum_{k=u}^{y-1} \delta_{yk} = \\ &= \sum_{y=1}^{u} p_y \delta_{y,u+1} - \sum_{y=u+1}^{K} p_y \delta_{yu} \leqslant \sum_{y=1}^{u} q_y \delta_{y,u+1} - \sum_{y=u+1}^{K} q_y \delta_{yu} = \Delta(u|\mathbf{x}'), \end{aligned}$$

where the inequality comes from Lemma 2.1, since the function $z(1) = \delta_{1,u+1}, \ldots, z(u) = \delta_{u,u+1}, z(u+1) = -\delta_{u+1,u}, \ldots, z(K) = -\delta_{Ku}$ is non-increasing according to the assumptions (2.16). This means that the difference in expected loss for any two contiguous class labels $u+1$ and u does not increase as we move from $\mathbf{x}$ to $\mathbf{x}'$. But this means that the difference in expected loss between *any* class labels v and u does not increase.

Now, suppose v is a Bayes classifier for $\mathbf{x}'$, i.e.:

$$v = \arg\min_{k \in Y} \mathbb{E}[L(y,k)|\mathbf{x}'].$$

Choose some $u < v$. We have:

$$0 > \mathbb{E}[L(y,v)|\mathbf{x}'] - \mathbb{E}[L(y,u)|\mathbf{x}'] \geqslant \mathbb{E}[L(y,v)|\mathbf{x}] - \mathbb{E}[L(y,u)|\mathbf{x}],$$

which means that u cannot be the Bayes classifier for $\mathbf{x}$. Thus, the Bayes classifier must be monotone.

Now we prove the "only if" part. Suppose that one of the conditions (2.16) is violated; without loss of generality, we may assume that the first one is violated, i.e. $l_{y_0 k_0} - l_{y_0,k_0-1} < l_{y_0+1,k_0} - l_{y_0+1,k_0-1}$ for some $k_0 > y_0$. We shall find some probability distribution and objects $\mathbf{x} \succeq \mathbf{x}'$ such that the Bayes classifier violates monotonicity condition, i.e. $h^*(\mathbf{x}) < h^*(\mathbf{x}')$.

First, notice that we can set $P(y = k|\mathbf{x}) = 0$ for each $\mathbf{x} \in X$, for every class label $k \notin \{y_0, k_0, k_0 - 1\}$. This will effectively eliminate other classes (they never occur in the problem) so that we end up with three-class problem which is much easier to analyze than a general K-class problem. Therefore, without loss of generality, we assume the following loss matrix:

$$[l_{ij}]_{K \times K} = \begin{pmatrix} 0 & l_{12} & l_{13} \\ l_{21} & 0 & l_{23} \\ l_{31} & l_{32} & 0 \end{pmatrix}.$$

Without loss of generality we also assume that the violation of (2.16) has the form $l_{23} > l_{13} - l_{12}$ (another possibility is $l_{21} > l_{31} - l_{32}$, but the analysis would be analogical). First, we will construct a probability distribution $z = (z_1, z_2, z_3)$ and later from this distribution we will construct distributions at points $\mathbf{x}$ and $\mathbf{x}'$. We will choose distribution z so that the expected loss for predicting class 2 is equal to the loss for predicting 3 and smaller than for class 1, i.e.:

$$\mathbb{E}_z[L(y,3)] = \mathbb{E}_z[L(y,2)], \tag{2.17}$$

$$\mathbb{E}_z[L(y,3)] < \mathbb{E}_z[L(y,1)], \tag{2.18}$$

where we denoted the expectation over the distribution z by $\mathbb{E}_z$ This implies:

$$\begin{aligned} z_1 l_{13} + z_2 l_{23} &= z_1 l_{12} + z_3 l_{32}, \\ z_1 l_{13} + z_2 l_{23} &< z_2 l_{21} + z_3 l_{31}. \end{aligned}$$

Using $z_3 = 1 - z_2 - z_1$ and knowing that both $l_{32} + l_{13} - l_{12}$ and $l_{13} + l_{31}$ are positive, we can transform these expressions to:

$$z_1 = A - Bz_2, \qquad z_1 < C - Dz_2, \tag{2.19}$$

where:

$$A = \frac{l_{32}}{l_{32} + l_{13} - l_{12}}, \quad B = \frac{l_{23} + l_{32}}{l_{32} + l_{13} - l_{12}}, \quad C = \frac{l_{31}}{l_{31} + l_{13}}, \quad D = \frac{l_{23} + l_{31} - l_{21}}{l_{31} + l_{13}}.$$

Notice that $A, C > 0$ and $B > 1$. We will prove some more inequalities. First, we show that $B - D > 0$:

$$\begin{aligned} B - D > 0 \iff & (l_{23} + l_{32})(l_{31} + l_{13}) > (l_{32} + l_{13} - l_{12})(l_{23} + l_{31} - l_{21}) \\ \iff & (l_{31} - l_{32})(l_{23} - l_{13} + l_{12}) + l_{12}(l_{23} + l_{32}) \\ & + l_{21}(l_{32} + l_{13} - l_{12}) > 0, \end{aligned} \tag{2.20}$$

which holds, because all the terms in the last equation are positive. Next, we show that $BC - AD > 0$:

$$\begin{aligned} BC - AD > 0 \iff & (l_{23} + l_{32}) l_{31} - l_{32}(l_{23} + l_{31} - l_{21}) > 0 \\ \iff & l_{23}(l_{31} - l_{32}) + l_{32} l_{21} > 0, \end{aligned} \tag{2.21}$$

which holds, because, again, all the terms are positive. Finally, we must show that $A - C < B - D$:

$$\begin{aligned} & A - C < B - D \iff l_{32}(l_{31} + l_{13}) - l_{31}(l_{32} + l_{13} - l_{12}) \\ & -(l_{23} + l_{32})(l_{31} + l_{13}) + (l_{23} + l_{31} - l_{21})(l_{32} + l_{13} - l_{12}) < 0 \iff \\ & -l_{23}(l_{31} - k_{32}) - l_{21}(l_{13} - l_{12} + l_{32}) - l_{12} l_{23} < 0, \end{aligned} \tag{2.22}$$

which holds because, again, all the terms on the left hand side are negative.

Let us substitute the first expression in (2.19) into the second expression to obtain:

$$A - z_2 B < C - z_2 D \iff z_2(B - D) > A - C + \iff z_2 > \frac{A - C}{B - D},$$

because $B - D > 0$ from (2.20). We now must show that there exist z_1, z_2 such that $0 < z_1, z_2 < 1$, $z_1 + z_2 < 1$, $z_1 = A - Bz_2$, and $z_2 > \frac{A-C}{B-D}$. Fix $z_2 = \epsilon + \max\{0, \frac{A-C}{B-D}\}$. From (2.22) it follows that $\frac{A-C}{B-D} < 1$, thus we can always found positive ϵ such that $0 < z_2 < 1$. Moreover:

$$z_1 = A - Bz_2 = -B\epsilon + \min\left\{A, A - B\frac{A - C}{B - D}\right\} = -B\epsilon + \min\left\{A, \frac{BC - AD}{B - D}\right\},$$

and since $A > 0$ and it follows from (2.21) that $\frac{BC-AD}{B-D} > 0$, we have $z_1 > 0$. Moreover, since $A < 1$, for sufficiently small ϵ we have $z_1 < 1$. Finally, notice that:

$$z_1 + z_2 = A - (B-1)z_2 = -\epsilon(B-1) + \min\left\{A, A - (B-1)\frac{A-C}{B-D}\right\}$$

(we used the fact that $B > 1$), which means that $z_1 + z_2 < 1$ for sufficiently small ϵ. Thus, all requirements are satisfied for z_1, z_2, z_3 to be a probability distribution for which (2.17)-(2.18) hold.

Since the inequality in (2.18) is a strict inequality, it will be still satisfied for another probability distribution $q = (q_1, q_2, q_3)$ such that $q_1 = z_1 + \gamma$ and $q_2 = z_2 - \gamma$ and $q_3 = z_3$ with sufficiently small γ. Similarly to the way we got the first equation in (2.19) from (2.17), we can show that the following holds:

$$\mathbb{E}_q[L(y,3)] < \mathbb{E}_q[L(y,2)] \iff q_1 < A - Bq_2,$$
$$\mathbb{E}_q[L(y,3)] > \mathbb{E}_q[L(y,2)] \iff q_1 > A - Bq_2.$$

It follows for *positive* γ that:

$$q_1 = z_1 + \gamma = A - Bz_2 + \gamma < A - Bz_2 + B\gamma = A - B(z_2 - \gamma) = A - Bq_2,$$

which means that for distribution q, class label 3 have the lowest cost. Moreover, if we choose another distribution $p = (p_1, p_2, p_3)$ such that $p_1 = z_1 - \gamma, p_2 = z_2 + \gamma, p_3 = z_3$, for the same positive γ, we have:

$$p_1 = z_1 - \gamma = A - Bz_2 - \gamma > A - Bz_2 - B\gamma = A - B(z_2 + \gamma) = A - Bp_2,$$

which means that for distribution p class label 2 have the lowest cost. But distribution p stochastically dominates distribution q, since $p_1 = z_1 - \gamma < z_1 + \gamma = q_1$ and $p_2 = q_2$. Thus, we can choose any $\mathbf{x}$, $\mathbf{x}'$ such that $\mathbf{x} \succeq \mathbf{x}'$ and assign $P(y = k|\mathbf{x}') := q_k$, $P(y = k|\mathbf{x}) := p_k$ for each k, and from the above analysis it follows that $h^*(\mathbf{x}) = 2 < 3 = h^*(\mathbf{x}')$, a contradiction. □

Proof of Lemma 2.2

In this section we prove the equivalence of two definitions of convex functions.

Lemma 2.2. *For any function $c\colon \mathbb{Z} \to \mathbb{R}$ which assigns to each integer a real number, the two following conditions are equivalent:*

1. *For every $i, j \in \mathbb{Z}$ and for every $\lambda \in [0,1]$ such that $\lambda i + (1-\lambda)j \in \mathbb{Z}$:*

$$c(i\lambda + (1-\lambda)j) \leqslant \lambda c(i) + (1-\lambda)c(j). \tag{2.23}$$

2. *For every $k \in \mathbb{Z}$:*

$$c(k) \leqslant \frac{c(k-1) + c(k+1)}{2}. \tag{2.24}$$

Proof. If the function satisfies (2.23), than (2.24) holds by choosing $i = k-1$, $j = k+1$ and $\lambda = \frac{1}{2}$.

Now, suppose that (2.24) holds. We will show that for all positive integers r, s we have:

$$c(k) \leqslant \frac{s}{s+r}c(k-r) + \frac{r}{s+r}c(k+s). \tag{2.25}$$

We will prove this by induction on r and s. For $r = s = 1$ (2.25) holds by assumption. Suppose it holds for every $r, s \leqslant N$ and we will prove that it then holds also for $r, s \leqslant N+1$. Multiple application of (2.25), first with some r, s and then with $1, r$, leads to:

$$c(k) \leqslant \frac{s}{s+r}\left(\frac{r}{r+1}c(k-r-1) + \frac{1}{r+1}c(k)\right) + \frac{r}{s+r}c(k+s),$$

or equivalently, by rearranging the terms, to:

$$c(k) \leqslant \frac{s}{s+r+1} c(k-r-1) + \frac{r+1}{s+r+1} c(k+s),$$

which means that (2.25) holds for $r \leqslant N+1$ and $s \leqslant N$. Applying once more (2.25) with $s, 1$, analogously as above, shows that (2.25) holds for $r, s \leqslant N+1$. This finishes the induction proof of (2.25).

Choose any $i, j \in \mathbb{Z}$ and any $\lambda \in [0,1]$ such that $k = \lambda i + (1-\lambda) j \in \mathbb{Z}$. But then $i = k - r$ and $j = k - s$ for some positive r, s. This means that $\lambda = \frac{s}{s+r}$ and from (2.25) it follows that (2.23) holds, which ends the proof. $\square$

Chapter 3

Nonparametric Methods

In this chapter, we consider the methods of probability estimation (isotonic regression) and classification (monotone approximation) which are based solely on the assumption about the mononicity constraints of probability distribution (principle of stochastic dominance). No other assumptions are made and hence the only information about the objects being used is obtained through the dominance relation. Those methods are called nonparametric, because we infer about the model using the largest possible class of functions under the monotonicity assumption: the class of *all* monotone functions.

The nonparametric methods are especially useful when the dominance relation is sufficiently dense (the dimensionality m is not too high) or when the only available information about the objects is given by the dominance relation $\succeq$ (e.g. we have no informations about the attribute vectors). Moreover, as we shall see in Chapter 5, the nonparametric methods can be very useful in combination with some parametric classification procedures, constituting a two-phase learning algorithm.

3.1 Nonparametric Probability Estimation by Isotonic Regression

When the distribution is monotonically constrained, it is enough (under some additional, mild assumption described later) to estimate the probabilities in a nonparametric way. We propose such a nonparametric method, taking into account only the monotonicity constraints expressed by the dominance relation. Our method is based on isotonic regression. Although isotonic regression has already been used to this end in binary-class case with linear preorder relation(Robertson et al., 1998), our approach for any K and partial preorder is new (Dembczyński et al., 2007b; Kotłowski et al., 2008).

3.1.1 Maximum Likelihood Estimation

The most popular method of probability estimation in statistics is the *maximum likelihood estimation* (MLE). Let D be the observed data (training set) and let $\boldsymbol{\theta}$ be some vector of parameters which are to be estimated. Then, we define the MLE estimate of $\boldsymbol{\theta}$, denoted by $\hat{\boldsymbol{\theta}}$, as the value of $\boldsymbol{\theta}$, for which the probability of D (likelihood) is maximal:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} P(D|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}; D),$$

where $L(\boldsymbol{\theta}; D)$ is the likelihood function. One usually works with the negative log-likelihood $\ell(\boldsymbol{\theta}) = -\log L(\boldsymbol{\theta}; D)$, since the global minimum of ℓ and global maximum of L coincide, i.e. $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta})$.

In the ordinal classification with monotonicity constraints we do not impose any specific probabilistic model apart from the stochastic dominance assumption. Therefore, for each object $\mathbf{x}_i, i = 1, \ldots, n$, we can regard the probability distribution $P(y|\mathbf{x}_i)$ as a set of parameters to estimate. By denoting $p_{ik} := P(y = k|\mathbf{x}_i)$, the likelihood function have the form:

$$L(\mathbf{p}|D) = \prod_{i=1}^{n} p_{i,y_i},$$

so that negative log-likelihood is:

$$\ell(\mathbf{p}) = -\sum_{i=1}^{n} \log p_{i,y_i}. \tag{3.1}$$

The only constraint that we assume is the stochastic dominance principle (2.3):

$$\mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow \sum_{l=1}^{k} p_{il} \leqslant \sum_{l=1}^{k} p_{jl}, \tag{3.2}$$

for every $i, j = 1, \ldots, n$, and for each $k = 1, \ldots, K$. Therefore, the nonparametric problem of MLE is defined as the problem of minimizing (3.1) under the constraints (3.2), and the constraints $p_{ik} \geqslant 0$ and $\sum_{k=1}^{K} p_{ik} = 1$, for $i = 1, \ldots, n$, to ensure that the axioms of the probability distribution hold. This is a nonlinear optimization problem, with a convex objective function and linear constraints, so it can be solved quite efficiently by general constraint optimization algorithms (Bazaraa et al., 2006). However, the objective function is not strictly convex when $K > 2$, so that the problem may not have a unique solution. Hence, we will solve the MLE problem only for a binary-class case, while for a general multi-class case we will propose a different estimation method, based on the reduction to $K - 1$ binary-class problems.

3.1.2 Binary-class Problem and Isotonic Regression

Binary-class MLE. Let us restrict to the case of $K = 2$ and for the sake of clarity, we will use the set of class labels $Y = \{0, 1\}$. Let $p_i = P(y = 1|\mathbf{x}_i)$ so that $P(y = 0|\mathbf{x}_i) = 1 - p_i$. Then, the problem of MLE can be rewritten as:

$$\begin{aligned} \text{minimize:} \quad & -\sum_{i=1}^{n} \big(y_i \log p_i + (1 - y_i) \log(1 - p_i)\big) \\ \text{subject to:} \quad & x_i \succeq x_j \Longrightarrow p_i \geqslant p_j \qquad i, j = 1, \ldots, n \\ & 0 \leqslant p_i \leqslant 1 \qquad i = 1, \ldots, n \end{aligned} \tag{3.3}$$

Although the problem (3.3) remains nonlinear, it can easily be shown that the objective function is strictly convex now, i.e. the optimal solution is unique. At this moment, we can use a consistency property (1.13) to significantly reduce the size of the problem. We remind that object $\mathbf{x}_i$ is consistent if for every $j = 1, \ldots, n$, it holds: $\mathbf{x}_i \succeq \mathbf{x}_j \rightarrow y_i \geqslant y_j$ and $\mathbf{x}_i \preceq \mathbf{x}_j \rightarrow y_i \leqslant y_j$. The reduction procedure is based on the following theorem.

Theorem 3.1. *Let $\hat{\mathbf{p}} = (\hat{p}_1, \ldots, \hat{p}_n)$ be the optimal solution of (3.3). Then, $\hat{p}_i = y_i$ if and only if object $\mathbf{x}_i$ is consistent.*

Proof. We consider the case $y_i = 1$ (the case $y_i = 0$ is analogous). If $\mathbf{x}_i$ is consistent, then from the definition of consistency (1.13), there is no other object $\mathbf{x}_j$, such that $\mathbf{x}_j \succeq \mathbf{x}_i$ and $y_j = 0$. Thus,

for every $\mathbf{x}_j$, such that $\mathbf{x}_j \succeq \mathbf{x}_i$, $y_j = 1$ and y_j is also consistent (otherwise, due to transitivity of dominance, $\mathbf{x}_i$ would not be consistent). Thus, we can set $\hat{p}_j = 1$ for $\mathbf{x}_j$ and $\hat{p}_i = 1$ for $\mathbf{x}_i$, and these are the values that minimize the objective function in (3.3), while satisfying the constraints.

Now, suppose $\hat{p}_i = 1$ and assume the contrary, that $\mathbf{x}_i$ is not consistent, i.e. there exists $\mathbf{x}_j$, $\mathbf{x}_j \succeq \mathbf{x}_i$, but $y_j = 0$. Then, due to the monotonicity constraints in (3.3), $\hat{p}_j \geqslant \hat{p}_i = 1$, so $\hat{p}_j = 1$, and the objective in (3.3) equals to the infinity, which is surely not the optimal solution to the minimization problem (since at least one feasible solution $\hat{p} \equiv \frac{1}{2}$ with a finite objective value exists). □

Thus, only consistent objects have probability estimates equal to 1 or 0. We can set $\hat{p}_i = y_i$ for each consistent object $\mathbf{x}_i$ and optimize (3.3) only for inconsistent objects, which usually gives a large reduction of the problem size (number of variables). We have experienced on real data that in most cases at least $80\% - 90\%$ of variables (objects) are removed.

In the next paragraph we show that (3.3) has the same optimal solution as the isotonic regression problem.

Isotonic regression. We remind that a vector $\mathbf{p}^* = (p_1^*, \ldots, p_n^*)$ is an isotonic regression of $\mathbf{y}$ if $\mathbf{p}^*$ is the solution of the following problem:

$$\begin{aligned} \text{minimize:} \quad & \sum_{i=1}^{n} (y_i - p_i)^2 \\ \text{subject to:} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow p_i \geqslant p_j \qquad i, j = 1, \ldots, n, \end{aligned} \tag{3.4}$$

so that $\mathbf{p}^*$ minimizes the squared error in the space of all monotone vectors $\mathbf{p} = (p_1, \ldots, p_n)$. Comparing with definition (1.19) from Chapter 1, we set the weights $\mathbf{w} = (w_1, \ldots, w_n)$ to be equal to ones[1]. Although squared error seems to be arbitrarily chosen, it can be shown that minimizing many other error functions we get the same vector $\mathbf{p}^*$ as in the case of (3.4). In particular, we show that this property applies also to the objective function of the MLE problem:

Theorem 3.2. *(Robertson et al., 1998) Let $\mathbf{p}^*$ be an isotonic regression of $\mathbf{y}$. Then, $\mathbf{p}^*$ is also the optimal solution to the MLE problem (3.3), i.e. $\hat{\mathbf{p}} = \mathbf{p}^*$.*

The proof is based on the analysis from (Robertson et al., 1998) and can be found in the Appendix. To summarize, the problem of MLE for binary-class case (3.3) can be solved by the isotonic regression (3.4), because the optimal solutions of both problems coincide. The isotonic regression is easier to solve due to a simpler objective function (quadratic); e.g. Burdakov et al. (2006) proposed a heuristic algorithm giving results close to optimum in $O(n^2)$ (this is in fact the fastest possible time for the order-restricted inference, since the order relation is in general of size $O(n^2)$). Moreover, by using Theorem 3.1 the size of the problem is usually reduced several times before the optimization process has even begun.

A simple example of isotonic regression is shown in Figure 3.1.

3.1.3 Multi-class Problem

Description of the estimation method. In the multi-class case, the MLE estimation is no longer equivalent to the isotonic regression. Moreover, the MLE estimation is not strictly convex, which results in non-unique solution (Dembczyński et al., 2007b). There is no well-established method for probability estimation for the case of $K > 2$ ordered classes. Below, we propose such an approach, based on multiple isotonic regression.

[1] Nevertheless, all the results shown in this chapter are also valid for objects with arbitrary positive weights.

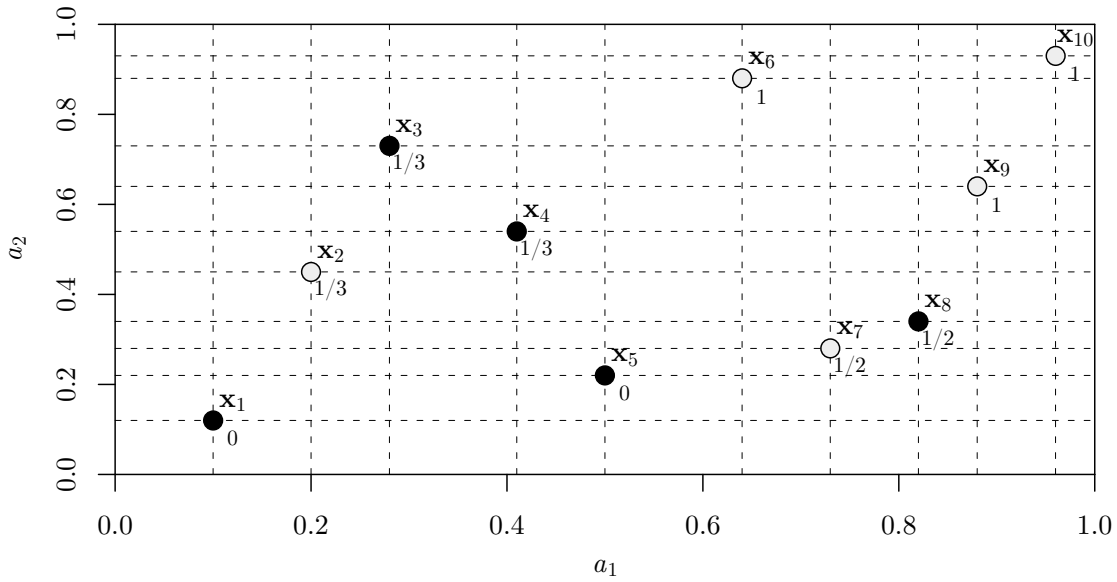

Figure 3.1: Binary-class example with two attributes. Objects with $y = 0$ are dark, while with $y = 1$ – light. The estimate of probability of class 1, $\hat{p}_i$, is shown. Notice that for consistent objects $(\mathbf{x}_1, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_9, \mathbf{x}_{10})$ it holds $y_i = \hat{p}_i$.

Let $Y = \{1, \ldots, K\}$ and for a given $\mathbf{x}_i$ let us define $K - 1$ dummy variables $y_{ik} = 1_{y_i \geqslant k}$, $k = 2, \ldots, K$. We can think of solving the general K-class problem in terms of solving $K - 1$ binary problems. In the k-th binary problem, dummy variables y_{ik} play the role of class labels with $Y = \{0, 1\}$, while variables of the problem correspond to estimating the probability $P(y \geqslant k|\mathbf{x}_i)$. Let us fix $k = 2, \ldots, K$. We define the vector of estimators $\hat{\mathbf{q}}_k = (\hat{q}_{1k}, \ldots, \hat{q}_{nk})$ of the probabilities $P(y \geqslant k|\mathbf{x}_i)$, as the isotonic regression of vector $\mathbf{y}_k = (y_{1k}, \ldots, y_{nk})$, i.e. the optimal solution to the problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} (y_{ik} - p_i)^2 \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow p_i \geqslant p_j \qquad i, j = 1, \ldots, n. \end{aligned} \tag{3.5}$$

Having obtained the solution of (3.5) for each $k = 2, \ldots, K$, we construct the estimators $\hat{p}_{ik}$ of $P(y = k|\mathbf{x}_i)$ as:

$$\hat{p}_{ik} = \begin{cases} \hat{q}_{ik} & \text{if } k = K \\ \hat{q}_{ik} - \hat{q}_{i,k+1} & \text{if } 2 \leqslant k < K \\ 1 - \hat{q}_{i,k+1} & \text{if } k = 1 \end{cases} \tag{3.6}$$

These estimators are unique because the isotonic regression is unique. They boil down to the previous approach (3.4) in binary-class case. However, as the $K - 1$ problems (3.5) are solved separately, we must guarantee that $\hat{q}_{ik} < \hat{q}_{i,k+1}$ will never happen, otherwise we would have negative probabilities $\hat{p}_{ik}$.

Properties of the estimators. We show that for each $i = 1, \ldots, n$, estimators $\{\hat{p}_{i1}, \ldots, \hat{p}_{iK}\}$ form a probability distribution, i.e. they are non-negative and sum to one. However, first we prove the following lemma:

Lemma 3.1. *Let $\hat{\mathbf{p}}$ be the isotonic regression of the class labels vector $\mathbf{y} = (y_1, \ldots, y_n)$. Suppose, we introduce a new vector of class labels $\mathbf{y}' = (y'_1, \ldots, y'_n)$, such that $y'_i \geqslant y_i$ for all $i = 1, \ldots, n$. Then, $\hat{\mathbf{p}}'$, the isotonic regression of $\mathbf{y}'$, has the following property: $\hat{p}'_i \geqslant \hat{p}_i$, for all $i = 1, \ldots, n$.*

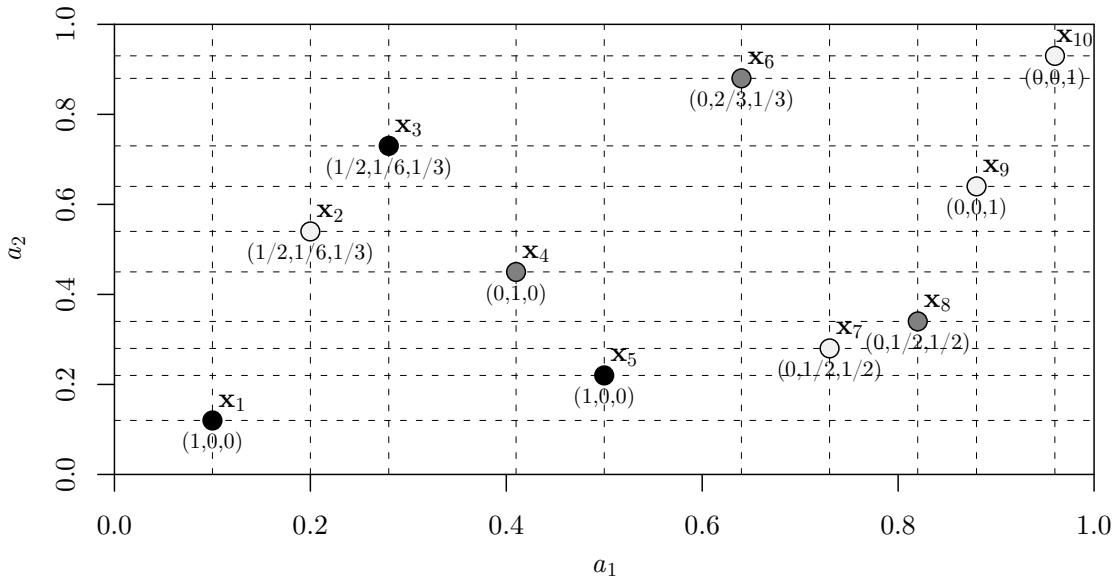

Figure 3.2: 3-class example with two attributes. Objects with $y = 1$ are black, with $y = 2$ – dark gray, and with $y = 3$ – light gray. The vector of probability estimates $(\hat{p}_{i1}, \hat{p}_{i2}, \hat{p}_{i3})$ is shown below each objects. Notice that for consistent objects $(\mathbf{x}_1, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_9, \mathbf{x}_{10})$ the probability concentrates on a single class y_i.

The proof can be found in the Appendix. Using Lemma 3.1, the desired properties of the estimators are now easy to show.

Theorem 3.3. *For each $i = 1, \ldots, n$, estimators $\{\hat{p}_{i1}, \ldots, \hat{p}_{iK}\}$ form a probability distribution, i.e. $\sum_{k=1}^{K} \hat{p}_{ik} = 1$, and for each k, $\hat{p}_{ik} \geqslant 0$.*

Proof. It immediately follows from the definition (3.6) that:

$$\sum_{k=1}^{K} \hat{p}_{ik} = 1 - \hat{q}_{i,2} + \sum_{k=2}^{K-1} \left(\hat{q}_{ik} - \hat{q}_{i,k+1}\right) + \hat{q}_{iK} = 1.$$

Now we prove the non-negativity of $\hat{p}_{ik}$. First, notice that the isotonic regression (3.4) is bounded between 0 and 1. This comes from the fact that the problem (3.4) has the same optimal solution as the problem (3.3), which explicitly includes the constraints $0 \leqslant p_i \leqslant 1$ (see Section 3.1.2). This shows that $\hat{p}_{i1} \geqslant 0$ and $\hat{p}_{iK} \geqslant 0$. To show that $\hat{p}_{ik} \geqslant 0$ for $k = 2, \ldots, K-1$, we must show that $\hat{q}_{ik} - \hat{q}_{i,k+1} \geqslant 0$. But this is guaranteed by Lemma 3.1, because class indices $y_{ik} = 1_{y_i \geqslant k}$ in the k-th problem are always greater than respective class indices $y_{i,k+1} = 1_{y_i \geqslant k+1}$. □

Summary. The problem of probability estimation for multi-class case is stated as the problem of solving $K - 1$ isotonic regression problems (3.5). The probability estimators are obtained from the optimal solution by (3.6). They always form a proper probability distribution, i.e. they are non-negative and sum to unity. Solving (3.5) in each case $k = 2, \ldots, K$, can be done in exactly the same way as in binary-class case (3.4). For instance, Theorem 3.1 also applies for each $k = 2, \ldots, K$.

A simple example of three-class problem is shown in Figure 3.2.

3.1.4 Extension Beyond the Training Set and Asymptotic Consistency

Until now, we dealt only with the estimation of the conditional probability distributions for objects from the training set D, i.e. at the points $\mathbf{x}_i$, $i = 1, \ldots, n$. However, one can simply extend the estimated probabilities to the whole space X.

Extension of probability estimates. Consider first the binary problem $Y = \{0, 1\}$ and probability estimate $\hat{p}_i$ for object $\mathbf{x}_i$. Since the estimates were obtained by solving the isotonic regression (3.4), it must hold $\mathbf{x}_i \succeq \mathbf{x}_j \rightarrow \hat{p}_i \geqslant \hat{p}_j$. The principle of stochastic dominance (2.3) for $K = 2$ says that the probability $p(\mathbf{x}) = P(y = 1|\mathbf{x})$ is a monotone function. Therefore, a valid extension $\hat{p}(\mathbf{x})$ of the vector of estimators $\hat{\mathbf{p}} = (\hat{p}_1 \ldots, \hat{p}_n)$ must satisfy two conditions:

1. $\hat{p}(\mathbf{x}) = \hat{p}_i$ (is the extension of the estimators).
2. For every $\mathbf{x}, \mathbf{x}' \in X$ it holds $\mathbf{x} \succeq \mathbf{x}' \rightarrow \hat{p}(\mathbf{x}) \geqslant \hat{p}(\mathbf{x})$ (is monotone).

Potharst and Feelders (2002) considered the extensions of monotone functions defined on the training set to the whole X. They showed that there is a minimal and a maximal extension, defined as:

$$\begin{aligned} \hat{p}^{\min}(\mathbf{x}) &= \max\{\hat{p}_i \colon \mathbf{x}_i \preceq \mathbf{x}\}, \\ \hat{p}^{\max}(\mathbf{x}) &= \min\{\hat{p}_i \colon \mathbf{x}_i \succeq \mathbf{x}\}, \end{aligned}$$

and every valid extension $\hat{p}(\mathbf{x})$ satisfies $\hat{p}^{\min}(\mathbf{x}) \leqslant \hat{p}(\mathbf{x}) \leqslant \hat{p}^{\max}(\mathbf{x})$ for every $\mathbf{x} \in X$. Moreover, every monotone function satisfying the above condition is a valid extension.

Therefore, it is worth considering the following parameterized extension:

$$p^{\lambda}(\mathbf{x}) = \lambda \hat{p}^{\min}(\mathbf{x}) + (1 - \lambda)\hat{p}^{\min}(\mathbf{x}), \tag{3.7}$$

for $\lambda \in [0, 1]$. The parameter λ can be tuned for a particular problem.

In multi-class case, the situation is analogous. Instead of $\hat{p}_i$, we have the estimators $\hat{q}_{ik}$ of $P(y \geqslant k|\mathbf{x}_i)$, for $k = 2, \ldots K$, obtained from (3.5). The stochastic dominance principle is equivalent to saying that $P(y \geqslant k|\mathbf{x}_i)$ must be a monotone function for each k. Therefore, the extensions $\{\hat{q}_2(\mathbf{x}), \ldots, \hat{q}_K(\mathbf{x})\}$ must be monotone. We can proceed analogously as before, defining the minimal, the maximal and λ-parametrized extension $\hat{q}_k^{\min}(\mathbf{x})$, $\hat{q}_k^{\max}(\mathbf{x})$ and $\hat{q}_k^{\lambda}(\mathbf{x})$.

Asymptotic consistency of the estimator. We say that the estimator $\hat{\theta}$ of parameter θ is *strongly consistent* if:

$$\hat{\theta} \xrightarrow[n \to \infty]{\text{a.s.}} \theta, \tag{3.8}$$

i.e. the estimator converges to the real value of the parameters almost surely (i.e. with probability 1), as n goes to infinity. We say that the estimator is weakly consistent, if (3.8) holds with convergence in probability.

Strong consistency of isotonic regression. Let us first assume $Y = \{0, 1\}$. To consider the asymptotic consistency of isotonic regression, let us extend the estimators of probabilities into the whole X using (3.7) for arbitrary λ. There is a large number of papers concerning the consistency of isotonic regression (see, for instance, Robertson and Wright (1975); Christopeit and Tosstorff (1987), or Robertson et al. (1998) for overview). In general, the isotonic regression, extended in the form of (3.7) for any λ, is strongly consistent for every point in the interior of X under the assumption that $P(y = 1|\mathbf{x})$ is a monotone function and under very mild additional assumption about probability distribution, e.g. if $P(\mathbf{x})$ is absolutely continuous with positive density (Christopeit and Tosstorff, 1987).

Now let us assume $Y = \{1, \ldots, K\}$. Suppose $P(\mathbf{x})$ is such that the binary isotonic regression is strongly consistent and let $P(\mathbf{x}, y)$ be monotonically constrained. Then, the strong consistency of the multiple isotonic regression $\hat{q}_k^{\lambda}(\mathbf{x})$ immediately follows from the definition (3.6) and from the fact that $P(y \geqslant k|\mathbf{x})$ is a monotone function for every $k = 2, \ldots, K$. In other words, the estimators

$\hat{q}_k^\lambda(\mathbf{x})$ converge (with probability 1) to the real probabilities as $n \to \infty$, as long as the isotonic regression is strongly consistent. Concluding, we are able to state the following theorem:

Theorem 3.4. *Let $P(\mathbf{x}, y)$ be monotonically constraint and let $P(\mathbf{x})$ be absolutely continuous with positive density. Then for each $\mathbf{x}$ in the interior of X, we have:*

$$\hat{q}_k^\lambda(\mathbf{x}) \xrightarrow[n\to\infty]{\text{a.s.}} P(y \geqslant k|\mathbf{x})$$

.

In other words, in practical (non-pathological) cases, the multiple isotonic regression estimators will converge to the true probabilities.

3.2 Monotone Approximation

Let us focus again on the ordinal classification problem. We will consider the *monotone approximation*, classification method based solely on the mononicity assumption of the probability distribution. The method consist in minimizing the empirical risk in the class of *all* monotone functions. We will also show that it can be though of as a general method for incorporating the monotonicity constraints into the learning process.

Although the problem of monotone approximation has already been considered several times in different contexts (see Section 1.3.3 for details), we are first to give the detailed analysis of its statistical properties, to show relationship to the isotonic regression and nonparametric maximum likelihood estimation and to summarize the asymptotic convergence issues (Dembczyński et al., 2007b; Kotłowski et al., 2008; Kotłowski and Słowiński, 2008). We also show how to reduce the computational complexity of the problem and how to handle non-uniqueness of the optimal solution.

3.2.1 Problem Statement

Problem formulation. The monotone approximation is based on relabeling objects from the training set in order to remove the inconsistencies and "monotonize" the data. Let us consider the minimization of the empirical risk (1.5) within the class of all monotone functions:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} L(y_i, d_i) \\ \text{subject to} \quad & x_i \succeq x_j \to d_i \geqslant d_j \qquad i, j = 1, \ldots, n \\ & d_i \in \{1, \ldots, K\} \qquad i = 1, \ldots, n, \end{aligned} \tag{3.9}$$

where $L(y, k)$ is any monotone loss function. In this section, we adopt the interpretation that optimal values $\hat{d}_i$ of the problem (3.9) are new labels of the objects. Then, the problem can be stated in the following way: reassign (relabel) the objects to make the dataset consistent (monotone) such that new class labels $\hat{d}_i$ are as close as possible to the original class labels y, where the closeness is measured in terms of the loss function. We believe the new labels $\hat{d}_i$ are closer to the Bayes classifier $h^*(\mathbf{x}_i)$ than the original, non-monotone labels y_i. Therefore, the monotone approximation can be thought of as the data "improvement" towards the Bayes classifier.

Algorithms for solving monotone approximation. Monotone approximation can be solved by either linear programming or network flow. In both cases, it must be transformed to a more useful form.

Let d_{ik}, for $k = 2, \ldots, K$, be a binary variable with the following interpretation "$d_{ik} = 1$ iff new class label of object $\mathbf{x}_i$ is at least k". Such interpretation implies that $d_{ik} \geqslant d_{i,k+1}$; for instance, for $K = 5$, $d_i = 1$ is decoded as $[d_{i1} = 0, d_{i2} = 0, d_{i3} = 0, d_{i4} = 0]$, d_2 is decoded as $[1, 0, 0, 0]$, d_3 as $[1, 1, 0, 0]$, d_4 as $[1, 1, 1, 0]$ and d_5 as $[1, 1, 1, 1]$. Thus, the new label of object $\mathbf{x}_i$ can be obtained from $d_i = 1 + \sum_{k=2}^{K} d_{ik}$. The monotonicity of new labels implies that for any $\mathbf{x}_i \succeq \mathbf{x}_j$ we must have $d_{ik} \geqslant d_{jk}$ for each $k = 2, \ldots, K$. Finally, the loss function can be reformulated as:

$$\begin{aligned} L(y_i, d_i) &= \sum_{k=2}^{y_i} (l_{y_i,k-1} - l_{y_i,k})(1 - d_{ik}) + \sum_{k=y_i+1}^{K} (l_{y_i,k} - l_{y_i,k-1}) d_{ik} \\ &= \sum_{k=2}^{K} (l_{y_i,k} - l_{y_i,k-1}) d_{ik} + \sum_{k=y_i+1}^{K} (l_{y_i,k-1} - l_{y_i,k}) \end{aligned} \tag{3.10}$$

and the second sum on the right-hand side can be dropped, because it is constant. Thus, we transformed problem (3.9) into the following problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} \sum_{k=2}^{K} (l_{y_i,k} - l_{y_i,k-1}) d_{ik} \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow d_{ik} \geqslant d_{jk} \qquad i, j = 1, \ldots, n; k = 2, \ldots, K, \\ & d_{i,k} \geqslant d_{i,k+1} \qquad i = 1, \ldots, n; k = 2, \ldots, K-1, \\ & d_{ik} \in \{0, 1\} \qquad i = 1, \ldots, n; k = 2, \ldots, K-1. \end{aligned} \tag{3.11}$$

This is a linear program with integer variables. However, the integer condition (last constraint) can be relaxed to $0 \leqslant d_{ik} \leqslant 1$ and we end up with an ordinary linear program. The relaxation of this constraint follows from the fact that the matrix of coefficients of the constraints is *totally unimodular* and the right hand sides of the constraints are integer. In such a case, every basic feasible (hence also optimal) solution is always integer. Therefore, we do need to impose integer constraints, because we will obtain an integer solution anyway (see Papadimitriou and Steiglitz (1998); Chandrasekaran et al. (2005) for more details).

Thus,(3.11) can be solved by linear programming. One can also show (Boros et al., 1995; Chandrasekaran et al., 2005) that this problem is a dual of the maximum network flow problem, so it can be solved in $O(n^3)$. However, linear program with an efficient solver was found to be a faster way to solve (3.11) on real datasets.

3.2.2 Reduction of the Problem Size

We provide in this section a general method for reduction of the size of the monotone approximation problem. A reduction method has been proposed for binary classification in (Chandrasekaran et al., 2005). We proposed a reduction method for 0-1 loss in multi-class problem in (Dembczyński et al., 2006a). Here we show how to reduce the problem (3.11) in the most general case.

Lower and upper labels. For each $\mathbf{x}_i$, let us define the *lower* and *upper class labels*, respectively as:

$$\begin{aligned} l_i &= \min\{y_j : \mathbf{x}_j \succeq \mathbf{x}_i, j = 1, \ldots, n\} \\ u_i &= \max\{y_j : \mathbf{x}_j \preceq \mathbf{x}_i, j = 1, \ldots, n\}. \end{aligned} \tag{3.12}$$

Lower and upper labels are the indicators of the inconsistencies in the dataset, as the following, simple lemma states:

Lemma 3.2. *Let l_i and u_i be the lower and upper class labels defined in (3.12). Then, we have:*

1. $l_i \leqslant y_i \leqslant u_i$.
2. $u_i = l_i$ *if and only if* $\mathbf{x}_i$ *is consistent.*
3. *For each* $\mathbf{x}_i \succeq \mathbf{x}_j$ *we have* $l_i \geqslant l_j$ *and* $u_i \geqslant u_j$.

Proof. We successively prove three parts of the theorem:

1. Since $\mathbf{x}_i \succeq \mathbf{x}_i$, y_i is in the set over which the minimum and maximum is taken in (3.12). This immediately implies $l_i \leqslant y_i \leqslant u_i$.
2. If $\mathbf{x}_i$ is consistent, then according to the definition under the equation (1.13), for every object $\mathbf{x}_j \succeq \mathbf{x}_i$ it must hold $y_j \geqslant y_i$. This implies that $l_i \geqslant y_i$ and from the property 1 we have $y_i = l_i$. Similarly, one can show that $y_i = u_i$.

   Assume $l_i = u_i$. This means that $y_i = l_i$, so for every object $\mathbf{x}_j \succeq \mathbf{x}_i$ it must hold $y_j \geqslant y_i$. From $y_i = u_i$ we conclude that for each object $\mathbf{x}_j \preceq \mathbf{x}_i$ it must hold $y_j \leqslant y_i$. Thus, $\mathbf{x}_i$ is consistent.
3. If $\mathbf{x}_i \succeq \mathbf{x}_j$, then $\{y_t \colon \mathbf{x}_t \succeq \mathbf{x}_i, t = 1, \ldots, n\} \subseteq \{y_t \colon \mathbf{x}_t \succeq \mathbf{x}_j, t = 1, \ldots, n\}$. This implies $l_i \geqslant l_j$, since the minimum of the subset must be greater than the minimum of the whole set. Analogously, one can show that $u_i \geqslant u_j$.

□

We will now prove one more lemma which will be needed in the construction of the reduction method:

Lemma 3.3. *Suppose $[l_{yk}]_{K \times K}$ is a monotone loss matrix, as defined in (2.4). Then, the loss function is strictly increasing, i.e.:*

$$\begin{aligned} l_{yk} &> l_{y,k+1} \qquad \text{if } k < y, \\ l_{y,k-1} &< l_{yk} \qquad \text{if } k > y. \end{aligned} \tag{3.13}$$

Proof. We will prove the first inequality in (3.13); the second one can be proved analogously. From (2.4) we have that for $k > y$, $l_{y,k+1} - l_{yk} \geqslant l_{y+1,k+1} - l_{y+1,k}$. Repeating this iteratively, we must finally get

$$l_{y,k+1} - l_{yk} \geqslant l_{y+2,k+1} - l_{y+2,k} \geqslant \ldots \geqslant l_{k,k+1} - l_{kk} > 0,$$

where the last inequality comes from $l_{kk} = 0$ and $l_{yk} > 0$ for $y \neq k$. □

Reduction procedure. The problem of monotone approximation, formulated in (3.11) has $n \times (K-1)$ variables. Removing any of those variables is desirable. Here we prove the theorem from which we know a priori the optimal values of some of the variables:

Theorem 3.5. *Let $\hat{d}_{ik}$, $i = 1, \ldots, n$, $k = 2, \ldots, K$, be any optimal solution to the problem (3.11). Let $\hat{d}_i = 1 + \sum_{k=2}^{K} \hat{d}_{ik}$. Then, we have $l_i \leqslant \hat{d}_i \leqslant u_i$.*

Proof. Since (3.11) is equivalent to (3.9), we will refer to the latter. Assume we have any optimal solution $\hat{d}_i$, $i = 1, \ldots, n$. Let I be a subset of those i for which $\hat{d}_i < l_i$. Similarly, let J be a subset of those i for which $\hat{d}_i > u_i$. Let us introduce solution $\tilde{d}_i$ such that $\tilde{d}_i = l_i$ for $i \in I$, $\tilde{d}_i = u_i$ for $i \in J$ and $\tilde{d}_i = \hat{d}_i$ otherwise. Since from Lemma 3.2 we know that $l_i \leqslant y_i \leqslant u_i$ and from Lemma 3.3 we know that the loss matrix is strictly increasing, it follows that $\tilde{d}_i$ has a lower objective value

that $\hat{d}_i$, if any of the sets I or J is nonempty. Indeed, for every $i \in I$ and every $i \in J$, the label $\tilde{d}_i$ is surely "closer" than $\hat{d}_i$ to the real label y_i. Therefore, it is enough to prove that the solution $\tilde{d}_i$ is feasible. Then, I and J must be empty, because otherwise it would contradict the optimality of $\hat{d}_i$.

To prove the feasibility of $\tilde{d}_i$ in the problem (3.9), we must show that:

$$\mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow \tilde{d}_i \geqslant \tilde{d}_j \quad i, j = 1, \ldots, n. \tag{3.14}$$

Notice that for $i \in I$, $\tilde{d}_i > \hat{d}_i$ and for $i \in J$, $\tilde{d}_i < \hat{d}_i$ Choose any $\mathbf{x}_i \succeq \mathbf{x}_j$. First we consider $i \in I$, then $i \in J$ and finally the case $i \notin I \cup J$:

1. Case $i \in I$. Then if $j \in I$, $\tilde{d}_i = l_i \geqslant l_j = \tilde{d}_j$. If $j \notin J$, $\tilde{d}_i > \hat{d}_i \geqslant \hat{d}_j \geqslant \tilde{d}_j$.
2. Case $i \in J$. Then $\tilde{d}_i = u_i \geqslant u_j \geqslant \tilde{d}_j$.
3. Case $i \notin I \cup J$. Then if $j \in I$, $\tilde{d}_i \geqslant l_i \geqslant l_j = \tilde{d}_j$. If $j \notin I$, $\tilde{d}_i = \hat{d}_i \geqslant \hat{d}_j \geqslant \tilde{d}_j$.

□

Theorem (3.5) implies that we can remove consistent objects from the optimization process, since we know a priori that $\hat{d}_i = y_i$ for such objects. Moreover, for other objects, we can fix some of d_{ik} to be 0 or 1 (and hence remove them from the optimization process) so that we ensure that $l_i \leqslant \hat{d}_i \leqslant u_i$ holds. This dramatically reduces the size of the problem: similarly to the isotonic regression, we have experienced from real data that in most cases more than $80 - 90\%$ of variables are removed.

3.2.3 Binary Monotone Approximation

Let us consider the simplest problem of monotone approximation, when $K = 2$ and $Y = \{0, 1\}$. The loss function has the form:

$$[l_y k]_{K \times K} = \begin{pmatrix} 0 & l_{01} \\ l_{10} & 0 \end{pmatrix},$$

and is always a monotone loss matrix. Let us denote $\alpha = \frac{l_{01}}{l_{01}+l_{10}}$. Then, one can easily show that the Bayes classifier $h^*(\mathbf{x})$ has one of the following forms:

$$\begin{aligned} h^*(\mathbf{x}) &= 1_{P(y=1|\mathbf{x}) \geqslant \alpha} \\ h^*(\mathbf{x}) &= 1_{P(y=1|\mathbf{x}) > \alpha}, \end{aligned}$$

or can be any monotone function between those two.

The monotone approximation problem (3.11) can be presented in the simplified form: since we have $K = 2$, we can omit the index k for the variables. Moreover, the objective function for a given i is $l_{01} d_i$ if $y_i = 0$ or $l_{10}(1 - d_i)$ if $y_i = 1$. It can be written more concisely as $w_{y_i} |y_i - d_i|$, where:

$$w_0 = \frac{l_{01}}{l_{01} + l_{10}} = \alpha \qquad w_1 = \frac{l_{10}}{l_{01} + l_{10}} = 1 - \alpha, \tag{3.15}$$

and such weights follow from dividing the loss function by $l_{01} + l_{10}$. Then, we can write (3.11) as:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} w_{y_i} |y_i - d_i| \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow d_i \geqslant d_j \qquad i, j = 1, \ldots, n. \end{aligned} \tag{3.16}$$

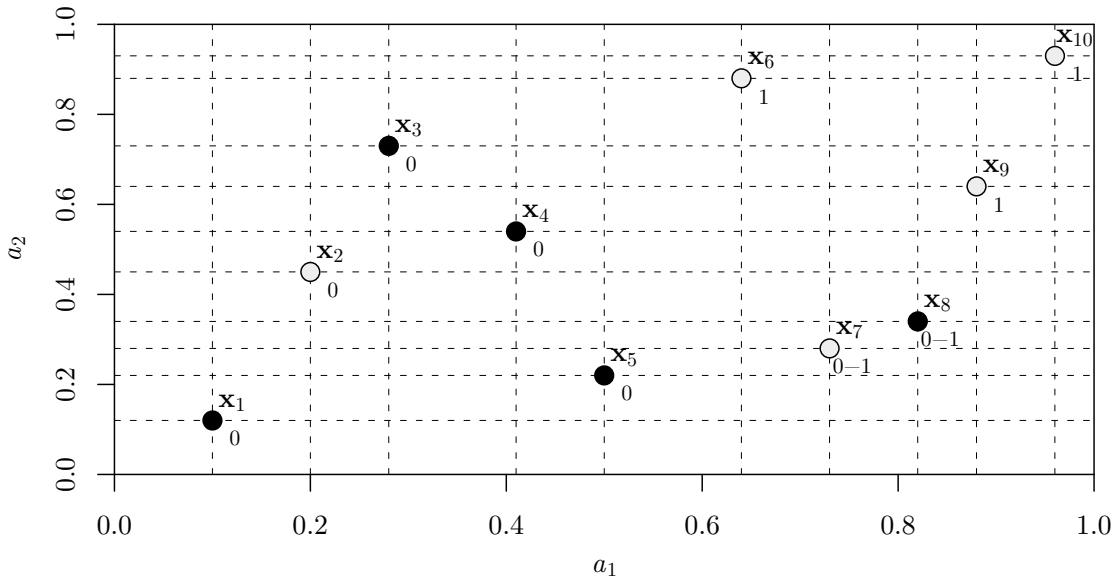

Figure 3.3: Binary-class monotone approximation with $\alpha = \frac{1}{2}$; the dataset is the same as in Figure 3.1. The new class label is shown below on the right of each object. The optimal solution is not unique; two objects have class labels $0-1$, which means that they are assigned label 0 in the lowest optimal solution and label 1 in the greatest optimal solution.

Notice that the relaxed constraint $0 \leqslant d_i \leqslant 1$ was dropped, because if there were any $d_i \geqslant 1$ (or $d_i \leqslant 0$) in any feasible solution, we could decrease their values down to 1 (or increase up to 0), obtaining a new feasible solution with smaller value of the objective function of (3.16).

We transformed the problem into (3.16) to show that it closely resembles isotonic regression (3.4). In the isotonic regression problem, we minimize L_2-norm (sum of squares) between vectors $\mathbf{y} = (y_1, \ldots, y_n)$ and $\mathbf{p} = (p_1, \ldots, p_n)$, while in (3.16) we minimize L_1-norm (sum of absolute values). In fact, both problems are closely connected, which is shown by the following theorem:

Theorem 3.6. *Suppose $\hat{\mathbf{p}} = (\hat{p}_1, \ldots, \hat{p}_n)$ is the optimal solution to the problem of isotonic regression (3.4). Then, the solution $\hat{\mathbf{d}}_* = (\hat{d}_{*1}, \ldots, \hat{d}_{*n})$ given by $\hat{d}_{*i} = 1_{\hat{p}_i > \alpha}$ for each $i = 1, \ldots, n$, and the solution $\hat{\mathbf{d}}^* = (\hat{d}^*_1, \ldots, \hat{d}^*_n)$ given by $\hat{d}^*_i = 1_{\hat{p}_i \geqslant \alpha}$ for each $i = 1, \ldots, n$, are the optimal solutions to the problem of binary monotone approximation (3.16) with weights (3.15).*

Moreover, if $\hat{\mathbf{d}} = (\hat{d}_1, \ldots, \hat{d}_n)$ is the optimal integer solution to the problem of binary monotone approximation, it must hold $\hat{d}_{*i} \leqslant \hat{d}_i \leqslant \hat{d}^*_i$, for all $i = 1, \ldots, n$. In particular, if $\hat{\mathbf{d}}_* = \hat{\mathbf{d}}^*$, then the solution to the binary monotone approximation problem is unique.

The proof can be found in the Appendix. Let us call $\hat{\mathbf{d}}^*$ the greatest, and $\hat{\mathbf{d}}_*$ the smallest optimal solutions. Theorem 3.6 states that if the MLE estimator (isotonic regression) $\hat{p}_i$ is greater (or smaller) than α, then the optimal value for the corresponding variable $\hat{d}_i$ in the binary monotone approximation problem (3.16) is 1 (or 0). The interest of this result lies in the fact that the functions $1_{\hat{p}_i \geqslant 1}$ and $1_{\hat{p}_i > 1}$ minimize the loss function on the training set D, while the functions $1_{P(y=1|\mathbf{x}_i) \geqslant 1}$ and $1_{P(y=1|\mathbf{x}_i) > 1}$ (Bayes classifiers) minimizes the expected loss (risk). Thus, the correspondence between probability estimates and the Bayes classifier estimate on D is the same as the correspondence between real probabilities and the real Bayes classifier.

We will often write α-binary monotone approximation, if we want to stress that the weights have the form $w_0 = \alpha$ and $w_1 = 1 - \alpha$.

Handling non-uniqueness of the solution. It follows from Theorem 3.6 that the monotone approximation may have non-unique solution. On the other hand, the isotonic regression is unique; moreover, if $\hat{p}_i > \frac{1}{2}$ then $\hat{d}_i = 1$, and if $\hat{p}_i < \frac{1}{2}$ then $\hat{d}_i = 0$. Therefore, the only non-unique variables in the monotone approximation are for those i, for which $\hat{p}_i = \frac{1}{2}$.

To investigate this issue in a greater detail, let us present a useful property of the isotonic regression. Suppose A is a subset of $\{1, \ldots, n\}$ and $\mathbf{f} = (f_1, \ldots, f_n)$ is a real-valued vector. We define $Av(f, A) = \frac{1}{|A|} \sum_{i \in A} f_i$ to be the average value of $\mathbf{f}$ on set A. Now suppose $\hat{\mathbf{p}}$ is the isotonic regression of $\mathbf{y}$. By a *level set* of $\hat{\mathbf{p}}$, denoted by $[\hat{p} = a]$, we mean the subset of $\{1, \ldots, n\}$ on which $\hat{\mathbf{p}}$ has constant value a, i.e. $[\hat{p} = a] = \{i \colon \hat{p}_i = a\}$. The following theorem holds:

Theorem 3.7. *(Robertson et al., 1998) Suppose $\hat{\mathbf{p}}$ is the isotonic regression of $\mathbf{y}$. If a is any real number such that the level set $[\hat{p} = a]$ is not empty, then $a = Av(\mathbf{y}, [\hat{p} = a])$.*

Theorem 3.7 states that for a given $\mathbf{x}_i$, $\hat{p}_i$ equals to the average of y_j over all the objects $\mathbf{x}_j$ having the same value $\hat{p}_j = \hat{p}_i$. Since there is a finite number of divisions of the dataset $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ into level sets, we conclude that there is a finite number of values that $\hat{\mathbf{p}}$ can possibly take. In our case, since $y_i \in \{0, 1\}$, all the values $\hat{p}_i$ must be of the form $\frac{r}{r+s}$, where r is the number of objects from class $y = 1$ in the level set, while s is the number of objects from class $y = 0$.

Using Theorem 3.7, we can construct the procedure of finding the greatest and the smallest optimal solutions in the binary case. First of all, notice that when α is *not* of the form $\frac{r}{r+s}$ for some integers $r, s \leqslant n$, then the binary monotone approximation is unique (there will be no $\hat{p}_i = \alpha$). On the other hand, if α *is* of the form $\frac{r}{r+s}$, let us increase α by sufficiently small ϵ, such that $\alpha + \epsilon$ is not of the form $\frac{r}{r+s}$ and there is no other number $\gamma = \frac{r}{r+s}$ for some $r, s \leqslant n$ such that $\alpha < \gamma < \alpha + \epsilon$. Then, the solution to the $(\alpha + \epsilon)$-binary monotone approximation is unique and is the same as the greatest solution to the α-binary monotone approximation, $\hat{\mathbf{d}}^*$, because there is no $\hat{p}_i$ such that $\alpha < \hat{p}_i \leqslant \alpha + \epsilon$. One can show that $\epsilon \leqslant n^{-2}$ is sufficient.

Similarly, decreasing α by ϵ will lead us to the smallest solution $\hat{\mathbf{d}}^*$. Thus, we have proved the following theorem.

Theorem 3.8. *If α is not of the form $\frac{r}{r+s}$ for some $r, s \leqslant n$, the α-binary monotone approximation is unique. Otherwise, the greatest α-binary monotone approximation $\hat{\mathbf{d}}^*$ can be found by increasing the value of α by $\epsilon \leqslant n^{-2}$ and solving the problem (3.16). Similarly, the smallest α-binary monotone approximation $\hat{\mathbf{d}}_*$ can be found by decreasing the value of α by $\epsilon \leqslant n^{-2}$ and solving again the problem (3.16).*

Summarizing, we see that finding the greatest and the lowest optimal solutions corresponds to solving the monotone approximation at most twice, with α slightly perturbed by $\pm\epsilon$, $\epsilon \leqslant n^{-2}$.

Example. A simple example of binary monotone approximation can be found in Figure 3.3. Comparison with Figure 3.1 shows the relationships between new labels and probability estimates, as stated in Theorem 3.6.

3.2.4 Linear Monotone Approximation

Monotone approximation with linear loss. Now we will investigate the problem of monotone approximation (3.11) with the extended linear loss matrix (2.9). We have already shown in Theorem 3.6 that there exists a correspondence between binary isotonic regression and binary monotone approximation. In the forthcoming theorem we will show that an analogous correspondence between multiple isotonic regression and monotone approximation with linear loss takes place. We will also show that both monotone approximation with extended linear loss and with the ordinary linear loss (2.8) lead to the same solution, similarly as it is on the population level (c.f. Theorem 2.6).

Theorem 3.9. *Let $\hat{\mathbf{q}}_k$ be the isotonic regression of $\mathbf{y}_k$, $k = 2, \ldots, K$, as defined in (3.5). Then, the solutions $\hat{\mathbf{d}}_* = (\hat{d}_{*1}, \ldots, \hat{d}_{*n})$ and $\hat{\mathbf{d}}^* = (\hat{d}_1^*, \ldots, \hat{d}_n^*)$ defined as:*

$$\hat{d}_{*i} = 1 + \textstyle\sum_{k=2}^{K} 1_{\hat{q}_{ik} > \alpha}, \qquad \hat{d}_i^* = 1 + \textstyle\sum_{k=2}^{K} 1_{\hat{q}_{ik} \geqslant \alpha}, \tag{3.17}$$

are the optimal solutions to the monotone approximation problem with extended linear-loss (2.9). Moreover, every other optimal solution $\hat{\mathbf{d}} = (\hat{d}_1, \ldots, \hat{d}_n)$ satisfies $\hat{d}_{*i} \leqslant \hat{d}_i \leqslant \hat{d}_i^*$, for each $i = 1, \ldots, n$.

Proof. Let us transform the objective function of the monotone approximation (3.11) for a given i, denoted by L_i:

$$\begin{aligned} L_i &= \sum_{k=2}^{K} (l_{y_i,k} - l_{y_i,k-1}) d_{ik} = \\ &= \sum_{k=2}^{K} \big(-(1-\alpha) y_{ik} (s(k) - s(k-1)) d_{ik} + \alpha (1 - y_{ik})(s(k) - s(k-1)) d_{ik} \big) \\ &= \sum_{k=2}^{K} (s(k) - s(k-1)) \big(-(1-\alpha) y_{ik} d_{ik} + \alpha (1 - y_{ik}) d_{ik} \big), \end{aligned}$$

where $y_{ik} = 1_{y_i \geqslant k}$, as usual. By adding the constant value (which does not change the optimization process) $\sum_{k=2}^{K} (s(k) - s(k-1)) y_{ik} (1-\alpha)$ we obtain:

$$\begin{aligned} L_i &= \sum_{k=2}^{K} (s(k) - s(k-1))(1-\alpha) y_{ik} (1 - d_{ik}) + \alpha (1 - y_{ik}) d_{ik} \\ &= \sum_{k=2}^{K} (s(k) - s(k-1)) w_{y_i k} |y_{ik} - d_{ik}|, \end{aligned}$$

where $w_0 = \alpha$ and $w_1 = 1 - \alpha$. Thus, the total loss has the form:

$$\sum_{k=2}^{K} (s(k) - s(k-1)) \left(\sum_{i=1}^{n} w_{y_{ik}} |y_{ik} - d_{ik}| \right). \tag{3.18}$$

For each k, the loss function looks exactly like the loss in the binary monotone approximation (3.16) (except the term $s(k) - s(k-1)$), where y_{ik} now plays the role of the binary class label. Unfortunately, those $K-1$ binary problems are not independent due to constraint $d_{ik} \geqslant d_{i,k+1}$ in (3.11), which involves the variables for different k. However, we will show that constraint $d_{ik} \geqslant d_{i,k+1}$ is not needed for linear loss and can be removed.

Indeed, we will show that the greatest optimal solutions to the $K-1$ binary problems solved independently still satisfy the constraint $d_{ik} \geqslant d_{i,k+1}$. Consider the optimal solution to the binary problems for some k and $k+1$. It follows that $y_{ik} \geqslant y_{i,k+1}$ for each i (because $1_{y_i \geqslant k} \geqslant 1_{y_i \geqslant k+1}$). Then, according to Lemma 3.1, the isotonic regression of $\mathbf{y}_k = (y_{1k}, \ldots, y_{nk})$, denoted by $\hat{q}_{ik}$, is greater than or equal to the solution to the isotonic regression of $\mathbf{y}_{k+1}$, denoted by $\hat{q}_{i,k+1}$. But then, from Theorem 3.6 it follows that the greatest optimal solution to the binary monotone approximation problem with y_{ik}, $\hat{d}_{ik}^* = 1_{\hat{q}_{ik} \geqslant \alpha}$ is greater than or equal to the greatest optimal solution to the binary monotone approximation problem with $y_{i,k+1}$, $\hat{d}_{i,k+1}^* = 1_{\hat{q}_{i,k+1} \geqslant \alpha}$. Thus, the constraint $d_{ik} \geqslant d_{i,k+1}$ is satisfied for the greatest optimal solution. One can similarly show the same for the smallest solution.

What we have proved above is that the solutions (3.17) composed of the greatest and the smallest optimal solutions to $K-1$ binary problems are *feasible* solutions of the original problem (i.e. linear monotone approximation (3.16)); but since the minimum of a more constrained problem cannot decrease, they are also the greatest and the smallest *optimal* solutions to the original problem. □

There are several important conclusions following from Theorem 3.9. First of all, the problem of monotone approximation (3.11) for extended linear loss can be solved by solving a sequence of $K-1$ simple weighted binary problems. Although it seems that we now have $K-1$ problem instead of one problem, from the computational point of view it is a great gain, because we reduced the problem with $(K-1) \times n$ variables to $K-1$ subproblems with n variables each, i.e. we decomposed a big problem into a sequence of smaller ones. This decomposition will be especially useful in Chapter 5, for dealing with rule ensemble models.

Moreover, a closer look at (3.17) reveals an interesting fact: for each $i = 1, \ldots, n$, every $\hat{d}_i$ such that $\hat{d}_{*i} \leqslant \hat{d}_i \leqslant \hat{d}_i^*$, is the $(1-\alpha)$-quantile of the probability distribution $\{\hat{p}_{i1}, \ldots, \hat{p}_{iK}\}$, obtained in (3.6) from the multiple isotonic regression. This corresponds exactly to the relationship between real probability distribution $\{p_{i1}, \ldots, p_{iK}\}$ and the Bayes classifier $h^*(\mathbf{x}_i)$. In other words, the relationship between the estimates on D is the same as the relationship between corresponding quantities on the population level.

The theorem also shows that there is no point in using sophisticated extended linear loss, because for every scale function $s(k)$, the solutions are identical. At first look, this seems to be quite counter-intuitive, since in medicine we could position class 1 ("ill") far away from class 2 ("healthy") and class 3 ("very healthy"), e.g. $s(1) = 0, s(2) = 0.8, s(3) = 1$. This would be done due to the fact that we do not care so much about the condition of the patient provided her or she is healthy. However, as the above analysis shows, this is pointless: we would obtain exactly the same results for e.g. $s(1) = 0, s(2) = 0.01$ and $s(3) = 1$. This shows the unusual property of the linear loss of being independent of a particular scale.

Finally, notice that similarly to the case of binary monotone approximation, we can give a simple procedure for finding the greatest and the smallest solutions. If α is *not* of the form $\frac{r}{r+s}$ for some $r, s \leqslant n$, then the linear monotone approximation is unique. Otherwise, we can increase α by $\epsilon \leqslant n^{-2}$ and solve the $K-1$ binary problems (3.16) to obtain the greatest linear monotone approximation. Similarly, we obtain the smallest linear monotone approximation if we decrease α by ϵ.

Example. Consider the example shown in Figure 3.4, illustrating how three-class problem is transformed to two binary problems. For simplicity we assume that $\alpha = \frac{1}{2}$, i.e. the loss function is an ordinary (symmetric) absolute error. Notice that for any object, the new class label in the top figure can be obtained from $\hat{d}_i = 1 + \sum_{k=2}^{K} \hat{d}_{ik}$, i.e. by summing up new class labels on middle and lower figures and adding 1. Moreover, new class labels on the middle figure are always greater or equal then those on the lower figure. This is exactly the conclusion of Theorem 3.9.

Remark. The monotone approximation with linear loss plays an important role in the subsequent chapters, therefore from now on we will often omit the term "linear loss" and call it simply monotone approximation or monotone α-approximation, if we want to stress that we are using particular value of asymmetry constant in (2.8).

3.2.5 Extension Beyond the Training Set and Asymptotic Consistency

Extensions of the monotone approximation. The monotone approximation (with linear loss) $\hat{\mathbf{d}}$ is defined only at training points $\mathbf{x}_i$, $i = 1, \ldots, n$. Since we are dealing with a class of all monotone functions, we can extend monotone approximation to X by using any monotone function $d\colon X \to Y$, such that $d(\mathbf{x}_i) = \hat{d}_i$, for each $i = 1, \ldots, n$. Similarly as in Section 3.1.4, we define the following

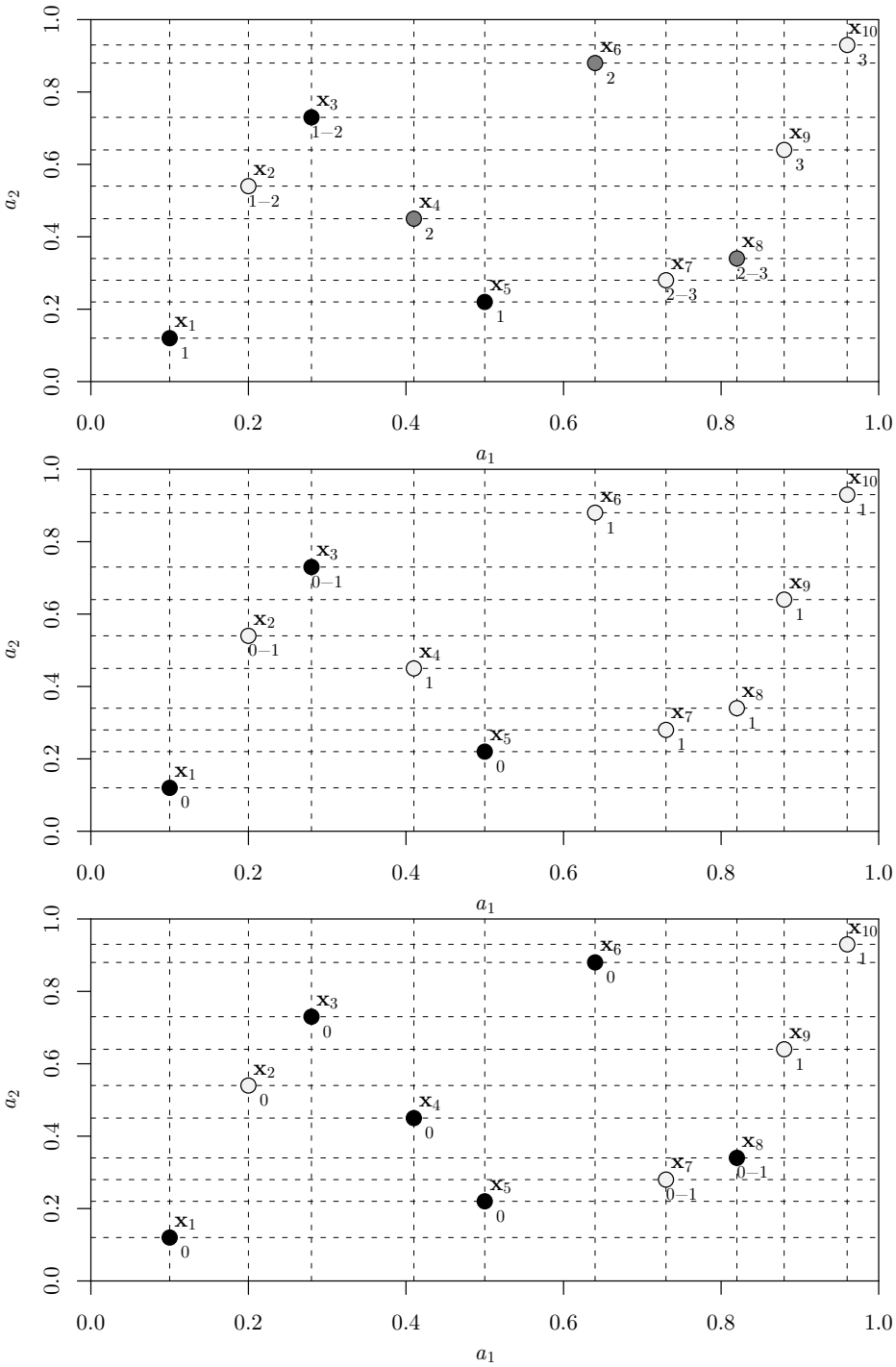

Figure 3.4: Monotone approximation with linear loss for $\alpha = \frac{1}{2}$. In the top figure the 3-class dataset is shown with new labels assigned; in the middle and bottom figures are shown the datasets used in 2 binary subproblems (with new labels). Label e.g. $2-3$ means that the object is assigned label 2 in the lowest optimal solution and label 3 in the greatest optimal solution.

minimal and maximal extensions:

$$\begin{aligned} d^{\min}(\mathbf{x}) &= \max\{\hat{d}_{*i} \colon \mathbf{x}_i \preceq \mathbf{x}\}, \\ d^{\max}(\mathbf{x}) &= \min\{\hat{d}_i^* \colon \mathbf{x}_i \succeq \mathbf{x}\}, \end{aligned} \tag{3.19}$$

We will now prove that every valid extension is capped between $d^{\min}(\mathbf{x})$ and $d^{\max}(\mathbf{x})$:

Theorem 3.10. *Let $d\colon X \to Y$ be monotone and let there exists an optimal solution to the monotone approximation $\hat{\mathbf{d}}$ such that $d(\mathbf{x}_i) = \hat{d}_i$, for each $i = 1, \ldots, n$ (i.e. $d(\mathbf{x})$ is a valid extension of the monotone approximation). Then, for each $\mathbf{x} \in X$, $d^{\min}(\mathbf{x}) \leqslant d(\mathbf{x}) \leqslant d^{\max}(\mathbf{x})$.*

Proof. We know that for each i, $\hat{d}_{*i} \leqslant \hat{d}_i \leqslant \hat{d}_i^*$. Choose any $\mathbf{x} \in X$. Then, since $d(\mathbf{x})$ is monotone, we must have that if $\mathbf{x}_i \preceq \mathbf{x}$ then $d(\mathbf{x}) \geqslant d(\mathbf{x}_i) = \hat{d}_i$. But it means that $d(\mathbf{x}) \geqslant \max\{\hat{d}_i \colon \mathbf{x}_i \preceq \mathbf{x}\} \geqslant \max\{\hat{d}_{*i} \colon \mathbf{x}_i \preceq \mathbf{x}\} = d^{\min}(\mathbf{x})$. Analogously, we can show that $d(\mathbf{x}) \leqslant d^{\max}(\mathbf{x})$. □

Unfortunately, the converse of Theorem (3.10) is not true: there may exist a monotone function $d\colon X \to Y$ such that for each $\mathbf{x} \in X$, $d^{\min}(\mathbf{x}) \leqslant d(\mathbf{x}) \leqslant d^{\max}(\mathbf{x})$, but $d(\mathbf{x})$ is not a valid extension of monotone approximation.

Strong consistency of monotone approximation. We will consider the problem of consistency of monotone approximation with linear loss. Let $h\colon X \to Y$ be a classifier. Let us denote $R_n(h)$ the risk of h when it is trained on the dataset $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$. We say that h is *weakly consistent* (Devroye et al., 1996) if:

$$\mathbb{E}[R_n(h)] \overset{n\to\infty}{\longrightarrow} R^*,$$

where the expectation is taken over the random training sets D of particular size n. In other words, as the size of the training set increases, the risk of classifier averaged over the random choice of training data approaches the Bayes risk, i.e. h approaches the best possible classifier h^*. We say that h is *strongly consistent* if:

$$R_n(h) \overset{n\to\infty}{\longrightarrow} R^*$$

holds with probability one, i.e. for almost every sequence of data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots$, h approaches the Bayes classifier h^*.

We will now prove that under mild assumption about the probability distribution, monotone approximation with linear loss is strongly consistent. To this end suppose $X = V \times \mathbb{R}^{m_0}$, where $V = V_1 \times \ldots \times V_{m-m_0}$ is a finite set. This corresponds to a very general situation encountered in real-life problems, where some of the attributes $V_1, \ldots, V_{m-m_0}$ are ordered and have finite domains while the rest of the attributes are continuous. Every vector $\mathbf{x} \in X$ we can be divided into $\mathbf{x} = \mathbf{x}_V \oplus \mathbf{x}_{\mathbb{R}}$, where $\mathbf{x}_V \in V$ and $\mathbf{x}_{\mathbb{R}} \in \mathbb{R}^{m_0}$. Now we can state the following theorem:

Theorem 3.11. *Assume $P(\mathbf{x}, y)$ is monotonically constrained and let $X = V \times \mathbb{R}^{m_0}$, where V is finite. Assume $P(\mathbf{x})$ has density[2]. Let $d(\mathbf{x})$ be any valid extension of the linear monotone approximation. The $d(\mathbf{x})$ is strongly consistent, i.e.*

$$R_n(d) \overset{n\to\infty}{\longrightarrow} R^*$$

with probability one.

[2] i.e. $P(\mathbf{x})$ is absolutely continuous with respect to the product measure $\mathcal{N}^{m-m_0} \times \lambda^{m_0}$, where $\mathcal{N}$ is a counting measure and λ is a Lebesgue measure.

Proof. We first prove the theorem when $Y = \{0, 1\}$. Let $\mathcal{A}$ be a collection of subsets of X. Let $\mathcal{N}_{\mathcal{A}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ be the number of different sets in:

$$\{\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \cap A \colon A \in \mathcal{A}\}$$

(Devroye et al., 1996). Let us define the *monotone layer* as the set A such that if $\mathbf{x} \in A$ then for every $\mathbf{x}' \succeq \mathbf{x}$ also $\mathbf{x}' \in A$. Then, notice that the set of all possible monotone functions $f \colon X \to Y$, denoted $\mathcal{F}$, corresponds to the set of all monotone layers $\mathcal{A}$ in the sense that for any layer A there is a corresponding function f such that $\mathbf{x} \in A$ if and only if $f(\mathbf{x}) = 1$.

Let $\mathbf{x} = \mathbf{x}_V \oplus \mathbf{x}_{\mathbb{R}}$, where $\mathbf{x}_V \in V$ and $\mathbf{x}_{\mathbb{R}} \in \mathbb{R}^{m_0}$. Let $|V|$ be the cardinality of V. The number of all possible monotone layers cannot exceed $2^{|V|} \times N$, where N is the number of all possible monotone layers on R^{m_0}. Therefore, we can bound:

$$\mathcal{N}_{\mathcal{A}}(\mathbf{x}_1, \ldots, \mathbf{x}_n) \leqslant 2^{|V|} \mathcal{N}_{\mathcal{A}}(\mathbf{x}_{V1}, \ldots, \mathbf{x}_{Vn}),$$

and similarly:

$$\mathbb{E}[\mathcal{N}_{\mathcal{A}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)] \leqslant 2^{|V|} \mathbb{E}[\mathcal{N}_{\mathcal{A}}(\mathbf{x}_{\mathbb{R}1}, \ldots, \mathbf{x}_{\mathbb{R}n})],$$

where the expectation is taken over all possible sets $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ according to $P(\mathbf{x})$. Notice that from the assumption, $P(\mathbf{x}_{\mathbb{R}})$ has density. Devroye et al. (1996) showed (Theorem 13.13 and remark below Corollary 13.3) that for $\mathbf{x}_{\mathbb{R}} \in R^{m_0}$ if $P(\mathbf{x}_{\mathbb{R}})$ has density, then:

$$\mathbb{E}[\mathcal{N}_{\mathcal{A}}(\mathbf{x}_{\mathbb{R}1}, \ldots, \mathbf{x}_{\mathbb{R}n})] = 2^{o(n)}.$$

But this means that also $\mathbb{E}[\mathcal{N}_{\mathcal{A}}(\mathbf{x}_1, \ldots, \mathbf{x}_n)] = 2^{o(n)}$. Then, we proceed analogously as in the proof of Corollary 13.3 in (Devroye et al., 1996), where the Vapnik-Chervonenkis inequality (Theorem 12.5) has been used. The main difference is that we have asymmetric costs of misclassification: the linear loss for binary problem can be written as $L(y, k) = \alpha 1_{k>y} + (1 - \alpha) 1_{k<y}$. However Vapnik-Chervonenkis inequality can be slightly modified to handle the uneven costs, c.f. Lemma 29.1 in (Devroye et al., 1996). Thus, similarly as in (Devroye et al., 1996)) we can conclude that:

$$R_n(d) \stackrel{n \to \infty}{\longrightarrow} \inf_{f \in \mathcal{F}} R(f)$$

with probability one, where $d(\mathbf{x})$ is the classifier which minimizes empirical risk in the class of all monotone functions $\mathcal{F}$. But $d(\mathbf{x})$ can be every valid extension of the binary monotone approximation, because every valid extension achieves the minimum of the empirical risk on the dataset. Moreover, since $P(\mathbf{x}, y)$ is monotonically constrained, from Corollary 2.4 we have that the Bayes classifier $h^* \in \mathcal{F}$ and therefore the infimum is achieved by h^*. Thus, we obtain:

$$R_n(d) \stackrel{n \to \infty}{\longrightarrow} R^* = R(h^*)$$

with probability one.

Now consider the general case $Y = \{1, \ldots, K\}$. We will first prove the theorem for $d^{\min}(\mathbf{x})$. Let $y_k = 1_{y \geqslant k}$ and $d_k(\mathbf{x}) = 1_{d^{\min}(\mathbf{x}) \geqslant k}$, for $k = 2, \ldots, K$. If we denote the linear loss (2.8) by $L(y, k)$, then it is easy to see that:

$$L(y, d^{\min}(\mathbf{x})) = \sum_{k=2}^{K} L(y_k, d_k(\mathbf{x})). \tag{3.20}$$

For each k, consider the random variable $y_k = 1_{y \geqslant k}$ and let $P(\mathbf{x}, y_k)$ denote the distribution induced from $P(\mathbf{x}, y)$. Notice that $P(\mathbf{x}, y_k)$ is monotonically constrained and has density. Moreover, $h_k^*(\mathbf{x}) = 1_{h^*(\mathbf{x}) \geqslant k}$ is the Bayes classifier for $P(\mathbf{x}, y_k)$ with linear loss. From Theorem 3.9 it follows that $d_k(\mathbf{x}_i) = 1_{\hat{q}_{ik} \geqslant \alpha}$ for each $\mathbf{x}_i$. From Theorem 3.6 $1_{\hat{q}_{ik}}$, $i = 1, \ldots, n$ is a monotone approximation

for k-th binary problem (with vector $\mathbf{y}_k$), so $d_k(\mathbf{x}_i)$ is the extension of k-th monotone approximation. Therefore, we can apply the theorem for binary-class case and conclude that:

$$R_n(d_k) \stackrel{n\to\infty}{\longrightarrow} R(h_k^*), \tag{3.21}$$

with probability one, i.e. for some set Ω_k such that $P(\Omega_k) = 1$. This means that for $\Omega = \bigcap_{k=2}^{K} \Omega_k$ (3.21) holds simultaneously for each k. But since $P(\Omega) = 1$ and from (3.20) we obtain:

$$R_n(d^{\min}) \stackrel{n\to\infty}{\longrightarrow} R^*$$

with probability one. Similar conclusion can be drawn for $d^{\max}$. Let $d\colon X \to Y$ be any valid extension of monotone approximation. Then $d^{\min}(\mathbf{x}) \leqslant d(\mathbf{x}) \leqslant d^{\max}(\mathbf{x})$ and we conclude that:

$$R_n(d) \stackrel{n\to\infty}{\longrightarrow} R^*$$

□

The distribution assumption in Theorem 3.11 is very weak and will be satisfied in almost every real problem of ordinal classification with monotonicity constraints. Therefore, monotone approximation is almost universally consistent under the monotonicity assumptions. Unfortunately, the theorem tells nothing about rates of convergence. Those rates may in fact be very slow, especially when the dimensionality of the space m is high. Therefore, monotone approximation will not be used directly as a classification method, except for the low-dimensional problems. Instead, it will be used as a preprocessing method for "monotonizing" the data. Then, the data will be passed to the proper learning method, which will train the monotone ensemble of weak classifiers. This issues will be discussed in Chapter 5.

Appendix: Proofs of the Theorems

Proof of Theorem 3.2

Before proving Theorem 3.2, we sketch the assumptions and the content of the theorem, which leads to the so called *generalized* isotonic regression. Details can be found in (Robertson et al., 1998).

Suppose that Φ is a convex function finite on an interval I containing the values of the coordinates of $\mathbf{y}$ (i.e. for each $i = 1, \ldots, n$, $y_i \in I$) and Φ has value $+\infty$ elsewhere. Let ϕ be a nondecreasing function on I such that for each $u \in I$ $\phi(u)$ is a *subgradient* of Φ, i.e. $\phi(u)$ is a number between the left derivative of Φ at u and the right derivative of Φ at u. For each $u, v \in I$ define the function $\Delta_\Phi(u, v)$ by:

$$\Delta_\Phi(u, v) = \Phi(u) - \Phi(v) - (u - v)\phi(v). \tag{3.22}$$

Then the following theorem holds:

Theorem 3.12. *(Robertson et al., 1998) Let* $\mathbf{p}^*$ *be an isotonic regression of* $\mathbf{y}$ *i.e.* $\mathbf{p}^*$ *solves (3.4). Then it holds:*

$$\sum_{i=1}^{n} \Delta_\Phi(y_i, p_i)) \geqslant \sum_{i=1}^{n} \Delta_\Phi(y_i, y_i^*) + \sum_{i=1}^{n} \Delta_\Phi(y_i^*, p_i)$$

for any monotone vector $\mathbf{p}$ *with the values of the coordinates in* I*, so that* $\mathbf{p}^*$ *minimizes*

$$\sum_{i=1}^{n} \Delta_\Phi(y_i, p_i)$$

in the space of all monotone vectors $\mathbf{p}$ *with values of the coordinates in* I*. The minimizing function is unique if* Φ *is strictly convex.*

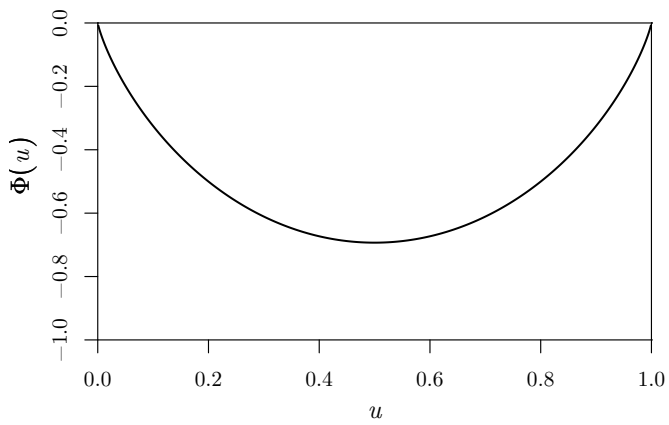

Figure 3.5: Function $\Phi(u) = u \ln u + (1-u)\ln(1-u)$.

Theorem 3.12 states that for any convex function Φ satisfying the assumptions, the isotonic regression minimizes also the function Δ_Φ. Thus, Theorem 3.12 can be used to show that the isotonic regression provides a solution for a wide variety of restricted estimation problems in which the objective function does not look at all like least squares (Robertson et al., 1998). Here, this property will be used to solve the MLE problem (3.3).

Theorem 3.2. *(Robertson et al., 1998) Let* $\mathbf{p}^*$ *be an isotonic regression of* $\mathbf{y}$*. Then,* $\mathbf{p}^*$ *is also the optimal solution to the MLE problem (3.3).*

Proof. Let $I = [0,1]$ and define Φ to be (Robertson et al., 1998):

$$\Phi(u) = \begin{cases} u \ln u + (1-u)\ln(1-u) & \text{for } u \in (0,1) \\ 0 & \text{for } u \in \{0,1\} \end{cases}$$

(see Fig. 3.5). One can show that Φ is indeed convex on I. The first derivative ϕ is given by:

$$\phi(u) = \begin{cases} -\infty & \text{for } u = 0 \\ \ln u - \ln(1-u) & \text{for } u \in (0,1) \\ +\infty & \text{for } u = 1. \end{cases}$$

Then, $\Delta_\Phi(u,v)$ for $u, v \in (0,1)$ is given by:

$$\Delta_\Phi(u,v) = u \ln u + (1-u)\ln(1-u) - u \ln v - (1-u)\ln(1-v). \tag{3.23}$$

It is easy to check that $\Delta_\Phi(u,v) = 0$ if $u = v = 1$ or $u = v = 0$, and that $\Delta_\Phi(u,v) = +\infty$ for $u = 0, v = 1$ or $u = 1, v = 0$. Now, suppose that we want to minimize the function $\sum_{i=1}^n \Delta_\Phi(y_i, p_i)$ between all monotone vectors $\mathbf{p}$ with the coordinates in the range $I = [0,1]$. Then, the first two terms in (3.23) depend only on y_i, so they can be removed from the objective function, thus leading to the problem of minimizing:

$$-\sum_{i=1}^n \big(y_i \ln p_i + (1-y_i)\ln(1-p_i)\big)$$

between all monotone vectors $\mathbf{p}$ with coordinates in the range I, which is exactly the MLE problem (3.3). □

Proof of Lemma 3.1

Lemma 3.1. *Let $\hat{\mathbf{p}}$ be the isotonic regression of the class indices vector $\mathbf{y} = (y_1, \ldots, y_n)$. Suppose, we introduce a new vector of class indices $\mathbf{y}' = (y'_1, \ldots, y'_n)$, such that $y'_i \geqslant y_i$ for all $i = 1, \ldots, n$. Then, $\hat{\mathbf{p}}'$, the isotonic regression of $\mathbf{y}'$ has the following property: $\hat{p}'_i \geqslant \hat{p}_i$, for all $i = 1, \ldots, n$.*

Proof. Assume the contrary, let $\hat{\mathbf{p}}'$ be the isotonic regression of $\mathbf{y}'$, and suppose there exists i, such that $\hat{p}'_i < \hat{p}_i$. Define two other solutions, $\hat{\mathbf{p}}^+$ and $\hat{\mathbf{p}}^-$ in the following way:

$$\hat{p}_i^+ = \max\{\hat{p}_i, \hat{p}'_i\}, \tag{3.24}$$

$$\hat{p}_i^- = \min\{\hat{p}_i, \hat{p}'_i\}. \tag{3.25}$$

Notice that $\hat{\mathbf{p}}^+ \neq \hat{\mathbf{p}}'$ and $\hat{\mathbf{p}}^- \neq \hat{\mathbf{p}}$, since for some i, $\hat{p}'_i < \hat{p}_i$. We show that $\hat{\mathbf{p}}^+, \hat{\mathbf{p}}^-$ are feasible solutions, i.e. they satisfy constraints of (3.4). Suppose $\mathbf{x}_i \succeq \mathbf{x}_j$. Then, since $\hat{\mathbf{p}}, \hat{\mathbf{p}}'$ are feasible, it follows that $\hat{p}_i \geqslant \hat{p}_j$ and $\hat{p}'_i \geqslant \hat{p}'_j$. But from definition of $\hat{p}_i^+$ we have that $\hat{p}_i^+ \geqslant \hat{p}_i$ and $\hat{p}_i^+ \geqslant \hat{p}'_i$, so it also holds that $\hat{p}_i^+ \geqslant \hat{p}_j$ and $\hat{p}_i^+ \geqslant \hat{p}'_j$. Then, $\hat{p}_i^+ \geqslant \max\{\hat{p}_j, \hat{p}'_j\} = \hat{p}_j^+$. Similarly, from the definition of $\hat{p}_j^-$ we have that $\hat{p}_j^- \leqslant \hat{p}_j$ and $\hat{p}_j^- \leqslant \hat{p}'_j$, so it also holds that $\hat{p}_j^- \leqslant \hat{p}_i$ and $\hat{p}_j^- \leqslant \hat{p}'_i$. But then $\hat{p}_j^- \leqslant \min\{\hat{p}_i, \hat{p}'_i\} = \hat{p}_i^-$. Thus, both $\hat{\mathbf{p}}^+, \hat{\mathbf{p}}^-$ are feasible. Let us denote the objective function of (3.4) by $F(\mathbf{y}, \mathbf{p}) = \sum_{i=1}^n (y_i - p_i)^2$. Then, we have:

$$\begin{aligned} F(\mathbf{y}', \hat{\mathbf{p}}^+) - F(\mathbf{y}', \hat{\mathbf{p}}') &= \sum_{i=1}^n \left(\hat{p}_i^{+2} - \hat{p}_i'^2 - 2y'_i\hat{p}_i^+ - 2y'_i\hat{p}'_i\right) = \\ &= \sum_{i=1}^n \left((\hat{p}_i^+ - \hat{p}'_i)(\hat{p}_i^+ + \hat{p}'_i) - 2y'_i(\hat{p}_i^+ - \hat{p}'_i)\right). \end{aligned} \tag{3.26}$$

Since from the definition (3.24) it holds that $\hat{p}_i^+ - \hat{p}'_i \geqslant 0$ and from the assumption of the theorem it holds $y'_i \geqslant y_i$, we have:

$$\sum_{i=1}^n 2y'_i(\hat{p}_i^+ - \hat{p}'_i) \geqslant \sum_{i=1}^n 2y_i(\hat{p}_i^+ - \hat{p}'_i), \tag{3.27}$$

so that:

$$F(\mathbf{y}', \hat{\mathbf{p}}^+) - F(\mathbf{y}', \hat{\mathbf{p}}') \leqslant \sum_{i=1}^n \left((\hat{p}_i^+ - \hat{p}'_i)(\hat{p}_i^+ + \hat{p}'_i) - 2y_i(\hat{p}_i^+ - \hat{p}'_i)\right). \tag{3.28}$$

Moreover, from (3.24)-(3.25) it holds that $\hat{p}_i^+ + \hat{p}_i^- = \hat{p}'_i + \hat{p}_i$, so that:

$$\hat{p}_i^+ - \hat{p}'_i = \hat{p}_i - \hat{p}_i^-, \tag{3.29}$$

and by adding $2\hat{p}'_i$ to both sides of (3.29):

$$\hat{p}_i^+ + \hat{p}'_i = 2(\hat{p}'_i - \hat{p}_i^-) + (\hat{p}_i + \hat{p}_i^-). \tag{3.30}$$

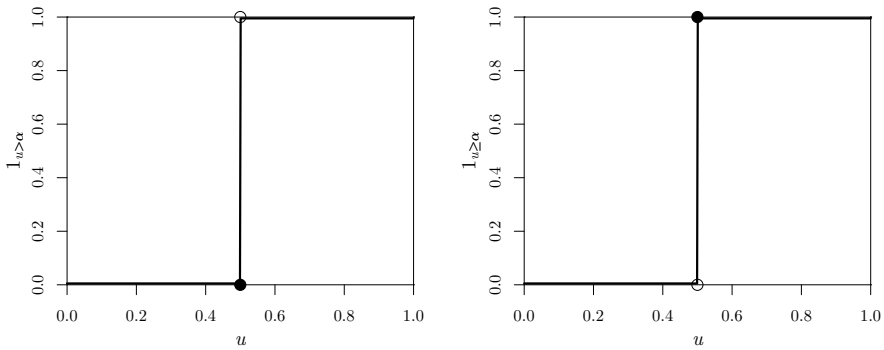

Figure 3.6: Functions $1_{u>\alpha}$ and $1_{u\geqslant\alpha}$ used in the Theorem 3.6, shown for value $\alpha = \frac{1}{2}$. They differ only in point $u = \alpha$.

Putting (3.29)-(3.30) into (3.28), we finally obtain:

$$\begin{aligned}
F(\mathbf{y}', \hat{\mathbf{p}}^+) \; - \; & F(\mathbf{y}', \hat{\mathbf{p}}') \\
&\leqslant \sum_{i=1}^{n} \big((2(\hat{p}'_i - \hat{p}_i^-) + (\hat{p}_i + \hat{p}_i^-))(\hat{p}_i - \hat{p}_i^-) - 2y_i(\hat{p}_i - \hat{p}_i^-)\big) \\
&= \sum_{i=1}^{n} \big(2(\hat{p}_i - \hat{p}_i^-)(\hat{p}'_i - \hat{p}_i^-) + (\hat{p}_i - \hat{p}_i^-)(\hat{p}_i + \hat{p}_i^-) - 2y_i(\hat{p}_i - \hat{p}_i^-)\big) \\
&= \sum_{i=1}^{n} \big(2(\hat{p}_i - \hat{p}_i^-)(\hat{p}'_i - \hat{p}_i^-) + \hat{p}_i^2 - 2y_i\hat{p}_i - \hat{p}_i^{-2} + 2y_i\hat{p}_i^-\big) \\
&= \sum_{i=1}^{n} 2(\hat{p}_i - \hat{p}_i^-)(\hat{p}'_i - \hat{p}_i^-) + F(y, \hat{p}) - F(y, \hat{p}^-) \\
&< \sum_{i=1}^{n} 2(\hat{p}_i - \hat{p}_i^-)(\hat{p}'_i - \hat{p}_i^-), \qquad (3.31)
\end{aligned}$$

where the last inequality comes from the assumption that $\hat{\mathbf{p}}$ is the isotonic regression of $\mathbf{y}$ and $\hat{\mathbf{p}} \neq \hat{\mathbf{p}}^-$. In the last sum, however, for each i, either $\hat{p}_i = \hat{p}_i^-$ or $\hat{p}'_i = \hat{p}_i^-$, so the sum vanishes. Thus, we have:

$$F(\mathbf{y}', \hat{\mathbf{p}}^+) - F(\mathbf{y}', \hat{\mathbf{p}}') < 0, \qquad (3.32)$$

which is a contradiction, because as $\hat{\mathbf{p}}'$ is the isotonic regression of $\mathbf{y}'$, it is the unique optimal solution for class indices $\mathbf{y}'$. □

Proof of Theorem 3.6

Theorem 3.6. *Suppose $\hat{\mathbf{p}} = (\hat{p}_1, \ldots, \hat{p}_n)$ is the optimal solution to the problem of isotonic regression (3.4). Choose some value $\alpha \in [0, 1]$. Then the solution $\hat{\mathbf{d}}_* = (\hat{d}_{*1}, \ldots, \hat{d}_{*n})$ given by $\hat{d}_{*i} = 1_{\hat{p}_i > \alpha}$ for each $i = 1, \ldots, n$ and the solution $\hat{\mathbf{d}}^* = (\hat{d}_1^*, \ldots, \hat{d}_n^*)$ given by $\hat{d}_i^* = 1_{\hat{p}_i \geqslant \alpha}$ for each $i = 1, \ldots, n$ (see Figure 3.6) are the optimal solutions to the problem of binary monotone approximation (3.16) with weights (3.15).*

Moreover, if $\hat{\mathbf{d}} = (\hat{d}_1, \ldots, \hat{d}_n)$ is the optimal integer solution to the problem of binary monotone approximation, it must hold $\hat{d}_{*i} \leqslant \hat{d}_i \leqslant \hat{d}_i^*$, for all $i = 1, \ldots, n$. In particular, if $\hat{\mathbf{d}}_* = \hat{\mathbf{d}}^*$, then the solution to the binary monotone approximation problem is unique.

Proof. Let us define a function $\Phi(u)$ on the interval $I = [0, 1]$ in the following way (see Figure 3.7):

$$\Phi(u) = \begin{cases} \alpha(u - \alpha) & \text{for } u \geqslant \alpha, \\ (1 - \alpha)(\alpha - u) & \text{for } u < \alpha. \end{cases} \qquad (3.33)$$

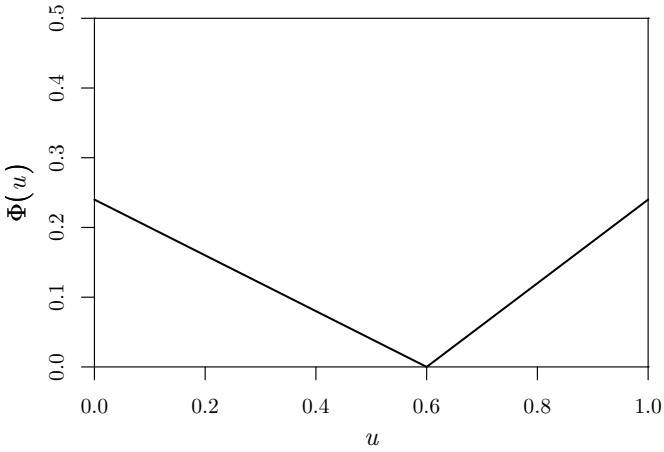

Figure 3.7: Function $\Phi(u)$) defined in (3.33) for $\alpha = 0.6$.

It is easy to check that $\Phi(u)$ is a convex function, but not a strictly convex function. Φ has derivative $\phi(u) = \alpha - 1$ for $u \in [0, \alpha)$, and $\phi(u) = \alpha$ for $u \in (\alpha, 1]$. At point $u = \alpha$, $\Phi(u)$ is not differentiable, but each value in the range $[\alpha - 1, \alpha]$ is a subgradient of $\Phi(u)$.

First, suppose we set $\phi(\alpha) = \alpha - 1$. We remind from (3.22) that:

$$\Delta_\Phi(u, v) = \Phi(u) - \Phi(v) - (u - v)\phi(v).$$

Now, assume $u \in \{0, 1\}$. To calculate $\Delta_\Phi(u, v)$, we need to consider four cases, depending what are the values of u and v:

1. $u = 0$, $v > \alpha$; then $\Phi(u) = \alpha(1 - \alpha)$, $\Phi(v) = \alpha(v - \alpha)$, $\phi(v) = \alpha$, so that $\Delta_\Phi(u, v) = \alpha$.
2. $u = 0$, $v \leqslant \alpha$; then $\Phi(u) = \alpha(1-\alpha)$, $\Phi(v) = (1-\alpha)(\alpha-v)$, $\phi(v) = \alpha-1$, so that $\Delta_\Phi(u, v) = 0$.
3. $u = 1$, $v > \alpha$; then $\Phi(u) = \alpha(1 - \alpha)$, $\Phi(v) = \alpha(v - \alpha)$, $\phi(v) = \alpha$, so that $\Delta_\Phi(u, v) = 0$.
4. $u = 1$, $v \leqslant \alpha$; then $\Phi(u) = \alpha(1 - \alpha)$, $\Phi(v) = (1 - \alpha)(\alpha - v)$, $\phi(v) = \alpha - 1$ so that $\Delta_\Phi(u, v) = 1 - \alpha$.

However, we can comprehensively write those results as:

$$\Delta_\Phi(u, v) = w_u |1_{v>\alpha} - u|,$$

for $u \in \{0, 1\}$, where w_u are given by (3.15). Thus, according to Theorem 3.12, $\hat{\mathbf{p}}$ is the optimal solution to the problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} w_{y_i} |1_{p_i>\alpha} - y_i| \\ \text{subject to} \quad & x_i \succeq x_j \Longrightarrow p_i \geqslant p_j \qquad i, j = 1, \ldots, n. \end{aligned} \tag{3.34}$$

Notice that $\hat{\mathbf{d}}_*$ is also the optimal solution to the problem (3.34), because $1_{u>\alpha}$ is nondecreasing on u, so if $\hat{\mathbf{p}}$ satisfies constraints of (3.34), then so does $\hat{\mathbf{d}}_*$. Moreover, $1_{1_{u>\alpha}} = 1_{u>\alpha}$, so the value of the objective function in (3.34) is the same for both $\hat{\mathbf{p}}$ and $\hat{\mathbf{d}}_*$. But $\hat{\mathbf{d}}_*$ is integer and, for integer solutions, problems (3.34) and (3.16) are the same, so $\hat{\mathbf{d}}_*$ is the solution to the problem (3.16) with the lowest objective value among all the integer solutions to this problem. But, from the analysis of the unimodularity of constraints matrix of (3.16) we know that if $\hat{\mathbf{d}}_*$ is the solution to (3.16) with the lowest objective value among the integer solutions, it is also the optimal solution, since there exists an optimal solution to (3.16), which is integer.

Now, setting $\phi(\alpha) = \alpha$, we repeat the above analysis, which leads to the function $1_{u \geqslant \alpha}$ instead of $1_{u>\alpha}$ and shows that also $\hat{\mathbf{d}}^*$ is the optimal solution to the problem (3.16).

We now prove the second part of the theorem. Assume $v \in \{0, 1\}$ and fix again $\phi(\alpha) = \alpha - 1$. To calculate $\Delta_\Phi(u, v)$, we consider again four cases, depending what are the values of u and v:

1. $u > \alpha$, $v = 0$; then $\Phi(u) = \alpha(u - \alpha)$, $\Phi(v) = \alpha(1 - \alpha)$, $\phi(v) = \alpha - 1$, so that $\Delta_\Phi(u, v) = u - \alpha > 0$.

2. $u \geqslant \alpha$, $v = 1$; then $\Phi(u) = \alpha(u - \alpha)$, $\Phi(v) = \alpha(1 - \alpha)$, $\phi(v) = \alpha$, so that $\Delta_\Phi(u, v) = 0$.

3. $u \leqslant \alpha$, $v = 0$; then $\Phi(u) = (1-\alpha)(\alpha-u)$, $\Phi(v) = \alpha(1-\alpha)$, $\phi(v) = \alpha-1$, so that $\Delta_\Phi(u, v) = 0$.

4. $u < \alpha$, $v = 1$; then $\Phi(u) = (1 - \alpha)(\alpha - u)$, $\Phi(v) = \alpha(1 - \alpha)$, $\phi(v) = \alpha$, so that $\Delta_\Phi(u, v) = \alpha - u > 0$.

From Theorem 3.12 it follows that:

$$\sum_{i=1}^{n} \Delta_\Phi(y_i, p_i) \geqslant \sum_{i=1}^{n} \Delta_\Phi(y_i, \hat{p}_i) + \sum_{i=1}^{n} \Delta_\Phi(\hat{p}_i, p_i) \tag{3.35}$$

for any monotone vector $\mathbf{p}$ with coordinates in the range $[0, 1]$. Notice that if the last term in (3.35) is nonzero, then $\mathbf{p}$ cannot be optimal to the problem (3.34) (since then $\hat{\mathbf{p}}$ has strictly lower cost than $\mathbf{p}$).

Suppose now that $\hat{\mathbf{d}}$ is the optimal integer solution to the binary monotone approximation problem (3.16). But then it is also the solution to the problem (3.34) with the lowest objective value between all the integer solutions (since both problems are exactly the same for integer solutions). Since $\hat{\mathbf{d}}_*$ is an optimal solution to the problem (3.34) and it is integer (so that there exists integer solution which is optimal), $\hat{\mathbf{d}}$ is also an optimal solution to this problem. Then, however, the last term in (3.35) must be zero, so for each $i = 1, \ldots, n$ it must hold $\Delta_\Phi(\hat{p}_i, \hat{d}_i) = 0$ (since all those terms are nonnegative). As $\hat{\mathbf{d}}$ is integer, it is clear from the above analysis of $\Delta_\Phi(u, v)$ for v being integer that it may only happen, if the following conditions hold:

$$\begin{aligned} \hat{p}_i > \alpha &\Longrightarrow \hat{d}_i = 1, \\ \hat{p}_i < \alpha &\Longrightarrow \hat{d}_i = 0, \end{aligned} \tag{3.36}$$

for all $i = 1, \ldots, n$. From the definitions of $\hat{\mathbf{d}}_*$ and $\hat{\mathbf{d}}^*$ it follows that for $\hat{p}_i = \alpha$ it holds that $\hat{d}_{*i} = 0$ and $\hat{d}_i^* = 1$, for $\hat{p}_i > \alpha$ it holds $\hat{d}_{*i} = \hat{d}_i^* = 1$ and for $\hat{p}_i < \alpha$ it holds $\hat{d}_{*i} = \hat{d}_i^* = 0$. From this and from (3.36) we conclude that:

$$\hat{d}_{*i} \leqslant \hat{d}_i \leqslant \hat{d}_i^*,$$

for all $i = 1, \ldots, n$, for any optimal integer solution $\hat{\mathbf{d}}$ to the problem (3.16). $\square$

Chapter 4

Stochastic Dominance-based Rough Set Approach

In this chapter, we explain the Dominance-based Rough Set Approach (DRSA) from the statistical point of view and introduce the extension of DRSA, based on the probabilistic model ordinal classification with monotonicity constraints. We start with the overview of classical rough set theory and DRSA. Then we introduce a statistical framework and method for probability estimation, which leads to the stochastic extension of DRSA (Dembczyński et al., 2007b; Kotłowski and Słowiński, 2008). Finally, we show the equivalence of the extension to the monotone confidence interval estimation. This Chapter heavily relies on the results from Chapter 3. It should be useful to the researchers working on the rough set theory; some of the results apply also to the "classical" rough set approach, based on the indiscernibility relation.

4.1 Dominance-based Rough Set Approach (DRSA)

The rough sets approach, proposed by Pawlak (1982), has been used to deal with the classification problems. It is not able, however, to handle inconsistencies coming from consideration of order relation and monotone relationships in the data, therefore it cannot be applied to the ordinal classification with monotonicity constraints. That is why Greco, Matarazzo, and Słowiński (1999a,b, 2001a) have proposed a new rough set approach that is able to deal with such kind of inconsistencies, called *Dominance-based Rough Set Approach* (DRSA). The origins of DRSA can be, however, trace back to the papers of Słowiński (1993, 1994); Pawlak and Słowiński (1994). Extensive surveys of DRSA can be found in (Greco et al., 2001a, 2004b,c; Słowiński et al., 2005). More information about the axiomatization of the DRSA can be found in (Słowiński et al., 2002a; Greco et al., 2004a).

Before describing DRSA, we overview the classical rough set approach.

4.1.1 Classical Rough Set Theory

Indiscernibility relation. The classical rough set approach (Pawlak, 1982, 1991; Pawlak et al., 1995; Pawlak, 2002; Pawlak and Skowron, 2007) neither takes into account monotonicity constraints nor classes and attributes are ordered. It is based on the assumption that objects having the same description (values on attributes) are indiscernible (similar) with respect to the available information. The *indiscernibility* relation thus generated induces a partition of the universe into

OBJECT	a_1	a_2	y	OBJECT	a_1	a_2	y	OBJECT	a_1	a_2	y
$\mathbf{x}_1$	1	1	1	$\mathbf{x}_9$	2	2	1	$\mathbf{x}_{17}$	3	2	2
$\mathbf{x}_2$	1	1	1	$\mathbf{x}_{10}$	2	2	1	$\mathbf{x}_{18}$	1	3	1
$\mathbf{x}_3$	2	1	1	$\mathbf{x}_{11}$	2	2	1	$\mathbf{x}_{19}$	1	3	3
$\mathbf{x}_4$	2	1	3	$\mathbf{x}_{12}$	2	2	1	$\mathbf{x}_{20}$	2	3	3
$\mathbf{x}_5$	2	1	2	$\mathbf{x}_{13}$	2	2	3	$\mathbf{x}_{21}$	2	3	3
$\mathbf{x}_6$	3	1	2	$\mathbf{x}_{14}$	2	2	3	$\mathbf{x}_{22}$	2	3	1
$\mathbf{x}_7$	3	1	2	$\mathbf{x}_{15}$	3	2	2	$\mathbf{x}_{23}$	3	3	3
$\mathbf{x}_8$	1	2	1	$\mathbf{x}_{16}$	3	2	2	$\mathbf{x}_{24}$	3	3	3

Table 4.1: Example: 24 objects described by 2 attributes a_1, a_2 and class label $y \in \{1, 2, 3\}$.

blocks of indiscernible objects, called *granules*. The indiscernibility relation I is defined as:

$$I = \{(\mathbf{x}_i, \mathbf{x}_j) \colon \quad x_{il} = x_{jl} \ \ \forall\, i, j = 1, \ldots, n, \forall; l = 1, \ldots, m\},$$

where x_d is the value of object $\mathbf{x}$ on attribute d. The equivalence classes of I are called *granules*. The equivalence class for an object $\mathbf{x}_i$ is denoted $I(\mathbf{x}_i)$.

Lower and upper approximations. Any subset of the universe may be expressed in terms of the granules either precisely (as a union of granules) or approximately only. In the latter case, the subset may be characterized by two ordinary sets, called *lower* and *upper approximations*. The subsets approximated in the classification problems are classes $Cl_k, k \in Y$, defined as the subsets of objects from the training set having class value k:

$$Cl_k = \{\mathbf{x}_i \colon y_i = k\}.$$

The lower and upper approximations of class Cl_k are defined, respectively, as:

$$\underline{Cl}_k = \{\mathbf{x}_i \colon I(\mathbf{x}_i) \subseteq Cl_k, i = 1, \ldots, n\},$$
$$\overline{Cl}_k = \{\mathbf{x}_i : I(\mathbf{x}_i) \cap Cl_k \neq \emptyset, i = 1, \ldots, n\}.$$

It always holds that:

$$\underline{Cl}_k \subseteq Cl_k \subseteq \overline{Cl}_k. \tag{4.1}$$

Therefore, if an object $\mathbf{x}_i$ belongs to $\underline{Cl}_k$, it is certainly also an element of Cl_k, while if $\mathbf{x}_i$ belongs to $\overline{Cl}_k$, it may belong to class Cl_k.

Variable precision. For application to the real-life data, some less restrictive definitions were introduced under the name of *variable precision rough sets* (VPRS) (Ziarko, 2001, 2005). The new definitions of approximations (where lower approximation is usually replaced by the term *positive region*) are expressed in the probabilistic terms in the following way. Let $\Pr(y = k|I(\mathbf{x}_i))$ be a probability that any object from granule $I(\mathbf{x}_i)$ belongs to the class Cl_k. Notice that in this definition we assume the probabilities are the same for each object within the same granule. The probabilities are unknown, but are estimated by frequencies

$$\Pr(y = k|I(\mathbf{x}_i)) = \frac{|Cl_k \cap I(\mathbf{x}_i)|}{|I(\mathbf{x}_i)|}.$$

Then, the lower approximation of class Cl_k is defined as:

$$\underline{Cl}_k = \{\mathbf{x}_i \colon \Pr(y = k|I(\mathbf{x}_i)) \geqslant u, i = 1, \ldots, n\}, \tag{4.2}$$

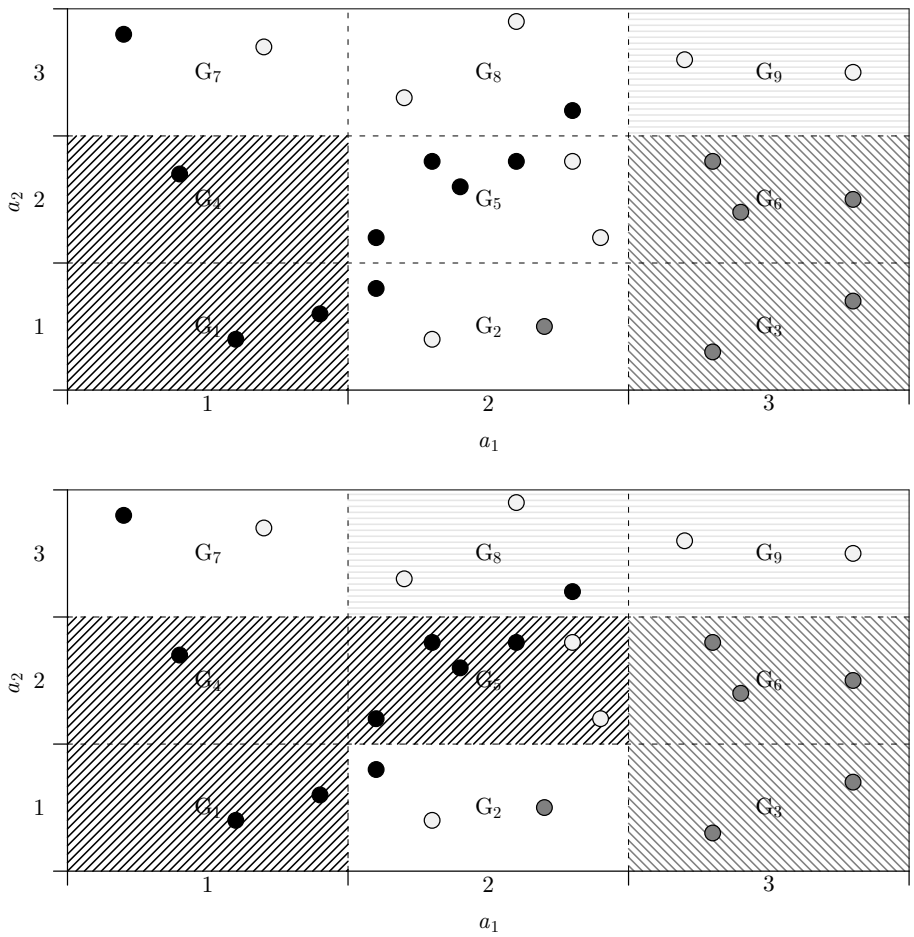

Figure 4.1: Example of three-class problem. Black points are objects from class 1, dark gray points – from class 2, light gray – from class 3. The value sets of two attributes a_1, a_2 are $\{1, 2, 3\}$. On the upper picture lower approximations are presented as the shaded area: $\underline{Cl}_1 = G_1 \cup G_4, \underline{Cl}_2 = G_3 \cup G_6, \underline{Cl}_3 = G_9$. On the lower picture, variable precision lower approximations are presented with precision threshold $u = 2/3$; we have $\underline{Cl}_1 = G_1 \cup G_4 \cup G_5, \underline{Cl}_2 = G_3 \cup G_6, \underline{Cl}_3 = G_8 \cup G_9$

so it is the sum of all granules, for which the probability of class Cl_k is at least equal to some threshold $u \geqslant \frac{1}{2}$. Similarly, the upper approximation of class Cl_k is defined as:

$$\overline{Cl}_k = \{\mathbf{x}_i \colon \Pr(y = k | I(\mathbf{x}_i)) \geqslant l, i = 1, \ldots, n\},$$

where $l \leqslant \frac{1}{2}$ is usually set to $1 - u$ for the complementarity reasons. An example of lower approximations for three-class toy problem is shown in Table 4.1.1 and in Figure 4.1.1. Recently, the VPRS model has been investigated from the viewpoint of some desirable properties of monotonicity in (Błaszczyński et al., 2008), and some new definitions of this model, ensuring the desirable properties, were given.

Notice that concepts of rough approximations are related to the training set only so they contribute to the description of the data and therefore are part of the knowledge discovery process. However rough sets can be used for prediction (classification) purposes as well. Firstly, one can search for the minimal description of objects (set of attributes) which does not increase the inconsistency of the data, called *reduct*; this process can be considered as feature selection. Moreover, one can use lower and upper approximations as a basis for induction of certain and possible rules, respectively (Stefanowski, 1998).

Variable precision as maximum likelihood estimation (MLE). It can be shown that frequencies used for estimating probabilities in are the MLE estimators under assumption of common class probability distribution within a given granule (notice that the assumption that each object in a given granule has the same probability distribution, can be thought of as a probabilistic version of principle of indiscernibility). Let us choose some granule $G = I(\mathbf{x})$. Let n_G be the number of objects in G, and for each class Cl_k, let n_G^k be the number of objects from this class in G. Then the class label y has a multinomial distribution when conditioned on granule G. Let us denote those probabilities $\Pr(y = k|G)$ by p_G^k. Then the conditional probability of observing the $n_G^1, \dots n_G^K$ objects in G, given $p_G^1, \dots, p_G^K$ (conditional likelihood) is the following:

$$L(p; G) = \prod_{k=1}^{K} (p_G^k)^{n_G^k}$$

so that the negative log-likelihood is:

$$\ell(p) = -\ln L(p; G) = -\sum_{k=1}^{K} n_G^k \ln p_G^k \tag{4.3}$$

The minimization of ℓ with additional constraint $\sum_{k=1}^{K} p_G^k = 1$ leads to the well-known fomula for MLE estimators $\hat{p}_G^k$ in multinomial distribution:

$$\hat{p}_G^k = \frac{n_G^k}{n_G} \tag{4.4}$$

which are exactly the frequencies used in VPRS. This observation will lead later to the stochastic generalization of DRSA.

4.1.2 Rough Set Theory for Ordinal Classification

The classical rough set approach is not able to handle inconsistencies coming from consideration of order relations and monotonicity constraints (Greco et al., 2000, 2001a, 2002b). Consider for example assigning companies into one of the bankruptcy risk classes, e.g. "low risk", "medium risk" and "high risk"; the classes are clearly (preference) ordered. Let the set of attributes be: product quality, market share and debt ratio; all those attributes are monotonically correlated with the risk classes: as the product quality and market share increase, and debt ratio decreases, the companies are assigned to lower risk classes. Consider two firms A and B. If firm A has a low value while firm B has a high value of the debt ratio, and evaluations of these firms on other attributes are equal, then, from bankruptcy risk point of view, firm A is at least as good as firm B. Suppose, however, that firm A has been assigned to a class of higher risk than firm B. This is obviously inconsistent with the monotone structure of the problems. Nevertheless, in the classical rough set approach, the two firms will be considered as just discernible and no inconsistency will be stated (Greco et al., 2001a; Słowiński et al., 2002b; Greco et al., 2007a).

For this reason, Greco, Matarazzo, and Słowiński (1999a,b, 2001a) have proposed a new rough set approach that is able to deal with this kind of inconsistencies. This innovation is based on substitution of the indiscernibility relation by a dominance relation in the rough approximations of classes, therefore the new theory has been named Dominance-based Rough Set Approach (DRSA).

In the rough set theory, the term *decision attribute* is often used for the class attribute, the term *decision value* is used for a class label and the term *condition attribute* is used for the attributes other than the decision attribute. We remind about this fact, because the meaning of the names appearing in this chapter (e.g. condition and decision granules, generalized decision, etc.) becomes then clear.

OBJECT	a_1	a_2	y	OBJECT	a_1	a_2	y
$\mathbf{x}_1$	0.1	0.12	1	$\mathbf{x}_6$	0.64	0.88	2
$\mathbf{x}_2$	0.2	0.54	3	$\mathbf{x}_7$	0.73	0.28	3
$\mathbf{x}_3$	0.28	0.73	1	$\mathbf{x}_8$	0.82	0.34	2
$\mathbf{x}_4$	0.41	0.45	2	$\mathbf{x}_9$	0.88	0.64	3
$\mathbf{x}_5$	0.5	0.22	1	$\mathbf{x}_{10}$	0.96	0.93	3

Table 4.2: A set of 10 objects described by 2 attributes a_1, a_2 and class label $y \in \{1, 2, 3\}$.

Dominance relation and dominance principle. The *dominance* relation $\succeq$ is defined as a binary relation on X in the following way (cf. Section 1.2): for any $\mathbf{x}, \mathbf{x}' \in X$ we say that $\mathbf{x}$ *dominates* $\mathbf{x}'$, $\mathbf{x} \succeq \mathbf{x}'$, if on every attribute, $\mathbf{x}$ has evaluation not worse than $\mathbf{x}$, $x_j \geqslant x'_j$, for all $j = 1, \ldots, m$. The dominance relation $\succeq$ is a partial pre-order on X, i.e. it is reflexive and transitive.

The *dominance principle* can be expressed as follows. For every $\mathbf{x}_i, \mathbf{x}_j$, where $i, j = 1, \ldots, n$, it holds:

$$\mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow y_i \geqslant y_j. \tag{4.5}$$

Notice that the dominance principle is related to the training set D only. The dominance principle follows from the monotone relationship between class labels and attribute values. However, in many real-life applications, the dominance principle is not satisfied, i.e. there exists at least one pair of objects violating (4.5). We say that an object $\mathbf{x}_i$ is *inconsistent* if there exists another object $\mathbf{x}_j$, such that $\mathbf{x}_i, \mathbf{x}_j$ violates (4.5). Otherwise, we say that object $\mathbf{x}_i$ is *consistent*. We will sometimes also use the following expression: object $\mathbf{x}_i$ is consistent with $\mathbf{x}_j$, if a pair $\mathbf{x}_i, \mathbf{x}_j$ satisfies (4.5).

Granules as dominance cones. The rough approximations concern granules resulting from information carried out by the class indices. The approximation is made using granules resulting from information carried out by (condition) attributes. These granules are called *decision* and *condition* granules, respectively. The decision granules can be expressed by unions of classes:

$$Cl_k^{\geqslant} = \{\mathbf{x}_i \colon y_i \geqslant k, i = 1, \ldots, n\}$$
$$Cl_k^{\leqslant} = \{\mathbf{x}_i \colon y_i \leqslant k, i = 1, \ldots, n\}.$$

The condition granules are dominating and dominated sets defined, respectively, as:

$$D^+(\mathbf{x}) = \{\mathbf{x}_i \colon \mathbf{x}_i \succeq \mathbf{x}, i = 1, \ldots, n\}$$
$$D^-(\mathbf{x}) = \{\mathbf{x}_i \colon \mathbf{x} \succeq \mathbf{x}_i, i = 1, \ldots, n\}.$$

Remark that decision granules and condition granules are orthogonal cones in decision and conditions space, respectively.

Lower and upper approximations. *Lower approximations* of $Cl_k^{\geqslant}$ and $Cl_k^{\leqslant}$ are defined as follows:

$$\underline{Cl}_k^{\geqslant} = \{\mathbf{x}_i \colon D^+(\mathbf{x}_i) \subseteq Cl_k^{\geqslant}, i = 1, \ldots, n\}, \tag{4.6}$$
$$\underline{Cl}_k^{\leqslant} = \{\mathbf{x}_i \colon D^-(\mathbf{x}_i) \subseteq Cl_k^{\leqslant}, i = 1, \ldots, n\}. \tag{4.7}$$

They include the objects which certainly belong to class $Cl_k^{\geqslant}$ (or $Cl_k^{\leqslant}$), i.e. without any ambiguity caused by inconsistency. Indeed, the certainty comes from the fact that object $\mathbf{x}_i$ belongs to the

lower approximation of class union $Cl_k^{\geqslant}$ (respectively $Cl_k^{\leqslant}$), if no other object in the training set contradicts it, i.e. $\mathbf{x}_i$ is consistent with every other object outside of $Cl_k^{\geqslant}$ (respectively $Cl_k^{\leqslant}$). Otherwise, if there exists an object outside $Cl_k^{\geqslant}$, which dominates $\mathbf{x}_i$, then due to the dominance principle (following from the monotonicity constraints) we cannot say that $\mathbf{x}_i$ belongs to $Cl_k^{\geqslant}$ with certainty.

Upper approximations of $Cl_k^{\geqslant}$ and $Cl_k^{\leqslant}$ are defined as:

$$\begin{aligned}\overline{Cl}_k^{\geqslant} &= \{\mathbf{x}_i \colon D^-(\mathbf{x}_i) \cap Cl_k^{\geqslant} \neq \emptyset, i = 1, \ldots, n\},\\ \overline{Cl}_k^{\leqslant} &= \{\mathbf{x}_i \colon D^+(\mathbf{x}_i) \cap Cl_k^{\leqslant} \neq \emptyset, i = 1, \ldots, n\}.\end{aligned}$$

They include the objects which possibly belong to class $Cl_t^{\geqslant}$ (or $Cl_t^{\leqslant}$), i.e. with a possible ambiguity caused by inconsistency. Notice that for any $k \in Y$, we have $Cl_k^{\geqslant} \cup Cl_{k-1}^{\leqslant} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. This is not the case with the lower approximations. Therefore, we define *the boundary (doubtful) region* for class unions $Cl_k^{\geqslant}$ and $Cl_{k-1}^{\leqslant}$ as:

$$B_k = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \backslash (\underline{Cl}_k^{\geqslant} \cup \underline{Cl}_{k-1}^{\leqslant}), \tag{4.8}$$

which includes the area which does not belong to lower approximations of class unions $Cl_k^{\geqslant}$ and $Cl_{k-1}^{\leqslant}$. Notice that DRSA decomposes the analysis of inconsistencies into $K-1$ binary problems: for each $k = 2, \ldots, K$ we have lower approximations $\underline{Cl}_k^{\geqslant}$, $\underline{Cl}_{k-1}^{\leqslant}$ and boundary B_k which together form the whole set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Such a decomposition will be also used in the stochastic extension of DRSA. A simple example of training set is presented in Table 4.2 and lower approximations of class unions are shown in Figure 4.2.

The *quality of approximation* is defined as a ratio of the number of objects from the dataset that are consistent with respect to the dominance principle, to the number of all objects from the dataset:

$$\gamma = 1 - \frac{\left|\bigcup_{k=2}^{K} B_k\right|}{n}. \tag{4.9}$$

This is a measure of inconsistency present in the dataset.

4.1.3 Generalized Decision in DRSA

For the purpose of this paper, we will focus our attention on another concept from DRSA (as we shall shortly see, equivalent to the notion of approximations), the generalized decision. Suppose, object $\mathbf{x}_i \in \underline{Cl}_k^{\geqslant}$. Since the lower approximation of class union $Cl_k^{\geqslant}$ is a region in which objects certainly belong to $Cl_k^{\geqslant}$, we can state that class index of $\mathbf{x}_i$ should be at least k. Choosing the greatest k, for which $\mathbf{x}_i \in \underline{Cl}_k^{\geqslant}$ holds (denoted by l_i), we know that the class index of $\mathbf{x}_i$ must be at least l_i and we cannot give more precise statement, since we are not certain that the class index of $\mathbf{x}_i$ is at least $l_i + 1$, because $\mathbf{x}_i \notin \underline{Cl}_{l_i+1}^{\geqslant}$. On the other hand, if $\mathbf{x}_i \in \underline{Cl}_k^{\leqslant}$, we known that the class index of $\mathbf{x}_i$ must be at most k. Similarly, choosing the lowest k for which $\mathbf{x}_i \in \underline{Cl}_k^{\leqslant}$ (denoted by u_i), we end up with the interval of classes $[l_i, u_i]$, for which we known that object $\mathbf{x}_i$ must belong to. This interval is often denoted by δ_i, and is called *generalized decision*:

$$\delta_i = [l_i, u_i], \tag{4.10}$$

where:

$$\begin{aligned} l_i &= \max\left\{k \colon \mathbf{x}_i \in \underline{Cl}_k^{\geqslant}, k = 1, \ldots, K\right\},\\ u_i &= \min\left\{k \colon \mathbf{x}_i \in \underline{Cl}_k^{\leqslant}, k = 1, \ldots, K\right\}. \end{aligned} \tag{4.11}$$

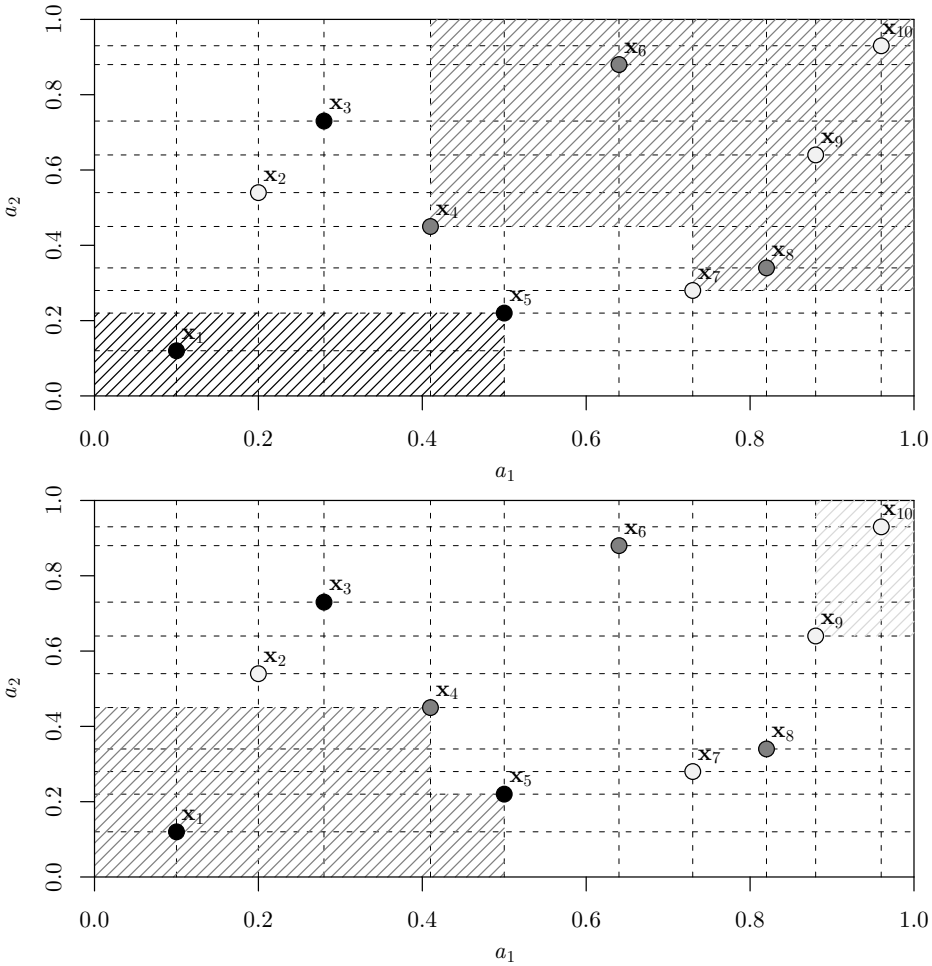

Figure 4.2: Example of three-class problem described in Table 4.2. Black points are objects from class 1, dark gray points – from class 2, light gray – from class 3. On the upper picture lower approximations $\underline{Cl}_1^{\leqslant} = \{\mathbf{x}_1, \mathbf{x}_5\}$ and $\underline{Cl}_2^{\geqslant} = \{\mathbf{x}_4, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9, \mathbf{x}_{10}\}$ are shown, on the lower picture – $\underline{Cl}_2^{\leqslant} = \{\mathbf{x}_1, \mathbf{x}_4, \mathbf{x}_5\}$ and $\underline{Cl}_3^{\geqslant} = \{\mathbf{x}_9, \mathbf{x}_{10}\}$. Approximations $\underline{Cl}_1^{\geqslant}$ and $\underline{Cl}_3^{\leqslant}$ are not shown, since they are trivially $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$.

The generalized decision determines an interval of decision classes to which an object may belong due to the inconsistencies with the dominance principle. Investigating the definitions of lower approximations (4.6)-(4.7) one can show that generalized decision can be easily computed without reference to the lower approximation:

$$\begin{aligned} l_i &= \min\{y_j : \mathbf{x}_j \succeq \mathbf{x}_i, j = 1, \ldots, n\}, \\ u_i &= \max\{y_j : \mathbf{x}_i \succeq \mathbf{x}_j, j = 1, \ldots, n\}. \end{aligned} \tag{4.12}$$

Thus, l_i is the lowest class, to which objects dominating $\mathbf{x}_i$ belong; u_i is the highest class, to which objects dominated by $\mathbf{x}_i$ belong. Obviously, $l_i \leqslant y_i \leqslant u_i$ and if $l_i = u_i$, then object $\mathbf{x}_i$ is consistent with respect to the dominance principle with every other object $\mathbf{x}_j$, for each $i, j = 1, \ldots, n$. The wider the generalized decision, the less precise knowledge about the object we have. One can show (it follows directly from definition) that if $\mathbf{x}_i \succeq \mathbf{x}_j$, then $l_i \geqslant l_j$ and $u_i \geqslant u_j$, i.e. functions l_i and u_i are monotone; in other words, if we replace the original class labels y_i by l_i (or u_i), then we will obtain consistent dataset. Notice that generalized decision is equivalent to lower and upper class labels (3.12) defined in Section 3.2.2.

Let us also remark that the description with generalized decisions is fully equivalent to the

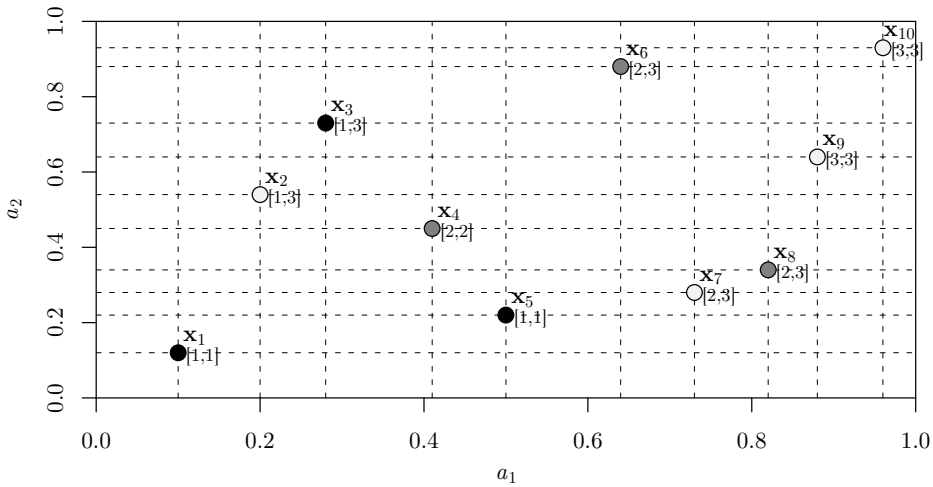

Figure 4.3: Generalized decisions $\delta_i = [l_i, u_i]$ (shown in brackets).

description with rough approximations. Namely, dominance-based lower approximations may be expressed using the generalized decision:

$$\begin{aligned}\underline{Cl}_k^{\geqslant} &= \{\mathbf{x}_i \colon l_i \geqslant k, i = 1, \ldots, n\},\\ \underline{Cl}_k^{\leqslant} &= \{\mathbf{x}_i \colon u_i \leqslant k, i = 1, \ldots, n\}.\end{aligned}$$

Generalized decisions are calculated in Figure 4.3 for the simple dataset from Table 4.2.

4.1.4 Variable Consistency in DRSA

The definitions of lower approximations and generalized decisions for DRSA are quite restrictive: an object may be excluded from the lower approximation due to a single inconsistent object. Indeed, from the definition, object $\mathbf{x}_i$ belong to lower approximation $\underline{Cl}_k^{\geqslant}$ if $D^+(\mathbf{x}_i) \subseteq Cl_k^{\geqslant}$. It is enough that there is one object $\mathbf{x}_j \in D^+(\mathbf{x}_i)$ (so that $\mathbf{x}_j \succeq \mathbf{x}_i$) with $y_j < k$ to exclude $\mathbf{x}_i$ from $\underline{Cl}_k^{\geqslant}$. Moreover, suppose there exist one object dominating all the other objects from dataset, but its class index is the lowest one. Then, all the objects from the dataset, apart from those belonging to the lowest class, will we excluded from *any* lower approximations (they all will fall into the boundary region).

That is why relaxed definitions of lower approximations have been introduced under the name *variable consistency* DRSA (VC-DRSA) (Greco et al., 2001b), which allow object $\mathbf{x}_i$ to be incorporated into lower approximations, if a high fraction of objects dominating $\mathbf{x}_i$ (or being dominated by $\mathbf{x}_i$) is consistent with $\mathbf{x}_i$. Thus, lower approximations (4.6)-(4.7) are redefined as:

$$\underline{Cl}_k^{\geqslant} = \left\{ \mathbf{x}_i \colon \frac{\left| D^+(\mathbf{x}_i) \cap Cl_k^{\geqslant} \right|}{|D^+(\mathbf{x}_i)|} \geqslant l, i = 1, \ldots, n \right\}, \tag{4.13}$$

$$\underline{Cl}_k^{\leqslant} = \left\{ \mathbf{x}_i \colon \frac{\left| D^-(\mathbf{x}_i) \cap Cl_k^{\leqslant} \right|}{|D^-(\mathbf{x}_i)|} \geqslant l, i = 1, \ldots, n \right\}, \tag{4.14}$$

The threshold l is called *consistency level* and is a parameter controlling the acceptable range of ambiguity. The upper approximations are defined by complementarity:

$$\begin{aligned}\overline{Cl}_k^{\geqslant} &= \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \backslash \underline{Cl}_{k-1}^{\leqslant}\\ \overline{Cl}_{k-1}^{\leqslant} &= \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \backslash \underline{Cl}_k^{\geqslant}.\end{aligned}$$

There is, however, a problem with definitions (4.13)-(4.14), since they may lead to the non-monotone assignments to the lower approximations. Consider the dataset from Table 4.2 shown in Figure 4.2. If we set $l = 4/5$ then $\mathbf{x}_6 \in \underline{Cl}_2^{\leqslant}$ since $\frac{|D^-(\mathbf{x}_6) \cap Cl_2^{\leqslant}|}{|D^-(\mathbf{x}_6)|} = \frac{5}{6}$. On the other hand, $\mathbf{x}_3 \notin \underline{Cl}_2^{\leqslant}$, because $\frac{|D^-(\mathbf{x}_3) \cap Cl_2^{\leqslant}|}{|D^-(\mathbf{x}_3)|} = \frac{2}{3}$; but it is counter-intuitive, since $\mathbf{x}_3 \preceq \mathbf{x}_6$ and, moreover, $y_3 < y_6$. This problem has been addressed in (Błaszczyński et al., 2007, 2008), where some modified definitions have been proposed which do not lead to non-monotone assignments.

4.2 Stochastic extension of DRSA (SDRSA)

4.2.1 DRSA as Most Informative Non-invasive Approach

Let us start with proving an interesting fact about DRSA, using the concept of generalized decision (see Section 4.1.3). The proven fact will show that if our approach should not make any additional assumptions that cannot be backed and if it should not invade into the training set by changing or removing the objects (such an approach is called *non-invasive* (Düntsch and Gediga, 2000; Cao-Van, 2003)), the DRSA is the best we can do. In other words, if we want to extend DRSA, we will need to modify the dataset to some extent.

Consider the family of class intervals $\{\alpha_i = [a_i, b_i] \colon i = 1, \ldots, n\}$. Let us define the following relation between such families: the family of class intervals $\{\alpha_i = [a_i, b_i] \colon i = 1, \ldots, n\}$ is *more informative* than the family $\{\beta_i = [c_i, d_i] \colon i = 1, \ldots, n\}$ if for each i, $\alpha_i \subseteq \beta_i$. We show now that the generalized decision (thus, also DRSA rough approximations) is in fact the unique optimal non-invasive approach that holds the maximum amount of information which can be obtained from the data (Dembczyński et al., 2007a):

Theorem 4.1. *The family of generalized decisions $\{\delta_i = [l_i, u_i] \colon i = 1, \ldots, n\}$ is the most informative family of class intervals, among families of class intervals of the form $\{\alpha_i = [a_i, b_i] \colon i = 1, \ldots, n\}$ with the following properties:*

1. *The sets $\{(\mathbf{x}_i, a_i) \colon i = 1, \ldots, n\}$ and $\{(\mathbf{x}_i, b_i) \colon i = 1, \ldots, n\}$, composed of objects with, respectively, class labels a_i and b_i assigned instead of y_i, are consistent with the dominance principle, i.e. if $\mathbf{x}_i \succeq \mathbf{x}_j$, then $a_i \geqslant a_j$ and $b_i \geqslant b_j$ (monotonicity).*

2. *For each $i = 1, \ldots, n$, it holds $a_i \leqslant y_i \leqslant b_i$ (non-invasiveness).*

Proof. It was already noticed in Section 4.1.3 that condition 2 holds for the generalized decisions δ_i. Condition 1 also holds since if $\mathbf{x}_i \succeq \mathbf{x}_j$ then $\{\mathbf{x}_r \colon \mathbf{x}_r \succeq \mathbf{x}_i\} \subseteq \{\mathbf{x}_r \colon \mathbf{x}_r \succeq \mathbf{x}_j\}$, so $l_i = \min\{y_r \colon \mathbf{x}_r \succeq \mathbf{x}_i\} \geqslant \min\{y_r \colon \mathbf{x}_r \succeq x_j\} = l_j$. Analogously, one can show that $u_i \geqslant u_j$.

We now prove the minimality (or maximal informativeness). For any object $\mathbf{x}_i$ and class interval $\alpha_i = [a_i, b_i]$ satisfying the conditions 1-2, it must hold that $a_i \leqslant l_i$. This is because $l_i = \min\{y_r \colon \mathbf{x}_r \succeq \mathbf{x}_i\}$, so there exists object $\mathbf{x}_j$ such that $\mathbf{x}_j \succeq \mathbf{x}_i$ and $l_i = y_j$. But due to conditions 1 and 2 it must hold that $a_i \leqslant a_j \leqslant y_j = l_i$. Analogously, one can show that it must hold that $b_i \geqslant u_i$. Thus, we conclude that for every family of class intervals $\{\alpha_i \colon i = 1, \ldots, n\}$ satisfying the conditions 1-2, it must hold that $\delta_i \subseteq \alpha_i$ for each $i = 1, \ldots, n$. Hence, it follows that $\{\delta_i \colon i = 1, \ldots, n\}$ is the most informative family. □

Thus, if we want to obtain more informative monotone class intervals, we need to be invasive. This constitutes the motivation of stochastic DRSA.

4.2.2 Stochastic Dominance-based Rough Sets

In this section, we extend the definitions of dominance-based lower approximations and generalized decision to the stochastic case.

Probabilistic model. In Section 4.1.1, when we showed that VPRS comes from the MLE, we have made the assumption that in a single granule, each object has the same conditional class probability distribution. This is due to the property of indiscrenibility of objects within a granule.

In case of DRSA, indiscernibility is replaced by a dominance relation, so that a different relation between the probabilities must hold. However, we have already found in Section 2.1 a rudimentary property of the probability distribution for ordinal classification with monotonicity constraints, the stochastic dominance principle:

$$\mathbf{x} \succeq \mathbf{x}' \Longrightarrow P(y \geqslant k|\mathbf{x}) \geqslant P(y \geqslant k|\mathbf{x}') \qquad \forall \mathbf{x}, \mathbf{x}' \in X, k = 1, \ldots, K. \tag{4.15}$$

In other words, if object $\mathbf{x}_i$ dominates object $\mathbf{x}_j$, probability distribution conditioned at point $\mathbf{x}_i$ *stochastically dominates* probability distribution conditioned at $\mathbf{x}_j$. This principle has the advantage of being a natural and intuitive extension of dominance principle (4.5) to the stochastic case.

Stochastic lower approximations. Having stated the probabilistic model, we introduce the stochastic DRSA by relaxing the definitions of lower approximations of classes:

$$\underline{Cl}_k^{\geqslant} = \{\mathbf{x}_i \colon P(y \geqslant k|\mathbf{x}_i) \geqslant \alpha, i = 1, \ldots, n\}, \tag{4.16}$$

$$\begin{aligned} \underline{Cl}_k^{\leqslant} &= \{\mathbf{x}_i \colon P(y \leqslant k|\mathbf{x}_i) \geqslant \alpha, i = 1, \ldots, n\} = \\ &= \{\mathbf{x}_i \colon P(y \geqslant k+1|\mathbf{x}_i) \leqslant 1-\alpha, i = 1, \ldots, n\}, \end{aligned} \tag{4.17}$$

where $\alpha \in (\frac{1}{2}, 1]$ is a fixed threshold. Thus, lower approximation of class union, say $Cl_k^{\geqslant}$, is a region, in which objects are assigned to $Cl_k^{\geqslant}$ with a high probability (at least α). The boundary region $B_k = X \backslash (\underline{Cl}_k^{\geqslant} \cup \underline{Cl}_{k-1}^{\leqslant})$ is the region in which objects belong to any of unions $Cl_k^{\geqslant}$ and $Cl_{k-1}^{\leqslant}$ with probability in the range $(1-\alpha, \alpha)$. Two special cases are important. When $\alpha = 1$, lower approximation corresponds to the certain region for a given class union and, as we shall shortly see, the stochastic definition boils down to the classical definition of the dominance-based lower approximation. When α becomes close to $\frac{1}{2}$, only objects for which $P(y \leqslant k-1|\mathbf{x}_i) = P(y \geqslant k|\mathbf{x}_i) = \frac{1}{2}$ are in the boundary B_k, which corresponds to the Bayes boundary between classes (Duda et al., 2000). Notice that we excluded values $\alpha \leqslant \frac{1}{2}$, because otherwise the lower approximations of complementary class unions would overlap.

Notice that it may happen for some object $\mathbf{x}_i$ that although it does not belong to the class union $Cl_k^{\geqslant}$, it belongs to $\underline{Cl}_k^{\geqslant}$ (because its class probability satisfies $\Pr(y \geqslant k|\mathbf{x}_i) \geqslant \alpha$). The interpretation of this fact is the following: although the class label of $\mathbf{x}_i$ observed in the dataset is smaller than k, i.e. $y_i < k$, such event is less likely than the event $y_i \geqslant k$; hence we should change its class union to the more probable one. Therefore, stochastic approximations lead to reassigning the objects. This is not a surprise, because from Theorem 4.1 it follows that in non-invasive way we cannot do better than DRSA.

Stochastic decision. Having defined the lower approximation, we can obtain generalized decision through the relations (4.11), similarly to the non-stochastic case (notice, however, that the formula (4.12) no longer holds in stochastic case):

$$\begin{aligned} l_i &= \max\{k \colon P(y \geqslant k|\mathbf{x}_i) \geqslant \alpha, k = 1, \ldots, K\} \\ u_i &= \min\{k \colon P(y \geqslant k+1|\mathbf{x}_i) \leqslant 1-\alpha, k = 1, \ldots, K\}. \end{aligned} \tag{4.18}$$

To distinguish between non-stochastic and stochastic definitions, the class intervals defined in (4.18) will be called *stochastic decision*. By carefully looking at (4.18) and reminding the definition of quantiles (see e.g. Section 3.2.4), one can notice that l_i is the greatest $(1-\alpha)$-quantile and u_i is the smallest α-quantile of the conditional class distribution $P(y|\mathbf{x}_i)$. Therefore, the stochastic decision $[l_i, u_i]$ is a confidence class interval such that $y \in [l_i, u_i]$ with probability at least $2\alpha - 1$. Moreover, we can prove that two important properties of generalized decision also hold in the stochastic case:

Theorem 4.2. *Let δ_i be a stochastic decision, defined by (4.18). Then, for every $i, j = 1, \ldots, n$, it holds:*

1. $l_i \leqslant u_i$,
2. $\mathbf{x}_i \succeq \mathbf{x}_j \rightarrow l_i \geqslant l_j$ and $u_i \geqslant u_j$.

Proof. From the definition of l_i, it follows that for each k such that $P(y \geqslant k) < \alpha$, it holds $l_i < k$. Therefore, it must hold $l_i < \min\{k\colon P(y \geqslant k) < \alpha\}$. But since $\alpha \in (\frac{1}{2}, 1]$, we have $\alpha > 1 - \alpha$ and therefore $\{k\colon P(y \geqslant k) \leqslant 1 - \alpha\} \subseteq \{k\colon P(y \geqslant k) < \alpha\}$. This implies that $l_i < \min\{k\colon P(y \geqslant k) \leqslant 1 - \alpha\}$, or equivalently $l_i \leqslant \min\{k\colon P(y \geqslant k + 1) \leqslant 1 - \alpha\} = u_i$. This proves the first property.

The second property follows directly from the consideration of linear loss function in Section 3.2.4. The Bayes classifier for such loss function is $(1-\alpha)$-quantile, where α is the parameter which reflects the strength of asymmetry. Due to the fact that linear loss is a monotone loss matrix, it follows from Theorem 2.2 that Bayes classifier (i.e. $(1 - \alpha)$-quantile function) is monotone under the stochastic dominance assumption for any value $\alpha \in [0, 1]$. This proves the second property. □

Notice that we do not longer have $l_i \leqslant y_i \leqslant u_i$, because it only holds with probability $2\alpha - 1$. The above relation holds with certainty only for $\alpha = 1$. We will shortly see that this case correspond to the non-stochastic DRSA. This is actually in the spirit of Theorem 4.1, where we showed that if both $l_i \leqslant y_i \leqslant u_i$ and monotonicity hold, the best solution is DRSA. Moreover, notice that as α increases, the stochastic decisions broaden (so they become less informative), but on the other hand the probability of catching the observed class value in the class interval increases.

The stochastic decision can be interpreted in the following way: although the original label of the object $\mathbf{x}_i$ is y_i, the object is associated with class interval $[l_i, u_i]$, which contains the most probable class labels. In a special case, when $l_i = u_i \neq y_i$, we may say that object $\mathbf{x}_i$ is *reassigned* from class y_i to $l_i = u_i$. This resembles the monotone approximation problem and, as we will shortly see, the resemblance is not accidental.

4.2.3 Probability Estimation

The stochastic lower approximations and stochastic decision are both defined with respect to the probabilities. However, the probability distribution is unknown. Therefore, we need to estimate the probabilities from data. Then, to obtain the lower approximations, we will plug the estimators into (4.17)-(4.16) instead of real probabilities. In Section 4.3.2, we will show how we can omit the step of probability estimation and directly obtain stochastic lower approximations.

To estimate the probabilities, we will use the method described in detail in Section 3.1, which we remind briefly here. This method is equivalent to the MLE for the binary class case. For the general $K-1$-class case, it is based on solving $K-1$ problems of isotonic regression, to obtain in the k-th problem the estimators of probabilities $P(y \geqslant k|\mathbf{x}_i)$.

For a given $\mathbf{x}_i$, let us define $K-1$ dummy variables $y_{ik} = 1_{y_i \geqslant k}$ for $k = 2, \ldots, K$. In the k-th binary problem, dummy variables y_{ik} play the role of class labels with $Y = \{0, 1\}$, while variables of

the problem correspond to estimating the probabilities $P(y \geqslant k|\mathbf{x}_i)$. Let $\hat{P}(y \geqslant k|\mathbf{x}_i)$, the estimator of corresponding probability, be defined as the optimal solution to the following isotonic regression problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} (y_{ik} - p_i)^2 \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow p_i \geqslant p_j \qquad i, j = 1, \ldots, n. \end{aligned} \tag{4.19}$$

This is the least squares estimation with the monotonicity constraints, which ensures that the probability estimators satisfy the stochastic dominance principle. In Section 3.1, we showed that although the $K-1$ problems are solved separately, it holds $\hat{P}(y \geqslant k|\mathbf{x}_i) \geqslant \hat{P}(y \geqslant k+1|\mathbf{x}_i)$, i.e. the estimators form a proper probability distribution.

Example. Consider the problem shown in Figure 4.4. On the top chart, the probability estimators $\big(\hat{P}(y \geqslant 2|\mathbf{x}_i), \hat{P}(y \geqslant 3|\mathbf{x}_i)\big)$ are shown (notice that $\hat{P}(y \geqslant 1|\mathbf{x}_i \equiv 1$, so it is never shown). Plugging those estimators into the definitions (4.16)-(4.16) and setting $\alpha = 0.6$, we obtain the lower approximations and stochastic decisions shown in the middle and lower pictures.

It is instructive to compare this with Figures 4.2 and 4.3 to see, what has changed comparing with non-stochastic case. First, notice that the lower approximation $\underline{Cl}_2^{\leqslant}$ expanded in stochastic case, by including the objects $\mathbf{x}_2$, $\mathbf{x}_3$ and $\mathbf{x}_6$, which were on the boundary in the non-stochastic case. The reason behind this phenomenon is the following: although $\mathbf{x}_2$ belongs to class Cl_3, it is dominated by objects $\mathbf{x}_3$ and $\mathbf{x}_6$, which belongs to lower classes. Consider estimating the probability $P(y \geqslant 3|\mathbf{x}_i)$ in the second binary problem, when class Cl_3 forms "positive" class $y = 1$ and classes Cl_1 and Cl_2 form negative class $y = 0$. Then, the object $\mathbf{x}_2$ is in minority (one against two objects from "negative" class); therefore, object $\mathbf{x}_2$ will get relatively low probability ($\frac{1}{3}$) of belonging to class Cl_3 and hence for threshold $\alpha = 0.6$ (rather unrestrictive) it can be incorporated into lower approximation $\underline{Cl}_2^{\leqslant}$. Notice that for $\alpha > \frac{2}{3}$, it would not be incorporated into $\underline{Cl}_2^{\leqslant}$ and would stay on the boundary.

Statistical explanation of DRSA. Consider the k-th binary problem of probability estimation. According to Theorem 3.1, if object $\mathbf{x}_i$ is consistent in the k-th binary problem, i.e. there exists no other object such that $\mathbf{x}_i \succeq \mathbf{x}_j$ and $y_{ik} < y_{jk}$, or such that $\mathbf{x}_i \preceq \mathbf{x}_j$ and $y_{ik} > y_{jk}$, then the probability estimate $\hat{P}(y \geqslant k|\mathbf{x}_i)$ will be equal to 1. Moreover, only such objects possess the property $\hat{P}(y \geqslant k|\mathbf{x}_i) = 1$. This implies that if we set $\alpha = 1$, the requirement $\hat{P}(y \geqslant k|\mathbf{x}_i) \geqslant \alpha = 1$ in the definition of stochastic lower approximations (4.16)-(4.17) will be satisfied only by objects consistent in the k-th binary problem.

On the other hand, only such objects are incorporated to the non-stochastic lower approximations (4.7)-(4.6), either $\underline{Cl}_{k-1}^{\leqslant}$ or $\underline{Cl}_k^{\geqslant}$. Indeed, one can simply show that condition $D^+(\mathbf{x}_i) \subseteq Cl_k^{\geqslant}$ is equivalent to $\mathbf{x}_j \succeq \mathbf{x}_i \rightarrow y_{jk} = 1$, while condition $D^-(\mathbf{x}_i) \subseteq Cl_k^{\leqslant}$ to $\mathbf{x}_j \preceq \mathbf{x}_i \rightarrow y_{jk} = 0$. Thus, we conclude the stochastic approximations boil down to the non-stochastic ones in the limit $\alpha = 1$.

From the above analysis, we have two statistical interpretations of the non-stochastic DRSA. From the point of view of lower approximations, DRSA estimates subsets of the training set D, for which either $P(y \geqslant k|\mathbf{x}_i) = 1$ or $P(y \leqslant k-1|\mathbf{x}_i) = 1$; those are the objects which certainly belong to one of the complementary class unions and they form respective lower approximations.

From the point of view of generalized decision, DRSA estimates the confidence intervals which cover the whole probability distribution, i.e. such intervals $[l_i, u_i]$ that $P(y \in [l_i, u_i]|\mathbf{x}_i) = 1$. In other words, those intervals are broad enough but not broader) to ensure that the observed class labels falls inside the intervals.

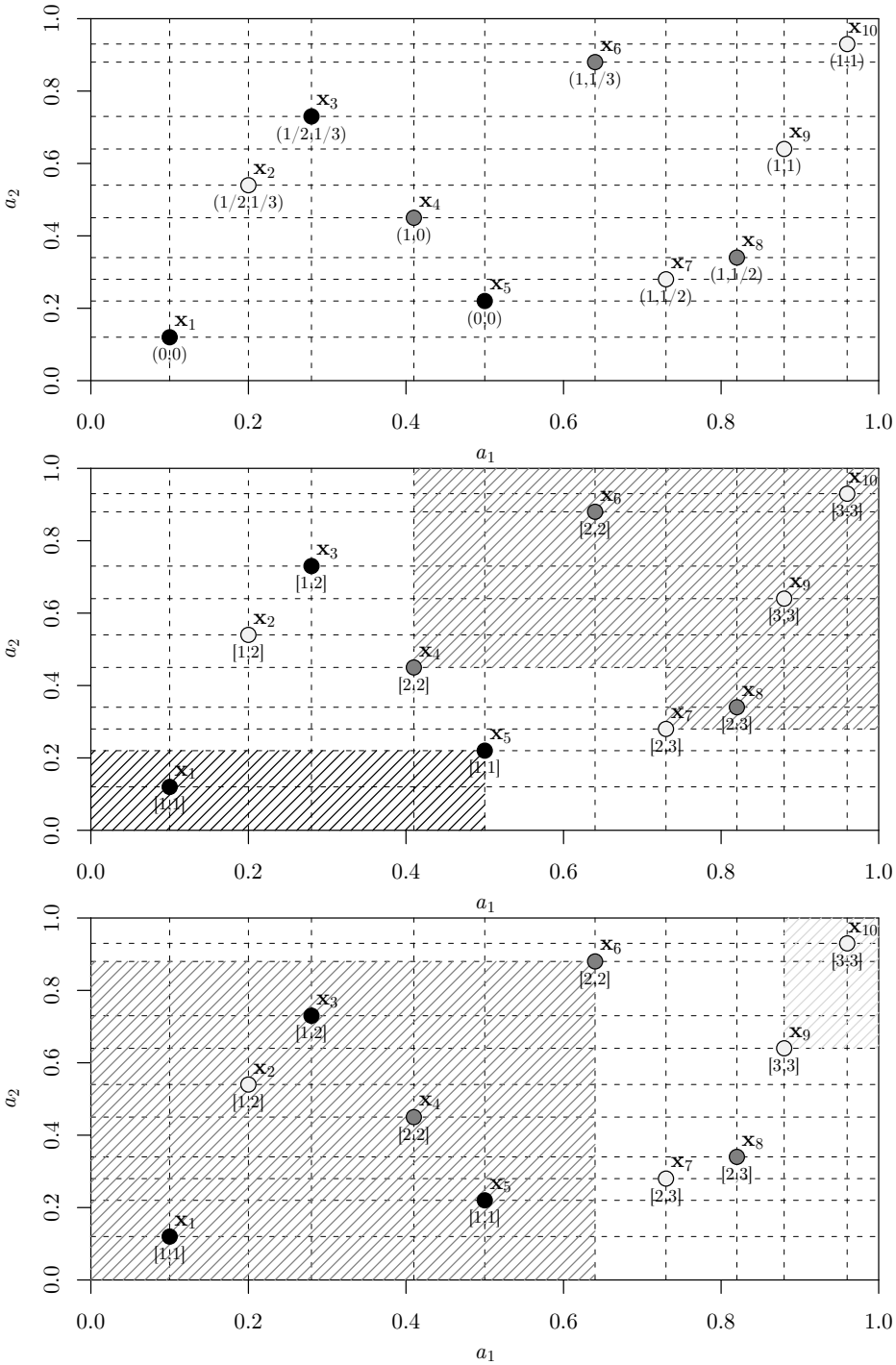

Figure 4.4: Estimation of the probabilities and stochastic approximations for dataset from Table 4.2. In the top figure, estimators $\left(\hat{P}(y \geqslant 2|\mathbf{x}_i), \hat{P}(y \geqslant 3|\mathbf{x}_i)\right)$ are shown for each object. In the middle figure there are marked stochastic lower approximations $\underline{Cl}_2^{\geqslant}$ and $\underline{Cl}_1^{\leqslant}$, and in the lower figure – $\underline{Cl}_3^{\geqslant}$ and $\underline{Cl}_2^{\leqslant}$. In both figures, stochastic decision is shown in brackets. We set $\alpha = 0.6$.

Influence of α. The parameter α can be chosen arbitrary and it is a sort of consistency level, similarly as in VC-DRSA. We have already discussed the limit $\alpha = 1$. As α decreases, we are able to tighten the stochastic decision but, on the other hand, some of original class labels y_i will fall outside those intervals. This is especially useful when the dataset is highly inconsistent.

For instance, consider the case when $\alpha = 1$ (non-stochastic case) and there exists one object $\mathbf{x}_0$ with $y_0 = 1$, which dominates 100 objects $\mathbf{x}_i$, $i = 1, \ldots, 100$, with class labels $y_i = 2$. Then, since every object will be inconsistent, we have $l_i = 1$ and $u_i = 2$ for $i = 0, \ldots, 100$; moreover, none of the objects will enter any lower approximation. However, there is probably something wrong with $\mathbf{x}_0$, rather than with other 100 objects (if we removed $\mathbf{x}_0$, every other object would be consistent). Choosing level $\alpha < 0.99$ will lead to assigning $l_i = 2$ and $u_i = 2$ for every $i = 0, \ldots, 100$. In other words, object $\mathbf{x}_0$ will be effectively reassigned to class Cl_2 and will enter to the lower approximation $\underline{Cl}_2^{\geqslant}$.

As $\alpha \to \frac{1}{2}$, all the stochastic decisions shrink to single points $l_i = u_i$, with exception of those objects, for which $\hat{P}(y \geqslant k|\mathbf{x}_i) = \frac{1}{2}$ for some k. Thus, most of the objects will be reassigned and obtain class labels $l_i = u_i$. This is very similar to the monotone approximation problem encountered in Chapter 3. In the next section, we will focus on the relationship between the monotone approximation and stochastic decision.

4.3 Statistical Learning View: Abstaining Classifiers

In this section, we will look at the problem of stochastic DRSA from the point of view of statistical learning and Bayesian decision theory, described in the Section 1.2.1. This will not only provide us with a comprehensive view o statistical DRSA, but will also computationally improve estimation of the lower approximations: we will show that they can be obtained without estimating the probabilities at all (Dembczyński et al., 2008a).

We will extend the theory considered in Chapter 1 to the case where the classifier is allowed to abstain from any answer. Such functions are known as *abstaining classifiers* (Chow, 1970; Pietraszek, 2005) and are analogous to a domain expert, who is able to say "I don't know", when his knowledge is not sufficient to draw any conclusion. Such an approach is much safer than making an uncertain decision and hence can be preferred in some domains (e.g. in medical diagnosis).

We will show that rough set (both classical and dominance-based) are perfectly tailored for considering the abstaining classifiers.

4.3.1 Statistical Learning View of Classical Rough Sets

Before we present the statistical learning view of stochastic DRSA, we will first consider the VPRS model. A Bayesian-decision approach has already been proposed for VPRS in (Yao and Wong, 1992; Yao, 2007); the theory presented here for VPRS differs slightly from that described in (Yao and Wong, 1992).

Let us consider the classification problem from the statistical learning point of view, as described in Section 1.2.1. The aim is to find a classifier $h(\mathbf{x})$ which minimizes the expected loss (1.1) over the data distribution. Since the probability distribution is unknown, we minimize instead the empirical risk (1.5) within some restricted class of functions. However, now we allow the classifier to refrain from the answer, which is denoted by $h(\mathbf{x}) =?$. The loss function suitable for the problem is the following:

$$L_\beta(y, h(\mathbf{x})) = \begin{cases} 0 & \text{if } h(\mathbf{x}) = y \\ 1 & \text{if } h(\mathbf{x}) \neq y \\ \beta & \text{if } h(\mathbf{x}) =? \end{cases} \tag{4.20}$$

As we see, there is a penalty β for refraining from the answer. To be consistent with the classical rough set theory, we assume that every function must be constant within each granule, i.e. for each $G = I(\mathbf{x}_i)$ for some object $\mathbf{x}_i$, we have:

$$\mathbf{x}_i, \mathbf{x}_j \in G \Longrightarrow h(\mathbf{x}_i) = h(\mathbf{x}_j) \qquad i, j = 1, \ldots, n, \tag{4.21}$$

which is in fact the principle of indiscernibility. Notice that this is the only restriction imposed on the class of functions. We now state:

Theorem 4.3. *The function $\hat{h}$, minimizing the empirical risk with loss function (4.20) between all functions satisfying (4.21) is equivalent to the VPRS in the sense that $\hat{h}(G) = k$ if and only if granule G belongs to the lower approximation of class k with the precision threshold $u = 1 - \beta$, otherwise $\hat{h}(G) =?$.*

Proof. As (4.21) is the only restriction imposed on the class of functions, we can analyze the value of any given function h independently for each granule. Let us choose then a granule $G = I(\mathbf{x}_i)$ for an object $\mathbf{x}_i$. Let us also denote the number of objects in G as n_G, and for each class index $k \in Y$, let n_G^k be the number of objects from class k in G. The total loss of a function h in the granule G is the following:

$$L(h(G)) = \begin{cases} n_G - n_G^k & \text{if } h(G) = k \\ \beta n_G & \text{if } h(G) =? \end{cases}.$$

This follows from the fact that if $h(G) = k$, then for each $\mathbf{x}_i \in G$ such that $y_i \neq k$, function h suffers loss 1. On the other hand, if $h(G) =?$, for each $\mathbf{x}_i \in G$, function h suffers loss β. The best strategy is then to choose the majority class in G or abstain from answer, depending on what loss is lower. The preferred strategy is to choose the majority class, if for some k it holds $n_G - n_G^k \leqslant \beta n_G$, or if:

$$\frac{n_G^k}{n_G} \geqslant 1 - \beta. \tag{4.22}$$

If no k satisfies this relation, the preferred strategy is to choose $\hat{h}(G) =?$. Comparing this result with Section 4.1.1, one can show that the decision $\hat{h}(G) = k$ is chosen if and only if the granule G belongs to the lower approximation of class k with the precision threshold $u = 1 - \beta$. If there is no lower approximation of any class to which G belongs, the function $\hat{h}$ abstains from answer. □

Concluding, the variable precision rough sets can be derived by considering the class of functions constant in each granule and choosing the function $\hat{h}$, which minimizes the empirical risk for loss function (4.20) with parameter $\beta = 1 - u$. For each granule G, if $G \subseteq \underline{Cl}_k$ for some $k \in Y$, then $\hat{h}(\mathbf{x}_i) = k$ for each $\mathbf{x}_i \in G$. Otherwise $\hat{h}(\mathbf{x}_i) =?$ (abstaining from the answer). As we see, classical rough set theory suits well for considering the problems, when the classification procedure is allowed to abstain from predictions for some $\mathbf{x}_i$.

4.3.2 Stochastic DRSA as Monotone Confidence Interval Estimation

The formulation of stochastic DRSA presented in this section, considered from the point of view of statistical learning, is completely new and first appeared in our paper (Dembczyński et al., 2007a). However, a Bayesian-decision view of VC-DRSA has been also proposed in (Greco et al., 2007b).

Interval functions. The case with stochastic DRSA is more sophisticated, because we assume the classifiers are monotone. We propose the following concept of abstaining classifiers in the ordinal classification with monotonicity constraints. Let us consider the classifier $h(\mathbf{x})$, which assigns to

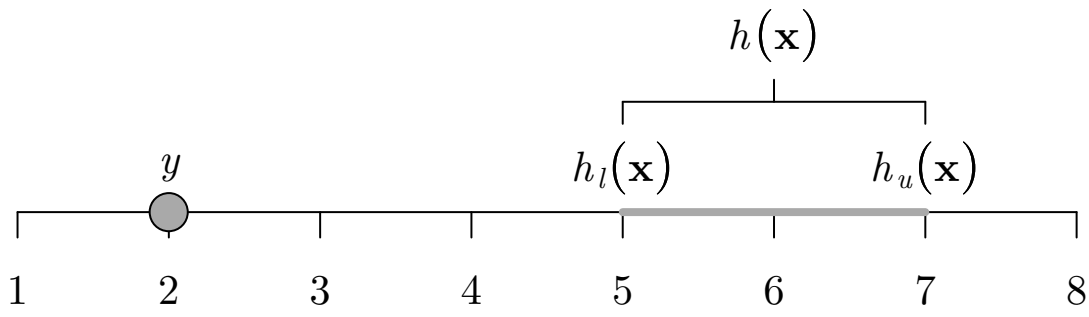

Figure 4.5: The problem with $K = 8$. The actual class label $y = 2$, while the predicted interval is $h(\mathbf{x}) = [h_l(\mathbf{x}), h_u(\mathbf{x})] = [5, 7]$. The total penalty is thus 2β (for imprecision) + 3 (for misclassification).

each point $\mathbf{x}$ the interval of classes, denoted $[h_l(\mathbf{x}), h_u(\mathbf{x})]$. The width of the interval reflects the imprecision of the classifier's response, i.e. to what degree it refrains from the precise answer. The lower and upper ends of each interval are supposed to be monotone functions:

$$\begin{aligned} \mathbf{x}_i \succeq \mathbf{x}_j &\implies h_l(\mathbf{x}_i) \geqslant h_l(\mathbf{x}_j) \qquad i, j = 1, \ldots, n, \\ \mathbf{x}_i \succeq \mathbf{x}_j &\implies h_u(\mathbf{x}_i) \geqslant h_u(\mathbf{x}_j) \qquad i, j = 1, \ldots, n. \end{aligned} \tag{4.23}$$

The interval loss function $L_{\text{int}}(y, h(\mathbf{x}))$ is composed of two terms. First term is a penalty for the size of the interval (degree of imprecision) and equals to $\beta(h_u(\mathbf{x}) - h_l(\mathbf{x}))$. Second term measures the accuracy of the classification and is zero, if $y \in [h_l(\mathbf{x}), h_u(\mathbf{x})]$, otherwise $h(\mathbf{x})$ suffers additional loss equal to the distance of y from the closer interval end:

$$L_{\text{int}}(y, h(\mathbf{x})) = \beta(h_u(\mathbf{x}) - h_l(\mathbf{x})) + 1_{y \notin [h_l(\mathbf{x}), h_u(\mathbf{x})]} \min\{|y - h_l(\mathbf{x})|, |y - h_u(\mathbf{x})|\} \tag{4.24}$$

This model incorporates the abstaining classifiers into the monotone framework in a very intuitive and consistent way. The interval of classes reflects the imprecision of the classifier. Parameter β establishes the trade-off between the precision of the response and its reliability. This is the first model of abstaining classifiers proposed for the ordinal classification with monotonicity constraints.

Interval loss minimization. Let us consider the problem of minimizing (4.24) on the dataset D, subject to constraint that h_u and h_l are monotone:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} L_{\text{int}}(y_i, h(\mathbf{x}_i)) \\ \text{subject to} \quad & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow h_l(\mathbf{x}_i) \geqslant h_l(\mathbf{x}_j) \qquad i, j = 1, \ldots, n \\ & \mathbf{x}_i \succeq \mathbf{x}_j \Longrightarrow h_u(\mathbf{x}_i) \geqslant h_u(\mathbf{x}_j) \qquad i, j = 1, \ldots, n \\ & h_u(\mathbf{x}_i) \geqslant h_l(\mathbf{x}_i) \qquad i = 1, \ldots, n \\ & h_u(\mathbf{x}_i), h_l(\mathbf{x}_i) \in \{1, \ldots, K\} \qquad i = 1, \ldots, n \end{aligned} \tag{4.25}$$

Let $L_\alpha(y_i, k)$ be the linear loss, as defined in Section 3.2.4:

$$L_\alpha(y, k) = \begin{cases} \alpha(k - y) & \text{if } k > y \\ (1 - \alpha)(y - k) & \text{if } k \leqslant y, \end{cases} .$$

One can show that:

Lemma 4.1. *It holds* $L_{\text{int}}(y, h(\mathbf{x})) = L_{1-\beta}(y, h_l(\mathbf{x})) + L_\beta(y, h_u(\mathbf{x}))$.

Proof. We prove the lemma by considering three cases $y > h_u(\mathbf{x})$, $h_l(\mathbf{x}) \leqslant y \leqslant h_u(\mathbf{x})$ and $y < h_l(\mathbf{x})$.

For $y > h_u(\mathbf{x})$, we have $L_{\text{int}}(y, h(\mathbf{x})) = y - h_u(\mathbf{x}) + \beta(h_u(\mathbf{x}) - h_l(\mathbf{x}))$, while $L_{1-\beta}(y, h_l(\mathbf{x})) = \beta(y - h_l(\mathbf{x}))$ and $L_\beta(y, h_u(\mathbf{x})) = (1-\beta)(y - h_u(\mathbf{x}))$.

Similarly, for $h_l(\mathbf{x}) \leqslant y \leqslant h_u(\mathbf{x})$ we have $L_{\text{int}}(y, h(\mathbf{x})) = \beta(h_u(\mathbf{x}) - h_l(\mathbf{x}))$, while $L_{1-\beta}(y, h_l(\mathbf{x})) = \beta(y - h_l(\mathbf{x}))$ and $L\beta(y, h_u(\mathbf{x})) = \beta(h_u(\mathbf{x}) - y)$.

Finally, for $y < h_l(\mathbf{x})$ we have $L_{\text{int}}(y, h(\mathbf{x})) = h_l(\mathbf{x}) - y + \beta(h_u(\mathbf{x}) - h_l(\mathbf{x}))$, while $L_{1-\beta}(y, h_l(\mathbf{x})) = (1-\beta)(h_l(\mathbf{x}) - y)$ and $L_\beta(y, h_u(\mathbf{x})) = \beta(h_u(\mathbf{x}) - y)$. □

Notice that the solution to the problem (4.25) can be non-unique. The question arises, whether we can solve both linear loss problems separately. We have for each i the constraint $h_u(\mathbf{x}_i) \geqslant h_l(\mathbf{x})$. However, one can prove that if we remove the constraint from the optimization process (which is equivalent to solving problems with h_u and h_l separately), the constraint is still satisfied at optimality:

Theorem 4.4. *Consider the interval loss minimization (4.25), relaxed by removing constraint* $h_u(\mathbf{x}_i) \geqslant h_l(\mathbf{x})$ *and assume* $\beta > \frac{1}{2}$. *Then, the optimal solution to this problem, which consists of the linear monotone approximation* $\hat{h}_u$ *with* $\alpha = \beta$ *and the linear monotone approximation* $\hat{h}_l$ *with* $\alpha = 1 - \beta$, *is also the optimal solution to the original problem of interval loss minimization (4.25).*

Proof. We showed in Section 3.2.4 that one obtains linear monotone approximation by solving $K-1$ independent binary monotone approximation problems with weights $w_0 = \alpha$ and $w_1 = 1 - \alpha$. Let $\hat{h}_{uk}(\mathbf{x}_i)$, for $i = 1, \ldots, n$ be the optimal solution in the k-th binary problem with $\alpha = \beta$. Similarly, let $\hat{h}_{lk}(\mathbf{x}_i)$, for $i = 1, \ldots, n$ be the optimal solution in the k-th binary problem with $\alpha = 1 - \beta$. Since $\beta > \frac{1}{2}$, we have $\beta \geqslant 1 - \beta$.

From Theorem 3.6, we know that $\hat{h}_{uk}(\mathbf{x}_i) \geqslant 1_{\hat{p}_i > \beta}$ and that $\hat{h}_{lk}(\mathbf{x}_i) \leqslant 1_{\hat{p}_i \geqslant 1-\beta}$, where $\hat{\mathbf{p}} = (\hat{p}_1, \ldots, \hat{p}_n)$ is the isotonic regression of $\mathbf{y}_k = (y_{1k}, \ldots, y_{nk})$. This implies that for each $i = 1, \ldots, n$, $\hat{h}_{uk} \geqslant \hat{h}_{lk}$. Since k was arbitrary, we have $\hat{h}_u(\mathbf{x}_i) \geqslant \hat{h}_l(\mathbf{x}_i)$. Thus, the relaxed condition $h_u(\mathbf{x}_i) \geqslant h_l(\mathbf{x})$ is satisfied anyway and the pair $(\hat{h}_u, \hat{h}_l)$ is the optimal solutions to the original problem (4.25). □

Stochastic DRSA is equivalent to interval loss minimization. Let us focus on the stochastic DRSA expressed in terms of the stochastic decision. Then, the following equivalence holds:

Theorem 4.5. *The function* $\hat{h}(\mathbf{x}_i) = [\hat{h}_l(\mathbf{x}_i), \hat{h}_u(\mathbf{x}_i)]$, *the solution to the interval loss minimization in the class of monotone intervals functions (4.25), such that* $\hat{h}_l$ *is the greatest* $(1-\beta)$*-linear monotone approximation and* $\hat{h}_u$ *is the smallest* β*-linear monotone approximation, is equivalent to the stochastic decision with threshold* $\alpha = 1 - \beta$*: for* $i = 1, \ldots, n$, $\hat{h}_l(\mathbf{x}_i) = l_i$ *and* $\hat{h}_u(\mathbf{x}_i) = u_i$.

Proof. Since the interval loss minimization separates into two linear monotone approximation problem, we will also consider functions $\hat{h}_l(\mathbf{x}_i)$ and $\hat{h}_u(\mathbf{x}_i)$ separately.

Let us start with $\hat{h}_l(\mathbf{x}_i)$. This is the $(1-\beta)$-linear monotone approximation, which can be obtained by solving $K-1$ binary monotone approximations. From the definition (substituting $\alpha = 1-\beta$) we have that $l_i = \max\{k\colon \hat{P}(y \geqslant k|\mathbf{x}_i) \geqslant 1-\beta\}$, where $\hat{P}(y \geqslant k|\mathbf{x}_i)$ is the probability estimator based on the isotonic regression of the vector $\mathbf{y}_k = (y_{1k}, \ldots, y_{nk})$, as described in Section 4.2.3. From Theorem 3.6 we know that $\hat{h}_{lk}(\mathbf{x}_i) = 1_{\hat{P}(y \geqslant k|\mathbf{x}_i) \geqslant 1-\beta}$; hence we have $l_i = \max\{k\colon \hat{h}_{lk}(\mathbf{x}_i) = 1\} = \hat{h}_l(\mathbf{x}_i)$.

Similarly, for $\hat{h}_u(\mathbf{x}_i)$, we have $u_i = \min\{k\colon \hat{P}(y \geqslant k+1|\mathbf{x}_i) \leqslant \beta\}$. However, we know that $\hat{h}_{uk}(\mathbf{x}_i) = 1_{\hat{P}(y > k|\mathbf{x}_i) > \beta}$; hence $u_i = \min\{k\colon \hat{h}_{u,k+1}(\mathbf{x}_i) = \hat{h}_l(\mathbf{x}_i)\}$. □

Let us notice that the optimal solution which appears in Theorem 4.5 (i.e. the "greatest-smallest" solution) is the solution of the minimal width of the intervals. Moreover, notice that in the context of what was written in Section 3.2.4, such solution is actually obtained by choosing value α decreased by a sufficiently small positive ϵ, for which the unique linear monotone approximation exists.

Concluding, the stochastic DRSA can be derived by considering the class of interval functions, for which the lower and upper ends of intervals are monotone vectors, and choosing the function $\hat{h}$, which minimizes the empirical risk with loss function (4.24) with parameter $\beta = 1 - \alpha$. For each $\mathbf{x}_i$, $i = 1, \ldots, n$, $\hat{h}(\mathbf{x}_i)$ is then a stochastic decision $[l_i, u_i]$.

4.3.3 Summary

We have presented the stochastic extension of DRSA, based the appropriate probabilistic model, satisfying stochastic dominance principle. The lower approximations were defined as the subsets of the dataset D, for which the class unions $Cl_k^{\geqslant}$ or $Cl_k^{\leqslant}$ are observed with probability at least α. Moreover, the equivalent formulation of stochastic approximation is given by introducing the stochastic decision, which assigns to each object class interval between $(1 - \alpha)$-quantile and α-quantile of the class conditional distribution.

Since the probability distribution is unknown, the stochastic approximations (or, equivalently, stochastic decisions) must be estimated from data. One can follow two approaches, based either on lower approximations or stochastic decision, and both lead to exactly the same results:

The approach based on lower approximations. For each class union $Cl_k^{\geqslant}$ and for each object $\mathbf{x}_i$, $i = 1, \ldots, n$, the probability $P(y \geqslant i|\mathbf{x}_i)$ is estimated by solving $K - 1$ isotonic regressions. Having obtained the probability estimates, one can calculate the lower approximations. Notice that isotonic regression has the pessimistic complexity $O(n^4)$; however there exists strong method of reduction of the problem size, described in Section 3.2.2; moreover, there are fast and reliable heuristic algorithms for isotonic regression, working in $O(n^2)$.

The approach based on stochastic decisions. For each object $\mathbf{x}_i$, $i = 1, \ldots, n$, stochastic decision $[l_i, u_i]$ is obtained by minimizing the interval loss on the dataset D within the space of all monotone vectors. This corresponds to solving $2(K - 1)$ binary monotone approximations; this reduces the complexity to $O(n^3)$, which is the improvement in comparison with $O(n^4)$ for isotonic regression. This improvement follows from the fact that we directly estimate the stochastic decisions (hence also lower approximations), without estimating the probabilities.

Chapter 5

Learning Monotone Rule Ensembles

In the previous chapters, we derived a theoretical basis for dealing with ordinal classification problems in the presence of monotonicity constraints. We proposed the monotone approximation, a nonparametric method of classification based on considering the class of all monotone functions. We also explained and extended DRSA from the stochastic point of view and proved it to be a generalization of monotone approximation to the interval loss functions.

In this chapter, we show how nonparametric methods are used in practice to solve the ordinal classification problems with monotonicity constraints. This is accomplished by the following two-phase procedure. In the first phase, we apply monotone approximation to the training data in order to get rid of the inconsistencies. In other words, we "monotonize" the training data. The second phase consists in building the model on the monotonized data. The model has the form of the *monotone rule ensemble*, i.e. the set of decision rules, in which rules are combined in an additive way to form a monotone function. The process of rule induction is based on the boosting strategy of learning. The model makes no errors on the training set, i.e. we say the model *separates* the monotonized data.

Notice that such a two-phase procedure is in the spirit of the rough set approach to data analysis: first the rough approximations are induced from the training set, and then a model is constructed from the approximations, which is consistent with the dataset (i.e. separates the data). In fact, monotonization of the data corresponds exactly to using stochastic dominance-based rough approximations with consistency level $\frac{1}{2}$ (see Chapter 4).

First, we give the overview of boosting, introduce the idea of margin and margin-based loss functions, and present linear programming formulation of boosting (LPBoost). Then, we formulate a general scheme of learning the monotone rule ensembles and describe the two-phase procedure. We examine the asymptotic consistency and generalization bounds of the procedure, which suggests the use of a linear programming boosting framework to generate an ensemble with the maximal value of the margin. This leads to the algorithm called *linear programming monotone rule ensembles* (LPRules).

At the end of this chapter, we describe another algorithm generating a monotone rule ensemble, based on the sigmoid approximation to the linear loss function, called *sigmoid loss monotone rule ensemble* (MORE).

To make our notation consistent with the notation used in the majority of boosting papers, we assume in this chapter that $Y = \{-1, 1\}$ in case of binary classification problems. Let us also define function $\operatorname{sgn}(x)$ to be equal to 1 if $x \geqslant 0$ and -1 for $x < 0$ (sign function).

Algorithm 5.1: Boosting procedure.

input : set of training examples D,
T – number of base learners to be generated.
output: ensemble of base learners $\{b_1, \ldots, b_T\}$.
$f_0(\mathbf{x}) := 0$ **for** $t = 1$ *to* T **do**
$(a_t, b_t) := \arg\min_{a \in \mathbb{R}, b \in \mathcal{B}} \sum_{(y_i, \mathbf{x}_i) \in D} L(y_i, f_{t-1}(\mathbf{x}_i) + ab(\mathbf{x}_i))$;
$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + a_t b_t(\mathbf{x}_i)$;
end

5.1 Boosting

5.1.1 Overview

Boosting appeared in the field of computation learning theory (Kearns and Vazirani, 1994), as the answer to the question, whether a "weak" learning algorithm which, performs just slightly better than random guessing, can be "boosted" into an arbitrarily accurate learning algorithm (Freund and Schapire, 1997). The first provable polynomial-time boosting algorithm was proposed by Schapire (1990). A fast development and great popularity of boosting started, however, from introducing the AdaBoost algorithm by Freund and Schapire (1997), which proved to be very efficient and surprisingly resistant to overfitting on real datasets (Quinlan, 1996; Breiman, 1998). Here, we follow the formulation of boosting introduced by Breiman (1999), Friedman et al. (1998) and Mason et al. (1999), who showed that boosting greedily minimizes a specific loss function on dataset. Note, however, that a dual point of view exists, explaining boosting as minimization of a particular Bregman distance (Kivinen and Warmuth, 1999; Collins et al., 2000; Warmuth et al., 2006).

Let us consider the binary-class case $Y = \{-1, 1\}$. Let $f(\mathbf{x})$ be some real-valued function, such that object $\mathbf{x}$ is classified according to the sign of $f(\mathbf{x})$, $h(\mathbf{x}) = \operatorname{sgn}(f(\mathbf{x}))$, i.e. $\mathbf{x}$ is classified to the "positive" class ($y = 1$) if $f(\mathbf{x}) \geqslant 0$, and $\mathbf{x}$ is classified to the "negative" class ($y = -1$) otherwise. In boosting, we assume that $f(\mathbf{x})$ has the form:

$$f(\mathbf{x}) = \sum_{t=1}^{T} a_t b_t(\mathbf{x}),$$

i.e. it is the linear combination (ensemble) of T functions $b_t \in \mathcal{B}$, called the *base classifiers*; it is usually assumed that $b_t(\mathbf{x}) \in \{-1, 1\}$; the set of base classifiers $\mathcal{B}$ can be, e.g., the space of all classification trees, decision rules, perceptrons, etc.

Let $L(y, f(\mathbf{x}))$ be a loss function, which penalizes the prediction of the real-valued function $f(\mathbf{x}) \in \mathbb{R}$; notice that this is a different loss function then the loss $L(y, h(\mathbf{x}))$ defined before, which penalized the classifier $h(\mathbf{x}) \in \{1, \ldots, K\}$. The function $f(\mathbf{x})$ is learned in the iterative way, by greedily minimizing the loss $L(y, f(\mathbf{x}))$ on the training set. We start with $f(\mathbf{x}) = 0$. Let $f_{t-1}(\mathbf{x})$ be the ensemble at the end of iteration t; a new base classifier $b_t(\mathbf{x})$ and its weight a_t are added to the ensemble without adjusting classifiers which have already been added, by minimizing the loss $L(y, f_{t-1}(\mathbf{x}) + ab(\mathbf{x}))$ on the dataset, with respect to $a \in \mathbb{R}$ and $b \in \mathcal{B}$. The method is formally presented as Algorithm 5.1.

There is no single way to extend the idea of boosting for an arbitrary number of classes. The most popular approach is to transform K-class problem into the sequence of binary problems. In "one-against-all" strategy, K binary class problems are constructed and in the k-th problem, the k-th class forms positive class $y = 1$, while $K - 1$ other classes form negative class $y = -1$. In "one-against-one" strategy, one constructs $K(K-1)/2$ binary problems and for each problem the objects

from one class are discriminated from the objects from another class. In both methods, one uses the binary class boosting several times and then combines the results. The less popular strategy, but well-motivated theoretically, is to solve a single problem with the vector margin function and vector loss function (Friedman et al., 1998; Zou et al., 2005).

In this thesis, we use a different procedure for handling multi-class case, tailored to the problem of ordinal classification.

5.1.2 Margin-based Loss Functions

The aim of the classification problem is to minimize the expected loss function, which will be called *target loss*. Although the loss function used in the boosting procedure (which will be called *training loss*) can be the same as the target loss, it leads to a poor performance of the classifiers trained in this way. This is caused by the fact that the target loss is usually discontinuous and insensitive to the changes of $f(\mathbf{x})$. For instance, consider 0-1 loss: any change of $f(\mathbf{x})$ with no change in sign does not affect the value of 0-1 loss. This makes greedy procedures working poorly, because there is no local improvement possible. Moreover, it is often the case that apart from choosing the class label minimizing the target loss, one needs to obtain the class probabilities, being the measure the 23- of prediction reliability. This also suggests using different loss function for training the classifier, which is capable of probability estimation.

Let us focus again on the binary class case with $Y = \{-1, 1\}$. Let us call $yf(\mathbf{x})$ a *margin* for $\mathbf{x}$; notice that $\mathbf{x}$ is correctly classified if the margin is positive; moreover, the higher the margin, the more certain the prediction. Many binary loss functions can be expressed as single argument functions, depending only on the margin, i.e. $L(y, f(\mathbf{x})) = L(yf(\mathbf{x}))$. For instance, 0-1 loss can be written as $L(u) = 1_{u>0}$, where $u = yf(\mathbf{x})$. Thus, 0-1 loss is insensitive to the value of the margin. Below we present the margin-sensitive loss functions, which can be used within the boosting procedure.

Log-likelihood and hinge loss. The negative log-likelihood loss function can be derived by considering the MLE approach (Friedman et al., 1998; Friedman, 2001) and thus it estimates the probabilities of class labels. It has the following form:

$$L_{\log}(u) = \ln\left(1 + e^{-2u}\right), \tag{5.1}$$

where $u = yf(\mathbf{x})$. The Bayes function for the log-likelihood loss, i.e. function $f^*(\mathbf{x})$ minimizing the expected loss, has the form:

$$f^*(\mathbf{x}) = \frac{1}{2}\ln\frac{P(y=1|\mathbf{x})}{P(y=-1|\mathbf{x})}. \tag{5.2}$$

This shows that log-likelihood loss estimates the (logit of) probabilities at $\mathbf{x}$.

Log-likelihood loss is a convex function, yet it is nonlinear. Therefore, it is sometimes approximated by a piecewise linear function:

$$L_h(u) = (1-u)_+ \tag{5.3}$$

(where $u_+ = u1_{u\geqslant 0}$ means the positive part), called *hinge loss*. Such a function can be incorporated into the mathematical programming problem in terms of the linear constraints. It was used in the linear programming formulation of boosting (Demiriz et al., 2001).

Exponential loss. Exponential loss is another convex function estimating the probabilities. It appeared (implicitly) in the first boosting algorithm, AdaBoost (Freund and Schapire, 1997). It has the following, simple form:

$$L_{\exp}(u) = e^{-u} \tag{5.4}$$

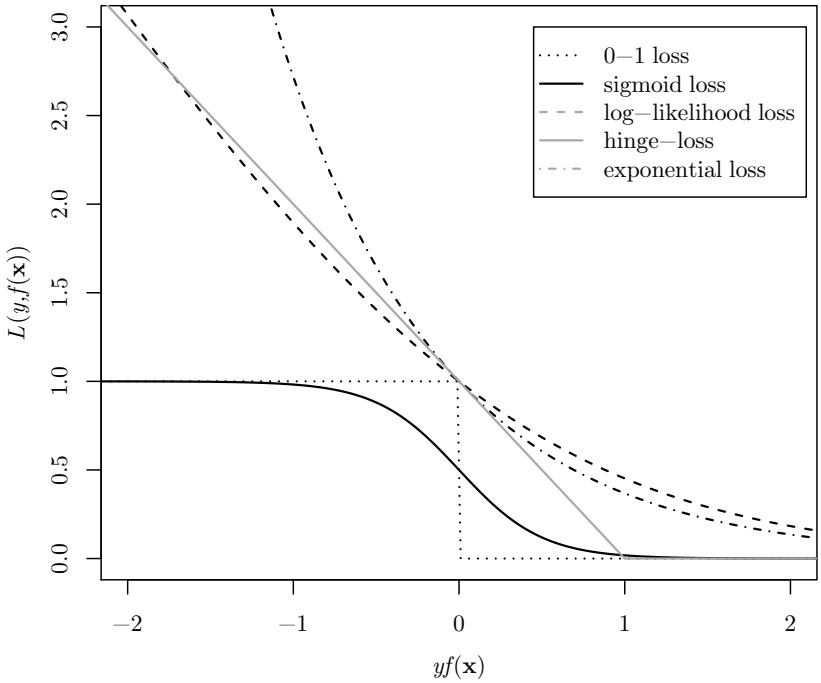

Figure 5.1: Margin-sensitive loss functions. 0-1 loss is shown for comparison.

where, as usual, $u = yf(\mathbf{x})$. Minimizing the expected value of (5.4) with respect to $P(y|\mathbf{x})$ leads to the Bayes function:

$$f^*(\mathbf{x}) = \frac{1}{2} \ln \frac{P(y=1|\mathbf{x})}{P(y=-1|\mathbf{x})} \tag{5.5}$$

which is the same as in case of the log-likelihood loss (5.2). Exponential loss is easy to cope with due to its multiplicative nature. It leads to a very simple reweighting scheme when used with boosting (in fact, this is how the boosting was originally formulated).

Sigmoid loss. Sigmoid loss function is the continuous approximation of the 0-1 loss:

$$L_{\text{sigm}}(u) = \frac{1}{1+e^u}. \tag{5.6}$$

Although not convex, it is differentiable. It does not estimate the probabilities, because the Bayes function is rather aberrant:

$$f^*(\mathbf{x}) = \begin{cases} +\infty & \Pr(y=1|\mathbf{x}) > \frac{1}{2}, \\ -\infty & \Pr(y=-1|\mathbf{x}) < \frac{1}{2}, \\ 0 & \text{otherwise.} \end{cases}$$

There are some theoretical justifications for using this loss function (Mason et al., 1999, 2000). The practical premise is that this loss function is the most similar to the 0-1 loss. Moreover, contrary to the exponential and the log-likelihood loss, this loss function is bounded (its range is from 0 to 1), therefore it is less sensitive to outliers (it does not grow to infinity for misclassified objects).

The margin-sensitive loss functions, along with 0-1 loss, are shown in Figure 5.1.

5.1.3 Margin Theory

It has been observed in experiments that boosting methods achieve very good prediction accuracy and usually do not overfit, even when the number of base classifiers is very large (Drucker and Cortes, 1996; Quinlan, 1996; Breiman, 1998). Moreover, even when boosting drives the training error down to zero, the test error still tends to decrease. This counterintuitive phenomenon has been explained in terms of the ability of boosting to produce combinations of classifiers with large margins.

Consider $y \in \{-1, 1\}$, let $f(\mathbf{x}) = \sum_{t=1}^{T} a_t b_t(\mathbf{x})$, as before, and assume that weights are non-negative and normalized, $\sum_{t=1}^{T} a_t = 1$, i.e. $f(\mathbf{x})$ is a convex combination of classifiers. We refer to the *margin distribution* as to the set of values $\{y_i f(\mathbf{x}_i), i = 1, \ldots, n\}$. The margin theorems (Schapire et al., 1998; Schapire and Singer, 1999; Breiman, 1999; Koltchinskii and Panchenko, 2002, 2006) bound the expected risk of $f(\mathbf{x})$ in terms of its margin distribution. We cite the result obtained by Koltchinskii and Panchenko (2006), which states that for every distribution $P(y, \mathbf{x})$, with probability $1 - \delta$, the following inequality holds:

$$P(yf(\mathbf{x}) \leqslant 0) \leqslant \inf_{\gamma \in (0,1]} \left[P_D(yf(\mathbf{x}) \leqslant \gamma) + M \left(\sqrt{\frac{d}{n\gamma^2}} + \sqrt{\frac{\log \frac{1}{\delta}}{n}}) \right) \right], \tag{5.7}$$

where d is the Vapnik-Chervonenkis dimension (Vapnik, 1998) of the base classifier, M is a universal constant and $P_D(\cdot)$ is the empirical probability distribution, i.e.:

$$P_D(yf(\mathbf{x}) \leqslant \gamma) = \frac{1}{n} \sum_{i=1}^{n} 1_{y_i f(\mathbf{x}_i) \leqslant \gamma}.$$

Thus, with high probability the expected 0-1 loss (generalization error) is bounded by the fraction of objects from the training set with margin $yf(\mathbf{x})$ greater than γ plus some complexity term which increases with decreasing value of γ.

The margin bound shows that ensembles with high margin on the training set generalize well, independently of their sizes. This led to the analysis of ensemble methods from the point of view of their margin distributions (Schapire et al., 1998; Breiman, 1999; Demiriz et al., 2001; Rätsch and Warmuth, 2005; Rudin et al., 2007a,b; Warmuth et al., 2008a).

5.1.4 LPBoost

Demiriz et al. (2001) introduced a new boosting algorithm formulated as a linear program, LPBoost (Linear Programming Boosting), which was later extended by Leskovec and Shawe-Taylor (2003) and then by Warmuth et al. (2008b). LPBoost has the property of directly maximizing the minimal margin on the dataset.

Let us consider the set of *all possible* base classifiers $\mathcal{B} = \{b_j : j = 1, \ldots, J\}$ (we can assume that $\mathcal{B}$ is finite because we have a finite training set and there are at most 2^n classifiers distinguishable on the training set). Let us denote the classification function by $f(\mathbf{x}) = \sum_{j=1}^{J} a_j b_j(\mathbf{x})$, with $a_j \geqslant 0$ and $\sum_{j=1}^{J} a_j = 1$. The problem of maximizing the minimal margin on the dataset can be formulated as a linear program (Breiman, 1999):

$$\begin{aligned} \max: \ & \rho \\ \text{subject to: } & y_i \textstyle\sum_{j=1}^{J} a_j b_j(\mathbf{x}_i) \geqslant \rho \qquad i = 1, \ldots, n, \\ & \textstyle\sum_{j=1}^{J} a_j = 1, \\ & a_j \geqslant 0 \qquad . \end{aligned} \tag{5.8}$$

Indeed, a closer look at the constraints of (5.8 reveals that ρ corresponds to the minimal margin on the training set. Let us refer to ρ as to the *hard margin*. Unfortunately, maximizing the minimal

margin corresponds to optimizing the "worst-case examples" and thus it may put too much attention to the noise or outliers, sacrificing the overall accuracy. Hence, the optimization goal is relaxed, leading to the *soft margin* case:

$$\begin{aligned} \max: \; & \rho - C \textstyle\sum_{i=1}^{n} \xi_i, \\ \text{subject to: } & y_i \textstyle\sum_{j=1}^{J} a_j b_j(\mathbf{x}_i) \geqslant \rho - \xi_i \qquad i = 1, \ldots, n, \\ & \textstyle\sum_{j=1}^{J} a_j = 1, \\ & x_i, a_j \geqslant 0. \end{aligned} \tag{5.9}$$

It can be shown that the parameter C corresponds to the number of objects violating the margin; strictly speaking, $\frac{1}{C}$ upper-bounds the number of margin errors, $\sum_{i=1}^{n} 1_{y_i f(\mathbf{x}_i) \leqslant \rho}$ (Rätsch et al., 2000). To solve (5.9), it is useful to consider the dual problem (Demiriz et al., 2001):

$$\begin{aligned} \min: \; & \beta \\ \text{subject to: } & \textstyle\sum_{i=1}^{n} q_i y_i b_j(\mathbf{x}_i) \leqslant \beta \qquad j = 1, \ldots, J, \\ & \textstyle\sum_{j=1}^{J} q_i = 1, \\ & 0 \leqslant q_i \leqslant C \qquad\qquad i = 1, \ldots, n. \end{aligned} \tag{5.10}$$

There are n dual variables q_i and J constraints, where each constraint corresponds to one base classifier. However, instead of taking into account all possible base classifiers at once, we use the column generation technique and add base classifiers iteratively to the problem. We start with the empty set of classifiers, denoted by $\mathcal{J}_0$. In each iteration, we choose the classifier which maximally violates the corresponding constraint in (5.10), i.e. the one with the highest value $\sum_{i=1}^{n} q_i y_i b_j(\mathbf{x}_i)$. We add it to $\mathcal{J}_0$ and solve the problem (5.10) restricted only to the classifiers from $\mathcal{J}_0$ (so called restricted dual). We proceed in this way until no classifier violates the constraints. Then this means we are at optimum of (5.10). The scheme of the LPBoost algorithm is presented as Algorithm 5.2 (Demiriz et al., 2001).

Notice that the minimum is always reached in a finite number of steps (as opposed to e.g. AdaBoost). The classifiers are chosen according to the value $\sum_{i=1}^{n} q_i y_i b_j(\mathbf{x}_i)$, which is the weighted classification error scaled from -1 to 1. In other words, in each iteration we choose the classifier which performs the best according to the current weights. From the dual point of view, we are at minimum if we find *the least favorable distribution* (Breiman, 1999) $q_i, i = 1, \ldots, n$. Notice that the regularization constant C prevents us from putting too much weight on any of the objects. Finally, notice that it follows from the duality slackness conditions (Papadimitriou and Steiglitz, 1998) that for each $i = 1, \ldots, n$ we have:

$$q_i \left(y_i \sum_{j \in \mathcal{J}_0} a_j b_j(\mathbf{x}_i) + \xi_i - \rho \right) = 0.$$

In other words, only the "hardest" objects, with margins lower then the soft margin ρ (i.e. those making margin errors) receive non-zero weights. The LPBoost formulation is simple and elegant. Moreover, it tends to produce a very sparse solution, with only few non-zero coefficients. This improves the interpretability of the model, which will appear to be especially useful for monotone rule ensembles.

5.2 Monotone Rule Ensembles

We now introduce an ensemble of specific base classifiers, known as decision rules. Since we deal with the ordinal classification with monotonicity constraints, we will restrict to the ensembles which are monotone functions.

Algorithm 5.2: Linear programming boosting (LPBoost).

input : set of training examples D,
C – regularization penalty.
output: ensemble of base learners $\{b_1, \ldots, b_T\}$.
$optimum := false$;
$\mathcal{J}_0 := \emptyset$;
$q_i := \frac{1}{n}, i = 1, \ldots, n$;
repeat
  $j_{\max} := \arg\max_{j=1,\ldots,J} \sum_{i=1}^{n} q_i y_i b_j(\mathbf{x}_i)$;
  if $\sum_{i=1}^{n} q_i y_i b_{j_{\max}}(\mathbf{x}_i) \leqslant \beta$ **then**
    $optimum := true$;
  end
  else
    $\mathcal{J}_0 := \mathcal{J}_0 \cup \{j_{\max}\}$;
    Solve the problem:

$$\begin{aligned} \min: & \sum_{i=1}^{n} \beta \\ \text{subject to:} & \sum_{i=1}^{n} q_i y_i b_j(\mathbf{x}_i) \leqslant \beta && j \in \mathcal{J}_0, \\ & \sum_{j=1}^{J} q_i = 1, \\ & 0 \leqslant q_i \leqslant C && i = 1, \ldots, n. \end{aligned}$$

    to obtain new weight vector q_i, $i = 1, \ldots, n$ and β;
  end
until *optimum = true;*
Obtain a_j, $j \in \mathcal{J}_0$ from the dual variables.

5.2.1 Decision Rules

We will consider a learning algorithm involving *decision rules*. Such decision rules are simple and interpretable logical statements of the form: "if *conditions* then *decision*". They can be treated as simple classifiers that give a constant response to examples satisfying the condition part, and abstain from the response for other examples.

The problem of induction of decision rules has been widely considered in machine learning (Michalski, 1983; Clark and Niblett, 1989; Cohen, 1995; Fürnkranz, 1996), logical analysis of data (Boros et al., 2000) and rough set approaches to knowledge discovery (Pawlak, 1991; Słowiński, 1992; Grzymala-Busse, 1992; Stefanowski, 1998; Greco et al., 2001c, 2005). The most popular algorithms were based on a sequential covering procedure (also known as separate-and-conquer approach). In this technique, a rule is learned which covers a part of the training examples, then examples are removed from the training set and the process is repeated until no examples remain.

The interest in decision rule models is still growing in machine learning – let us mention such algorithms as RuleFit (Friedman and Popescu, 2005), SLIPPER (Cohen and Singer, 1999), Lightweight Rule Induction, (LRI) (Weiss and Indurkhya, 2000), ENDER (Błaszczyński et al., 2006b,a; Dembczyński et al., 2008c,e), MLRules (Dembczyński et al., 2008d). All these algorithms follow a specific iterative approach to decision rule generation by treating each decision rule as a subsidiary base classifier in the ensemble. This approach can be seen as a generalization of the sequential covering, because it approximates the solution of the prediction task by sequentially adding new rules to the ensemble without adjusting those that have already been added (RuleFit is an exception since it generates the trees first and then transforms them into rules). Each rule is fitted by concentrating

on objects which were hardest to classify correctly by rules already present in the ensemble. All these algorithms can be explained within the framework of boosting.

However, all the algorithms mentioned above were designed to deal with regular classification problems, where neither the order relation nor monotonicity constraints present in the data are taken into account. Here, we propose a methodology for monotone rule induction, i.e. generating rules that are adapted to the ordinal classification problem and do not violate monotonicity constraints.

5.2.2 Monotone Rule Ensembles for Binary Classification

We assume the binary-class case $Y = \{-1, 1\}$, so that "positive" class is labeled with $+1$, while "negative" class is labeled with -1.

Definition. Let X_j be the value set of the j-th attribute, i.e. the set of all possible values for the j-th attribute. Condition part of the rule consist of *elementary conditions* of the form $x_j \geqslant s_j$ or $x_j \leqslant s_j$ for some $s_j \in X_j$. Let Φ denote the set of elementary conditions constituting the condition part of the rule. We can define *rule* as a function $\Phi(\mathbf{x})$ which is equal to 1 or -1 if an objects $\mathbf{x}$ satisfies all the conditions of the rule (i.e. object is *covered* by the rule), otherwise $\Phi(\mathbf{x}) = 0$. As we will see, rules with $\Phi(\mathbf{x}) = 1$ votes for a higher class, while rules with $\Phi(\mathbf{x}) = -1$ votes for a lower class. We assume that either $\Phi(\mathbf{x}) \geqslant 0$ for all $\mathbf{x} \in X$, or $\Phi(\mathbf{x}) \leqslant 0$ for all $\mathbf{x} \in X$. In other words, for all covered objects, function $\Phi(\mathbf{x})$ consequently returns the same value, either -1 or $+1$. Here we consider a real-valued classification function which is a linear combinations of T decision rules:

$$f(\mathbf{x}) = \sum_{t=1}^{T} a_t \Phi_t(\mathbf{x}), \tag{5.11}$$

where $a_t, t = 1, \ldots, T$ are *positive* coefficients which will be called *weights* of the rules. Object $\mathbf{x}$ is classified to the class indicated by the sign of $f(\mathbf{x})$, i.e. $h(\mathbf{x}) = \operatorname{sgn}(f(\mathbf{x}))$. The combination (5.11) has a very simple interpretation as a voting procedure: rules with $\Phi(\mathbf{x}) \geqslant 0$ vote for the positive class, while rules with $\Phi(\mathbf{x}) \leqslant 0$ – for the negative class. Object $\mathbf{x}$ is classified to the class with a higher vote (which is equivalent to classification according to the sign of $f(\mathbf{x})$).

Monotonicity of rule ensemble. We assume the monotonicity constraints are present in the data and thus we require that function $f(\mathbf{x})$ must be monotone. The following theorem establishes the sufficient conditions for the monotonicity of rule ensemble:

Theorem 5.1. *Let $f(\mathbf{x})$ be a rule ensemble, i.e. a function of the form (5.11). Then, in order to maintain monotonicity of $f(\mathbf{x})$, it is sufficient that for each rule $t = 1, \ldots, T$, $\Phi_t(\mathbf{x}) \geqslant 0$ and all elementary conditions in Φ_t are of the form $x_j \geqslant s_j$, or $\Phi_t(\mathbf{x}) \leqslant 0$ and all elementary conditions are of the form $x_j \leqslant s_j$.*

Proof. Assume that all the elementary conditions are of the form $x_j \geqslant s_j$ if $\Phi_t(\mathbf{x}) \geqslant 0$ and are of the form $x_j \leqslant s_j$ if $\Phi_t(\mathbf{x}) \leqslant 0$. In each case, a single rule $r_t(\mathbf{x}) = a_t \Phi_t(\mathbf{x})$ is a monotone function. Indeed, suppose $\mathbf{x} \succeq \mathbf{x}'$ and consider $\Phi_t(\mathbf{x}) \geqslant 0$. Then, since $x_j \geqslant x'_j$, if $x'_j \geqslant s_j$, then also $x_j \geqslant s_j$, so that $\Phi_t(\mathbf{x}) \geqslant \Phi_t(\mathbf{x}')$ and thus $r_t(\mathbf{x}) \geqslant r_t(\mathbf{x}')$. Similarly, consider $\Phi_t(\mathbf{x}) \leqslant 0$. Then, since $x_j \geqslant x'_j$, if $x_j \leqslant s_j$ then also $x'_j \leqslant s_j$ so that $\Phi_t(\mathbf{x}) \geqslant \Phi_t(\mathbf{x}')$ and thus $r_t(\mathbf{x}) \geqslant r_t(\mathbf{x}')$. Hence, $f(\mathbf{x})$ is the sum of monotone functions, so is a monotone function. □

Rules, which satisfy the conditions of Theorem 5.1 are called *monotone*. Moreover, monotone rules with $\Phi_t(\mathbf{x}) \geqslant 0$ will be called *upward*, while those with $\Phi_t(\mathbf{x}) \leqslant 0$ will be called *downward*. The function $f(\mathbf{x})$ of the form (5.11), consisting of monotone rules is called *monotone rule ensemble*.

5.2.3 Monotone Rule Ensembles with Linear Loss

Let us go back to the general K-class case and consider the rule ensemble algorithm for minimizing the linear loss function (2.8):

$$L(y,k) = \begin{cases} \alpha(k-y) & \text{if } k > y \\ (1-\alpha)(y-k) & \text{if } k \leqslant y, \end{cases} \tag{5.12}$$

We will show how to solve the general problem by decomposition into $K-1$ binary problems.

Decomposition procedure. Suppose we have an access to the learning algorithm for binary classification problems, which can produce classifier $h(\mathbf{x}) \in \{-1,1\}$ with small weighted 0-1 loss:

$$L(h(\mathbf{x}),y) = w_y 1_{yh(\mathbf{x})<0}, \tag{5.13}$$

for $y \in \{-1,1\}$, where $w_{-1} = \alpha$ and $w_1 = 1-\alpha$. It is not hard to verify that the weighted 0-1 loss is exactly the linear loss (5.12) for binary-class case.

Let y be the class label and let us define $y_k = \operatorname{sgn}(y-k)$ for $k = 2,\ldots,K$, similarly as in Chapter 3 (with the exception that now $y \in \{-1,1\}$). Therefore, $y_k = 1$ corresponds to the class union "at least k", while $y_k = -1$ corresponds to the class union "at most $k-1$". Suppose we train $K-1$ binary classifiers described above, $k = 2,\ldots,K$, where the k-th classifier $h_k(\mathbf{x})$ is trained on the dataset with original class labels y substituted by labels y_k. We combine binary classifiers into a single K-class classifier in the following way:

$$h(\mathbf{x}) = 1 + \sum_{k=2}^{K} 1_{h_k(\mathbf{x})=1}. \tag{5.14}$$

Notice that we do not assume classifiers are consistent, i.e. we do not assume $h_k(\mathbf{x}) \geqslant h_{k'}(\mathbf{x})$ when $k < k'$. We will show that the final classifier makes error not greater then the sum of errors of base classifiers:

Theorem 5.2. *Let $L(y,h_k(\mathbf{x}))$ be the linear loss, given by (5.13), suffered by the k-th binary classifier and let $L(y,h(\mathbf{x}))$ be the linear loss suffered by a composite classifier defined by (5.14). Then we have:*

$$L(y,h(\mathbf{x})) \leqslant \sum_{k=2}^{K} L(y_k,h_k(\mathbf{x}))$$

Proof. For simplicity, we denote $h_k(\mathbf{x})$ by h_k and $h(\mathbf{x})$ by h. We have:

$$\begin{aligned} \sum_{k=2}^{K} L(y_k,h_k) &= \sum_{k=2}^{K} ((1-\alpha)1_{y_k=1}1_{h_k=-1} + \alpha 1_{y_k=-1}1_{h_k=1}) \\ &= \sum_{k=2}^{y} (1-\alpha)1_{h_k=-1} + \sum_{k=y+1}^{K} \alpha 1_{h_k=1} \\ &\geqslant (1-\alpha)(y-h)1_{h<y} + \alpha(h-y)1_{h>y} = L(y,h) \end{aligned}$$

□

Thus, we see that all we need is an accurate learning algorithm for binary problems with weighted 0-1 loss.

Monotonicity of the composite classifier. In the ordinal classification with monotonicity constraints we require that the classifier $h(\mathbf{x})$ must be monotone. One can simply show that if binary classifiers $h_k(\mathbf{x})$ are all monotone, then so is $h(\mathbf{x})$. Indeed, assume $\mathbf{x} \succeq \mathbf{x}'$. Then,

$$h(\mathbf{x}) = \sum_{k=2}^{K} h_k(\mathbf{x}) \geqslant \sum_{k=2}^{K} h_k(\mathbf{x}') = h(\mathbf{x}').$$

Hence, if for each k, $h_k(\mathbf{x})$ is a rule ensemble composed of monotone rules, then $h(\mathbf{x})$ is a monotone function.

5.2.4 Two-phase Procedure of Learning

We now give more details about how the binary classifiers $h_k(\mathbf{x})$ are learned. Let us fix some $k = \{2, \ldots, K\}$. We first transform the dataset $D = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ into the dataset $D_k = \{(\mathbf{x}_i, y_{ik}), i = 1, \ldots, n\}$, such that $y_{ik} = \operatorname{sgn}(y_i - k)$. Next, we perform monotone approximation of D_k and obtain the monotonized dataset $D'_k = \{(\mathbf{x}_i, y'_{ik}), i = 1, \ldots, n\}$, where y'_{ik} are new labels, corresponding to the monotone approximation $\hat{d}_{ik}$. Notice that it follows from the definition that y'_{ik} are monotone, i.e. $\mathbf{x}_i \succeq \mathbf{x}_j \to y'_{ik} \geqslant y'_{jk}$.

As soon as the data are monotonized, we train monotone rule ensemble $f_k(\mathbf{x})$ on D'_k such that it makes no errors on the dataset. In other words, $y'_{ik} f_k(\mathbf{x}_i) > 0$ for all $i = 1, \ldots, n$, and we say that such rule ensemble *separates* D'_k. This constitutes a *two-phase procedure* of learning the monotone rule ensemble: first, the dataset is monotonized, and then, the monotone rule ensemble separating the dataset is constructed.

To prove the validity of our procedure, we must show that a monotone rule ensemble separating the data always exists. It appears that the existence of a separating rule ensemble is strictly related to the monotonicity (consistency) of the dataset.

Theorem 5.3. *There exists a monotone rule ensemble $f(\mathbf{x})$ separating a dataset $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, $y_i \in \{-1, 1\}$, if and only if D is monotone, i.e. for each $i, j = 1, \ldots, n$, we have $\mathbf{x}_i \succeq \mathbf{x}_j \to y_i \geqslant y_j$.*

Proof. From the monotonicity of $f(\mathbf{x})$ it follows that $\mathbf{x}_i \succeq \mathbf{x}_j \to f(\mathbf{x}_i) \geqslant f(\mathbf{x}_j) \to \operatorname{sgn}(f(\mathbf{x}_i)) \geqslant \operatorname{sgn}(f(\mathbf{x}_j))$. Moreover, $f(\mathbf{x})$ separates D so that we have $\operatorname{sgn}(y_i f(\mathbf{x}_i)) = \operatorname{sgn}(y_j f(\mathbf{x}_j)) = 1$. Since $y_i \in \{-1, 1\}$, it follows that $y_i \geqslant y_j$, so D is monotone.

Assume D is monotone. Then, for each $\mathbf{x}_i$ such that $y_i = 1$ we construct a rule $\Phi(\mathbf{x}) = 1_{\mathbf{x} \succeq \mathbf{x}_i}$ with $a_i = 1$; this is a valid decision rule. Similarly, for each $\mathbf{x}_i$ with $y_i = -1$ we construct a rule $\Phi(\mathbf{x}) = -1_{\mathbf{x} \preceq \mathbf{x}_i}$ with $a_i = 1$. Since D is monotone, rules with $\Phi_i(\mathbf{x}) \geqslant 0$ cover only the objects with $y_j = 1$, while rules with $\Phi_i(\mathbf{x}) \leqslant 0$ cover only the objects with $y_j = -1$. Hence, rule ensemble makes no errors on D. □

Since D'_k is monotone, we are guaranteed that a separating monotone rule ensemble exists. Moreover, to create separating rule ensemble we *must* first monotonize the data.

5.2.5 Consistency and Generalization Bounds for the Two-phase Procedure

Strong consistency. This section brings theoretical justification for the two-phase procedure of learning monotone ensembles. We start with showing that for a very large class of monotonically constrained probability distributions, including attributes with both discrete and continuous domains, the two-phase procedure is strongly consistent.

Theorem 5.4. *Let $Y = \{1, \ldots, K\}$ and consider minimizing the linear loss. Assume $P(\mathbf{x}, y)$ is monotonically constrained and let $X = V \times \mathbb{R}^{m_0}$, where V is finite and $m_0 \leqslant m$. Assume $P(\mathbf{x})$ has density. Then the classifier $h(\mathbf{x})$ returned by the two-phase procedure is strongly consistent, i.e.*

$$R_n(h) \stackrel{n\to\infty}{\longrightarrow} R^* = R(h^*)$$

with probability one, where $R_n(h)$ is the risk of h when trained on the dataset of size n and h^* is a Bayes classifier.

Proof. Fix $k = 2, \ldots, K$. The monotone rule ensemble $h_k(\mathbf{x})$ separates set D'_k and is monotone, which implies that $h_k(\mathbf{x})$ is a valid monotone extension of the monotone approximation $y'_{ik}, i = 1, \ldots, n$. Thus, from Theorem 3.11 we have

$$R_n(h_k) \stackrel{n\to\infty}{\longrightarrow} R(h^*_k)$$

with probability one. This event holds for all $k = 2, \ldots, K$ simultaneously also with probability one (because K is finite). From Theorem 5.2 we have:

$$R^* \leqslant R_n(h) \leqslant \sum_{k=2}^{K} R_n(h_k) \stackrel{n\to\infty}{\longrightarrow} \sum_{k=2}^{K} R(h^*_k) = R^*,$$

which ends the proof. □

Theorem 5.4 shows that the two-phase method achieves asymptotically the smallest possible risk, but it does not tell anything about the rate of convergence or about any non-asymptotic error bounds. The rest of this section is devoted to bounding the deviations of the two-phase procedure from the Bayes risk in terms of the margin achieved on the monotonized dataset.

Margin theorem for uneven misclassification costs. We start our analysis with extending the margin bound (5.7) into the case of weighted 0-1 loss. Consider the binary class case $Y = \{-1, 1\}$ and real-valued function $f(\mathbf{x})$. Fix some margin value γ and define the violation of the margin for object $\mathbf{x}_i$ as event $\{y_i f(\mathbf{x}_i) \leqslant \gamma w_{y_i}\}$, where $w_{-1} = \alpha$ and $w_1 = 1 - \alpha$, as usual. Notice that this definition incorporates the uneven costs of misclassification into the margin: it effectively increases the margin for more penalized class and hence prompts the classifier to predict this class more often. Similar procedure has been used by Karakoulas and Shawe-Taylor (1999) to derive AdaUBoost, boosting algorithms for uneven costs of misclassification. Let us also denote:

$$R^{\gamma}_{\text{emp}}(f) = \frac{1}{n} \sum_{i=1}^{n} w_i 1_{y_i f(\mathbf{x}_i) \leqslant \gamma w_{y_i}} \tag{5.15}$$

for the fraction of weighted margin errors and:

$$R(f) = \mathbb{E}[w_y 1_{yf(\mathbf{x})<0}]$$

for the expected weighted 0-1 loss (risk). The following theorem holds:

Theorem 5.5. *Let $f(\mathbf{x}) = \sum_j a_j b_j(\mathbf{x})$ be the convex combination of classifiers $b_j(\mathbf{x}) \in \{-1, 1\}$, i.e. a_j are non-negative and $\sum_j a_j = 1$. Then, with probability $1-\delta$, for every distribution $P(y, \mathbf{x})$, the following inequality holds:*

$$R(f) \leqslant \inf_{\gamma \in (0,1]} \left[R^{\gamma}_{\text{emp}}(f) + M \left(\sqrt{\frac{d}{n\gamma^2}} + \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right) \right], \tag{5.16}$$

where d is the Vapnik-Chervonenkis dimension of the base classifier and M is some universal constant.

The proof closely follows the proof for the ordinary (symmetric) 0-1 loss and is given in the appendix. The main difference between the asymmetric and symmetric case is in the choice of the contraction functions. Notice that the asymmetric margin leads to the contractions $\gamma\phi^1(x)$ and $\gamma\phi^{-1}(x)$ which lead, in turn, to the bound $1/\gamma$. If we used symmetric margin, we would obtain contractions $\alpha\gamma\phi^1(x)$ and $(1-\alpha)\gamma\phi^{-1}(x)$ which would lead to the bound $1/(\min\{\alpha, 1-\alpha\}\gamma)$. If α were close to 0 or 1, such a bound would become very loose. This motivates our choice of the asymmetric margin in the algorithm.

Generalization bounds for the two-phase method. We are now ready to give our main theorem for the two-phase method. The theorem shows that the accuracy of the ensemble with large margin remains close to the accuracy of the Bayes classifier. There are four problems which are to be addressed:

1. Single rule returns one of three values $\{-1, 0, 1\}$, contrary to the ordinary classifier which returns one of two values $\{-1, 1\}$. To cope with such three-valued classifiers, we will show that there always exists an ensemble of ordinary base classifiers having their outputs in the set $\{-1, 1\}$, which is equivalent to the rule ensemble and has at least the same value of the margin.

2. We must extend the theorem to the multi-class problem with linear loss. We will do it by bounding the linear loss of the composite classifier in terms of sum of the losses of binary classifiers.

3. The ensemble is learned on the monotonized data, which are not i.i.d. anymore. We will by-pass this problem by noticing that the ensemble learned on monotonized data without misclassification errors makes on the original data no more errors than the number of relabeled objects.

4. In margin theorems, one compares the fraction of margin errors to the real accuracy of the ensemble. Here we compare the accuracy of the ensemble with the Bayes risk by noticing that with high probability the fraction of relabeled objects does not exceed much the Bayes risk.

Let $L(y, k)$ be the linear loss (5.12) and define $R(h) = \mathbb{E}[L(y, h(\mathbf{x})]$ to be the expected linear loss of the composite classifier (returned by the two-phase method). The following theorem holds:

Theorem 5.6. *Assume $P(\mathbf{x}, y)$ is monotonically constrained. Let $h(\mathbf{x})$ be the classifier returned by the two-phase method. Let $f_k(\mathbf{x})$, $k = 2, \ldots, K$, be the k-th rule ensemble trained on the monotonized dataset D'_k, such that there exists $\gamma_k > 0$ for which $y'_{ik} f_k(\mathbf{x}_i) \geqslant \gamma_k w_{y'_{ik}}$ for all $i = 1, \ldots, n$ (i.e. $f_k(\mathbf{x})$ achieves hard margin γ_k on the dataset D'_k). Then, with probability at least $1 - \delta$:*

$$R(h) \leqslant R^* + M \left(2(K-1)\sqrt{\frac{\log \frac{2(K-1)}{\delta}}{n}} + \sqrt{\frac{m}{n}} \sum_{k=2}^{K} \frac{1}{\gamma_k} \right).$$

for some universal constant M.

Theorem 5.6 states that an ensemble with a high value of the hard margin on the monotonized data performs not much worse than the Bayes classifier.

5.3 Linear Programming Rule Ensembles (LPRules)

In the previous section, we presented a two-phase procedure of learning rule ensembles. First, the K-class problem is converted into $K-1$ binary problems, such that in the k-th problem, objects

with class labels at least k are discriminated from objects with class labels at most $k-1$, leading to the training set D_k. Next, we monotonize D_k to obtain consistent dataset D'_k and rules are trained on D'_k such that all objects from the training set are correctly classified and the ensemble achieves hard margin γ_k on the dataset. We bounded the difference between the accuracy of the classifier made in such a way and the accuracy of the Bayes classifier in terms of the margin γ_k. This suggests that ensembles achieving large margin on the dataset have better generalization power.

In this section, we show how to obtain an ensemble with a high margin value. Our algorithm is based on the LPBoost framework, which is known to directly maximize the margin. As pointed out in (Demiriz et al., 2001), LPBoost ends at optimal solution with the highest possible value of the margin. On the other hand, solutions produced by LPBoost are very sparse. It is a very desirable property in our case, because a compact rule ensemble is much easier to interpret.

5.3.1 Rule Induction

Fix k and consider the k-th binary classification problem. As the dataset D'_k is monotonized, it follows from Theorem 5.3 that it is separable by the set of monotone rules. Hence, we can use the "hard margin" linear program (5.8). After translating the program into the rule framework and adapting it to the problem of searching for the asymmetric margin (with uneven costs of misclassification), we have:

$$\begin{aligned} \max: \ & \rho \\ \text{subject to: } & y_i \textstyle\sum_{j=1}^{J} a_j \Phi_j(\mathbf{x}_i) \geqslant w_{y_i}\rho \qquad i = 1, \ldots, n, \\ & \textstyle\sum_{j=1}^{J} a_j = 1, \\ & a_j \geqslant 0, \end{aligned} \tag{5.17}$$

where $w_{-1} = \alpha$, $w_1 = 1 - \alpha$ and the sums are over all possible rules. As there are at most n possible values that objects from D'_k can take on each attribute, the number of rules J can reach $2n^m$ in the worst case. Therefore, we are not able to solve the problem directly but we can use a column generation method in the dual program, similarly as described in Section 5.1.4, but with one exception. Demiriz et al. (2001) noticed that pushing the weights away from zero increases the stability of the algorithms. Following this observation, we add a set of constraints $w_{y_i} q_i \geqslant \frac{\kappa}{n}$ parametrized by a nonnegative value κ. Notice that $\kappa \leqslant 1$ in order to keep the dual feasible. Hence, the "hard margin" dual takes the form:

$$\begin{aligned} \min: \ & \beta \\ \text{subject to: } & \textstyle\sum_{i=1}^{n} q_i y_i \Phi_j(\mathbf{x}_i) \leqslant \beta \qquad j = 1, \ldots, J, \\ & \textstyle\sum_{j=1}^{J} w_{y_i} q_i = 1, \\ & w_{y_i} q_i \geqslant \tfrac{\kappa}{n} \qquad i = 1, \ldots, n. \end{aligned} \tag{5.18}$$

The procedure of rule induction is described as Algorithm 5.3. Notice that we need a linear programming solver to obtain a new weight vector in each iteration. We also need a procedure for finding the rule conditions Φ_j with a high value $\sum_{i=1}^{n} q_i y_i \Phi_j(\mathbf{x}_i)$. This value will be called *edge* (Breiman, 1998) and will be denoted by $E_q(j)$. The edge corresponds to the rescaled weighted zero-one error on the training set. Thus, in each iteration we try to find a rule with the highest possible accuracy according to the distribution with weights q_i. This is very similar to the way in which base classifiers are generated in other boosting algorithms. Notice that the minimum is always reached in a finite number of steps when for all rules we have $E_q(j) \leqslant \beta$.

Algorithm 5.3: Linear Programming Rule Ensembles (LPBoost).

input : separable set of training examples D.
output: rule ensemble $\{\Phi_1, \ldots, \Phi_T\}$ with weights $\{a_1, \ldots, a_T\}$.
$optimum := false$;
$\mathcal{J}_0 := \emptyset$;
$q_i := \frac{1}{n}, i = 1, \ldots, n$;
repeat
  $j_{\max} := \arg\max_{j=1,\ldots,J} \sum_{i=1}^{n} q_i y_i \Phi_j(\mathbf{x}_i)$;
  if $\sum_{i=1}^{n} q_i y_i \Phi_{j_{\max}}(\mathbf{x}_i) \leqslant \beta$ **then**
    $optimum := true$;
  end
  else
    $\mathcal{J}_0 := \mathcal{J}_0 \cup \{j_{\max}\}$;
    Solve the problem:

$$\begin{aligned} \min: & \sum_{i=1}^{n} \beta \\ \text{subject to:} & \sum_{i=1}^{n} q_i y_i \Phi_j(\mathbf{x}_i) \leqslant \beta && j \in \mathcal{J}_0, \\ & \sum_{j=1}^{J} w_{y_i} q_i = 1, \\ & w_{y_i} q_i \geqslant \tfrac{\kappa}{n} && i = 1, \ldots, n. \end{aligned}$$

    to obtain new weight vector q_i, $i = 1, \ldots, n$ and β;
  end
until *optimum = true;*
Obtain a_j, $j \in \mathcal{J}_0$ from the dual variables, choose from $\mathcal{J}_0$ a subset $\mathcal{T}$ of rules with nonzero weights and return as the final rule ensemble.

5.3.2 Single Rule Generation

Let us now focus on the main aspect of Algorithm 5.3 which is efficiently executing the operation:

$$\arg\max_{j=1,\ldots,J} \sum_{i=1}^{n} q_i y_i \Phi_j = \arg\max_{j=1,\ldots,J} E_q(j).$$

Since the number of rules is very large, one cannot do arg max by checking the edges of all rules, thus, it is reasonable to use a heuristic procedure instead. The procedure aims at finding the rule with a high value of the edge. In each iteration of the algorithm, one runs the procedure twice, first generating an upward rule and next a downward rule. Then, rule with a higher edge is chosen, while the other rule is discarded. The procedure for generating upward and downward rules is very similar. The only difference is the set of allowed conditions ($x_j \geqslant s_j$ for upward, and $x_j \leqslant s_j$ for downward rules) and the class which we name to be *positive*. When upward rule is generated, objects with $y = 1$ are positive, while for downward rule, positive objects belong to the class $y = -1$.

Description of the procedure. The procedure is presented as Algorithm 5.4. We start with the most general rule, covering the whole X. Then, conditions are added subsequently until the rule covers only positive examples. The quality of each candidate condition ϕ is defined as the ratio of the sum of weights of negative examples to the sum of weights of positive examples removed by adding the condition. In other words, let $\mathcal{P} \subseteq \{1, \ldots, n\}$ be a set of positive examples, let Φ be a set of conditions before adding ϕ, while Φ' is a set of conditions after adding ϕ. Then, the quality

Algorithm 5.4: Rule Generation.

input : weight vector $\{q_1, \ldots, q_n\}$,
set of positive examples $\mathcal{P} \subseteq \{1, \ldots, n\}$.
output: Rule conditions Φ.
$t := 0$;
$\Phi_t := \emptyset$;
repeat
  choose condition ϕ with the highest quality $Q(\phi)$ defined by (5.19);
  $\Phi_{t+1} := \Phi_t \cup \phi$;
  $t := t + 1$;
until $\forall_{i \notin \mathcal{P}} \Phi_t(\mathbf{x}_i) = 0$.
Choose from $\{\Phi_0, \ldots, \Phi_t\}$ a rule with the highest edge, $\Phi_{\max}$;
Remove from $\Phi_{\max}$ those conditions, which removal does not decrease the edge;
return $\Phi_{\max}$;

of candidate condition ϕ is defined as:

$$Q(\phi) = \frac{\sum_{i \notin \mathcal{P}} q_i 1_{\Phi(\mathbf{x}_i) \neq \Phi'(\mathbf{x}_i)}}{\sum_{i \in \mathcal{P}} q_i 1_{\Phi(\mathbf{x}_i) \neq \Phi'(\mathbf{x}_i)}}. \tag{5.19}$$

If the denominator is zero, the quality is infinite. We assume that when we compare two conditions with infinite qualities, the one with higher numerator is chosen ("more infinite" one). This assumption essentially simplifies the notation, as we do not need to handle the case with zero denominator separately. The quality (5.19) has a very intuitive meaning: we always choose a condition which decreases the number of covered positive examples as little as possible and decreases the number of covered negative examples as much as possible. Notice that the edge of the rule increases only if $Q(\phi) > 1$. Nevertheless, we keep adding conditions even when $Q(\phi) < 1$, until the rule covers positive examples only. The reason for this is that we keep in the memory a rule with the highest edge encountered so far, so we can always roll back the procedure and return to this rule. On the other hand, adding conditions until the rule covers no negative examples helps to avoid local minima of the edge.

Notice that we use a ratio measure (5.19) rather than the difference between positive and negative examples (which precisely corresponds to the edge of the rule). This is due to the fact that a ratio measure prefers more prudent steps, i.e. conditions chosen using $Q(\phi)$ remove smaller numbers of examples. To make this clear, suppose we choose between two conditions: ϕ_1 removes 2 positive and 3 negative examples, while ϕ_2 removes 8 positive and 6 negative examples from the rule. Although ϕ_2 increases the edge by 2, while ϕ_1 by 1, the ratio measure $Q(\phi)$ prefers ϕ_1, because it corresponds to a more gentle step, with possibility of improvement in later steps.

Having chosen the condition with the highest edge, denoted by $\Phi_{\max}$, we enter a "pruning" phase which consists in removing spare conditions, i.e. conditions which do not decrease the edge. This makes the rule simpler (having less conditions) which, in turn, increases its generalization ability.

Finally, notice that the rule generation procedure described above can be substituted by any other reasonable procedure which is capable of finding a rule which minimizes a weighted error on the training set. The main Algorithm 5.3 remains unchanged.

Complexity issues. The computational complexity of the two-phase procedure is rather poor in the worst case, because solving the linear program may take a long time (yet still polynomial

in n and T). Moreover, the monotone approximation has time complexity $O(n^3)$. Nevertheless, both procedures work very fast on the real datasets, much below the worst-case time. An additional speed-up is achieved by using all the methods for reducing the complexity of the monotone approximation, described in Chapter 3.

The rule generation procedure is linear in n, therefore it scales very well with the size of the dataset. The scaling behavior is theoretically quadratic in m in the worst case, when the number of conditions is comparable to the number of attributes. In real cases, however, the number of conditions is much smaller than the number of attributes, which makes the procedure scalable also with m.

5.4 Sigmoid Loss Monotone Rule Ensembles (MORE)

We present in this section another approach to rule induction. The algorithm has been introduced in (Dembczyński et al., 2008b) and was the first boosting-based algorithm for ordinal classification with monotonicity constraints. Therefore, it was simply named *monotone rule ensembles* and abbreviated MORE. Here, we will refer to the algorithm as *sigmoid loss monotone rule ensembles*, however we keep the original abbreviation (MORE).

The algorithm was not motivated by the margin theorem presented in Section 5.2.5. It is rather motivated by statistical considerations described by Friedman et al. (1998) and developed by Friedman and Popescu (2004). Moreover, it uses a specific, non-convex approximation of the zero-one loss function – the sigmoid loss. The algorithm also works by transforming the general K-class problem into $K-1$ binary subproblems, similarly as it was presented in Section 5.2. The main difference is the method of combining binary classifiers into the multi-class classifier, which will be described below.

5.4.1 Combining Binary Classifiers

We start with presenting a different method of combining $K-1$ binary classifiers into a single K-class classifier, originally used in MORE (Dembczyński et al., 2008b). Let us assume that each binary classifier has the form $h_k(\mathbf{x}) = \operatorname{sgn}(f_k(\mathbf{x}))$, where $f_k(\mathbf{x})$ is a real-valued function (e.g. ensemble of classifiers) 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 $\operatorname{sgn}(f_k(\mathbf{x}))$. Then, we could take advantage of additional information by using the value of $f_k(\mathbf{x})$ instead of merely the sign. 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:

$$f_{k+1}(\mathbf{x}) \leqslant f_k(\mathbf{x}), \tag{5.20}$$

because we are always more certain about classifying object to the set $\{k, \ldots, K\}$ than to its subset $\{k+1, \ldots, K\}$. If (5.20) holds, then the classification procedure $h(\mathbf{x})$ is doubtless: we seek for k, for which the sequence $f_k(\mathbf{x}), k = 2, \ldots, K$, changes the sign and classify $\mathbf{x}$ to the class k. This is equivalent to comprehensively writing $h(\mathbf{x}) = 1 + \sum_{k=2}^{K} 1_{f_k(\mathbf{x}) \geqslant 0}$.

However, binary problems are solved independently, so we cannot guarantee that such constraints hold in each case. We deal with the violation of the constraints (5.20) using the isotonic regression in the following way. Fix $\mathbf{x}$ and notice that from (5.20) 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 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 exactly the problem of isotonic regression, as described in Section 1.3.3. It can be thought of as "monotonizing" the function $f_k(\mathbf{x})$ violating the constraints (5.20).

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:

$$vote_k(\mathbf{x}) = \sum_{l=2}^{k} f_l(\mathbf{x}) \tag{5.21}$$

The following theorem holds:

Theorem 5.7. *Consider the classifier:*

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

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 $k \leqslant h$ it holds $vote_h \geqslant vote_k$. To show that $b = h$, it is enough to show that if $k > b$ then $vote_b > vote_k$ and if $k \leqslant b$ then $vote_b \geqslant vote_k$.

Let us define $G_+ = \{k \colon g_k \geqslant 0\}$ 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 (Dykstra et al., 1999), which states that for every $k = 2, \ldots, K$ it holds:

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

Suppose $k \leqslant b$. Then:

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

so that $vote_b \geqslant vote_k$. 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. □

We would like to have the monotonicity property of the classifier created according to Theorem 5.7, i.e. if $\mathbf{x} \succeq \mathbf{x}'$ then $h(\mathbf{x}) \geqslant h(\mathbf{x}')$. The following theorem gives sufficient conditions for monotonicity:

Theorem 5.8. *For each $k = 2, \ldots, K$, let $f_k(\mathbf{x})$ be a monotone function, i.e.: $\mathbf{x} \succeq \mathbf{x}' \rightarrow f_k(\mathbf{x}) \geqslant f_k(\mathbf{x}')$. Then, the classifier $h(\mathbf{x})$, obtained by choosing the class label with the highest vote (5.21), is a monotone function.*

Proof. Choose any $r, s \in \{2, \ldots, K\}$ such that $r \geqslant s$. Then:

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

It follows from the definition that if $k \leqslant h(\mathbf{x}')$ then $vote_{h(\mathbf{x}')}(\mathbf{x}') - vote_k(\mathbf{x}') \geqslant 0$. But this also means that $vote_{h(\mathbf{x}')}(\mathbf{x}) - vote_k(\mathbf{x}) \geqslant 0$ which, in turn, means that $h(\mathbf{x}) \geqslant h(\mathbf{x}')$. □

We presented an alternative method of combining binary classifiers which takes into account the prediction confidence of each classifier. The main problem of this method is that one cannot bound the loss of the composed classifiers by means of the votes of binary classifiers, as it was done in Theorem 5.2. This convinced us to abandon this method in case of using with LPRules. Nevertheless, although the loss of a composite classifier cannot be simply bounded, nothing prevents this method to work well in practice on real-life problems. Therefore, this method was finally used with MORE.

5.4.2 Rule induction with Sigmoid Loss

In each of the binary subproblems, rule induction is performed by minimizing the weighted 0-1 loss (linear loss) function on the dataset. For $Y = \{-1, 1\}$, 0-1 loss can be expressed as $L_{0-1}(yf(\mathbf{x})) = w_y 1_{yf(\mathbf{x})<0}$. This function, however, is neither smooth nor differentiable. Therefore, following the arguments raised in Section 5.1.2, we approximate 0-1 loss with the sigmoid function (5.6):

$$L_{\text{sigm}}(yf(\mathbf{x})) = w_y \frac{1}{1 + e^{yf(\mathbf{x})}}. \tag{5.22}$$

Thus, we minimize the following empirical risk:

$$R_{\text{emp}}(f) = \sum_{i=1}^{n} L_{\text{sigm}}(y_i f(\mathbf{x}_i)) \tag{5.23}$$

However, finding a set of rules minimizing (5.23) is computationally hard. That is why we follow the boosting strategy, i.e. the rules are added one by one, greedily minimizing (5.23). We start with the "default" rule defined as:

$$a_0 = \arg\min_a R_{\text{emp}}(a) = \arg\min_a \sum_{i=1}^{n} L_{\text{sigm}}(ay_i). \tag{5.24}$$

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 $f_{t-1}(\mathbf{x})$ be a classification function after $t-1$ iterations, consisting of the first $t-1$ rules and the default rule. In the t-th iteration, a decision rule can be obtained by solving:

$$(a_t, \Phi_t(\mathbf{x})) = \arg\min_{\Phi, a} R_t(\Phi, a) \tag{5.25}$$

where we defined:

$$R_t(\Phi, a) = \sum_{i=1}^{n} L_{\text{sigm}}(y_i(f_{t-1}(\mathbf{x}_i) + a\Phi(\mathbf{x}_i))). \tag{5.26}$$

Monotonization of the data. Similarly as in LPRules, we are able to do the monotone approximation of the k-th dataset D_k and feed the k-th learning algorithm with the monotonized data D'_k. Now, however, the monotonization process is not necessary as we do not aim at separating the dataset with rules. Nevertheless, the process of rule induction is a greedy optimization procedure

Algorithm 5.5: Monotone Rule Ensemble (MORE).

input : set of training examples D,
T – number of rules to be generated.
output: monotone rule ensemble $\{r_1, \ldots, r_T\}$.
$f_0(\mathbf{x}) := \arg\min_{\pm a} R_0(1, \pm a)$;
for $t = 1$ *to* T **do**
  $\Phi_t = \arg\min_{\Phi} R_t(\Phi, a)$;
  $r_t(\mathbf{x}) = a\Phi_t(\mathbf{x})$;
  $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + r_t(\mathbf{x})$;
end

and monotonization may be helpful for the process to converge faster. This results in a smaller number of rules and better interpretability of the model. What is more, we did not observe any significant deficiency in accuracy, when monotonization is applied.

Since the monotone approximation usually takes much less time than the process of rule induction, using it as a preprocessing step does not cost much computational effort, therefore it was incorporated into the rule induction procedure.

5.4.3 Single Rule Generation

The exact solution of (5.25) is still computationally hard, because we need to determine the optimal $\Phi_t(\mathbf{x})$ and a_t simultaneously. Therefore, we restrict our algorithm to the case, in which all rules have weights of the same magnitude, equal to a, i.e. we only allow $a_t = a$. The magnitude a is a fixed parameter of the algorithm. With such a restriction, (5.25) becomes:

$$\Phi_t(\mathbf{x}) = \arg\min_{\Phi} R_t(\Phi, a), \tag{5.27}$$

and it requires calculating only two loss values at points $f_{t-1}(\mathbf{x}_i)$ and $f_{t-1}(\mathbf{x}_i) \pm a$ ($+a$ for upward rules and $-a$ for downward rules) for every object $\mathbf{x}_i$. The default rule is solved in a similar way by restricting the minimization of (5.24) to the values $a_0 \in \{-a, a\}$. In each subsequent iteration, problem (5.27) can be solved via a heuristic procedure for rule generation, defined as follows.

The procedure generates first the upward rule and next the downward rule. Then, both rules are compared and the one with lower empirical risk is chosen, while another one is discarded. The procedures for generating the upward and downward rules are almost identical, therefore they will be presented simultaneously:

- At the beginning, Φ_t is empty (no elementary conditions are specified), i.e. $\Phi_t(\mathbf{x}) \equiv 1$.
- In each step, an elementary condition $x_j \geqslant s_j$ (for upward rule) or $x_j \leqslant s_j$ (for downward rule) is added to Φ that minimizes $R_t(\Phi, a)$. 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 $x_j^{(i-1)} \geqslant x_j^{(i)}$, for $i = 2, \ldots, n$. Each elementary condition of the form $x_j \geqslant s_j$ (for upward rule) or $x_j \leqslant s_j$ (for downward rule) for each $s_j = \frac{x_j^{(i-1)} + x_j^{(i)}}{2}$ is tested.
- The previous step is repeated until $R_t(\Phi, a)$ 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. The procedure for finding Φ_t resembles

the way the decision trees are generated. Here, we look only for one path from the root to the leaf. Moreover, let us notice that minimal value of $R_t(\Phi, a)$ is a natural stop criterion in building a single rule. The whole procedure is presented as Algorithm 5.5.

5.4.4 Analysis of the Step Length

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

$$f(\mathbf{x}) = \pm a + \sum_{t=1}^{T} a\Phi_t(\mathbf{x}) = a\left(\pm 1 + \sum_{t=1}^{T} \Phi_t(\mathbf{x})\right) = a\tilde{f}(\mathbf{x}),$$

where $\tilde{f}(\mathbf{x}) = \pm 1 + \sum_{t=1}^{T} \Phi_t(\mathbf{x})$. Then:

$$L_{\text{sigm}}(yf(\mathbf{x})) = \frac{1}{1 + \exp(ay\tilde{f}(\mathbf{x}))}.$$

Thus, small values of a cause the loss function to broaden and the changes in the slope of the function are smaller (the loss becomes similar to the linear function). On the other hand, large values of a cause the sigmoid loss to become similar to the 0-1 loss. In general, large values of a correspond to a larger complexity (Mason et al., 1999), 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 5.9. *Minimization of (5.27) for any twice differentiable loss function $L(yf(\mathbf{x}))$ and any a is equivalent to the minimization of:*

$$R_t(\Phi, a) = \sum_{i \in R_-} w_i^t + \frac{1}{2} \sum_{\Phi(\mathbf{x}_i)=0} \left(w_i^t - a v_i^t\right). \tag{5.28}$$

where we define:

$$R_- = \{i\colon\ \Phi(\mathbf{x}_i) y_i < 0\} \tag{5.29}$$

$$w_i^t = -\frac{\partial}{\partial u} L(u)\Big|_{u = y_i f_{t-1}(\mathbf{x}_i)} \tag{5.30}$$

$$v_i^t = \frac{1}{2}\frac{\partial^2}{\partial u^2} L(u)\Big|_{u = y_i f_{t-1}(\mathbf{x}_i) \pm \gamma_i a y_i}. \tag{5.31}$$

for some $\gamma_i \in [0, 1]$, where the "plus" sign in (5.31) is for rules $\Phi(\mathbf{x}_i) \geqslant 0$, while the "minus" sign is for rules $\Phi(\mathbf{x}_i) \leqslant 0$.

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 $L_i = L(y_i(f_{t-1}(\mathbf{x}_i)))$, using (5.30)-(5.31) and substituting $u = y_i f_{t-1}(\mathbf{x}_i)$ and $z = \Phi(\mathbf{x}_i) a y_i$, we have for every $\mathbf{x}_i$ such that $\Phi(\mathbf{x}_i) \neq 0$:

$$L(y_i(f_{t-1}(\mathbf{x}_i) + a\Phi(\mathbf{x}_i))) = L_i - a y_i \Phi(\mathbf{x}_i) w_i^t + a^2 \Phi^2(\mathbf{x}_i) v_i^t.$$

Thus, the empirical risk becomes:

$$R_t(\Phi, a) = \sum_{\Phi(\mathbf{x})_i \neq 0} \left(L_i - a y_i \Phi(\mathbf{x}_i) w_i^t + a^2 v_i^t\right) + \sum_{\Phi(\mathbf{x})_i = 0} L_i.$$

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

$$R_t(\Phi, a) = \sum_{\Phi(\mathbf{x})_i \neq 0} -ay_i\Phi(\mathbf{x}_i)w_i^t + a^2 v_i^t = \sum_{i \in R_-} aw_i^t - \sum_{i \in R_+} aw_i^t + \sum_{\Phi(\mathbf{x})_i \neq 0} a^2 v_i^t,$$

where $R_+ = \{i\colon\ \Phi(\mathbf{x}_i)y_i > 0\}$ and R_- is defined in (5.29). 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 $\sum_{\Phi(\mathbf{x}_i)\neq 0} = \sum_{i=1}^{n} - \sum_{\Phi(\mathbf{x}_i)=0}$ to obtain:

$$R_t(\Phi, a) = \sum_{i=1}^{n}(a^2 v_i^t - aw_i^t) + 2a \sum_{i \in R_-} w_i^t + a \sum_{\Phi(\mathbf{x}_i)=0} (w_i^t - av_i^t),$$

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

Thus, a establishes a trade-off between misclassified and unclassified examples. Values v_i^t 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 a increases, uncovered examples satisfying $y_i f_{t-1}(\mathbf{x}_i) > 0$ ("correctly classified") are penalized less, while the penalty for uncovered "misclassified" examples ($y_i f_{t-1}(\mathbf{x}_i) < 0$) 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).

Appendix: Proofs of the Theorems

Proof of Theorem 5.5

Theorem 5.6. *Let $f(\mathbf{x}) = \sum_j a_j b_j(\mathbf{x})$ be the convex combination of classifiers $b_j(\mathbf{x}) \in \{-1, 1\}$, i.e. a_j are non-negative and $\sum_j a_j = 1$. Then, with probability $1 - \delta$, for every distribution $P(y, \mathbf{x})$ the following inequality holds:*

$$R(f) \leqslant \inf_{\gamma \in (0,1]} \left[R^{\gamma}_{\text{emp}}(f) + M \left(\sqrt{\frac{d}{n\gamma^2}} + \sqrt{\frac{\log \frac{1}{\delta}}{n}} \right) \right], \tag{5.32}$$

where d is the Vapnik-Chervonenkis dimension of the base classifier and M is some universal constant.

Proof. Our proof is very similar to the proof of Koltchinskii and Panchenko (2002). We follow more accessible version of the proof given by Lugosi (2002)[1] and describe here only the differences from the unweighted loss case.

Let us define two functions $\phi^1(x)$ and $\phi^{-1}(x)$ as follows:

$$\phi^1(x) = \begin{cases} 1 - \alpha & \text{if } x \leqslant 0 \\ 0 & \text{if } x \geqslant \gamma(1-\alpha) \\ 1 - \alpha - x/\gamma & \text{if } x \in (0, \gamma(1-\alpha)) \end{cases}$$

$$\phi^{-1}(x) = \begin{cases} \alpha & \text{if } x \leqslant 0 \\ 0 & \text{if } x \geqslant \gamma\alpha \\ \alpha - x/\gamma & \text{if } x \in (0, \gamma\alpha)) \end{cases}.$$

[1] Lugosi (2002) proves the theorem for a fixed value of γ, but it can be extended to all values of γ in the same way as in (Koltchinskii and Panchenko, 2002).

Observe that $w_y 1_{yf(\mathbf{x})<0} \leqslant \phi^y(yf(\mathbf{x})) \leqslant w_y 1_{yf(\mathbf{x})\leqslant\gamma w_y}$. Thus, we have:

$$\sup_{f\in\mathcal{F}}(R(f) - R^{\gamma}_{\text{emp}}(f)) \leqslant \sup_{f\in\mathcal{F}}\left(\mathbb{E}\phi^y(yf(\mathbf{x})) - \frac{1}{n}\sum_{i=1}^{n}\phi^{y_i}(y_i f(\mathbf{x}_i))\right),$$

where $\mathcal{F}$ is the class of all ensembles. Using the bounded difference inequality (McDiarmid, 1989; Devroye et al., 1996) with $c_i = n^{-1}\max\{\alpha, 1-\alpha\} \leqslant n^{-1}$ we bound the distance of the right-hand side from its expectation exactly as in (Lugosi, 2002). Define $z = yf(\mathbf{x})$ and $z_i = y_i f(\mathbf{x}_i)$. By symmetrization argument we have:

$$\mathbb{E}\sup_{f\in\mathcal{F}}\left(\mathbb{E}\phi^y(z) - \frac{1}{n}\sum_{i=1}^{n}\phi^{y_i}(z_i)\right) \leqslant \mathbb{E}\sup_{f\in\mathcal{F}}\left(\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\phi^{y_i'}(z_i') - \phi^{y_i}(z_i))\right)$$

where $\sigma_i, i = 1,\ldots,n$, are i.i.d. symmetric sign variables, and $(\mathbf{x}_i', y_i')$ forms a sample of size n, being independent of $(\mathbf{x}_i, y_i)$ and having the same distribution. Let us define $\phi(x) = \frac{\phi^1(x)+\phi^{-1}(x)}{2}$. Then, using:

$$\begin{aligned}\phi^{y_i'}(z_i') - \phi^{y_i}(z_i) \;=\; & (\phi^{y_i'}(z_i') - \phi(z_i')) - (\phi^{y_i'}(0) - \phi(0)) + (\phi(z_i') - \phi(z_i)) \\ & + (\phi^{y_i}(0) - \phi^{y_i'}(0)) - (\phi^{y_i}(z_i) - \phi(z_i)) - (\phi^{y_i}(0) - \phi(0))\end{aligned}$$

we bound:

$$\begin{aligned}\mathbb{E}\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\phi^{y_i'}(z_i') - \phi^{y_i}(z_i)) \leqslant & 2\mathbb{E}\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\phi^{y_i}(z_i) - \phi(z_i) - \phi^{y_i}(0) + \phi(0)) \\ + \mathbb{E}\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\phi(z_i') - \phi(z_i)) & + \mathbb{E}\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\phi^{y_i}(0) - \phi^{y_i'}(0)). \qquad (5.33)\end{aligned}$$

The last term on the right-hand side is equal to zero. The middle term can be bounded as in (Lugosi, 2002) by noticing that $\psi(x) = \gamma(\phi(x) - \phi(0))$ is a contraction. By noticing the symmetry $\phi^{y_i}(z_i) - \phi(z_i) = -(\phi^{-y_i}(z_i) - \phi(z_i))$ and exploiting the fact that σ_i are i.i.d. symmetric sign variables, the expression under sup in the first term on the right-hand side equals to $\phi^1(z_i) - \phi(z_i) - \phi^1(0) + \phi(0)$, which is also the contraction after multiplying by γ. Thus, in summary we bound:

$$\mathbb{E}\sup_{f\in\mathcal{F}}\left(\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\phi^{y_i'}(z_i') - \phi^{y_i}(z_i))\right) \leqslant \frac{4}{\gamma}\mathbb{E}\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i f(\mathbf{x}_i).$$

Notice that the bound is twice the bound of the symmetric case which, in turn, doubles universal constant M in comparison to the symmetric case. The rest of the proof proceeds exactly in the same way as in (Lugosi, 2002). □

Proof of Theorem 5.6

Theorem 5.7. *Assume $P(\mathbf{x}, y)$ is monotonically constrained. Let $h(\mathbf{x})$ be a classifier output by the two-phase method. Let $f_k(\mathbf{x})$, $k = 2,\ldots,K$, be k-th rule ensemble trained on the monotonized dataset D_k' such that there exists $\gamma_k > 0$ for which $y_{ik}' f_k(\mathbf{x}_i) \geqslant \gamma_k w_{y_{ik}'}$ for all $i = 1,\ldots,n$ (i.e. $f_k(\mathbf{x})$ achieves hard margin γ_k on the dataset D_k'). Then, with probability at least $1-\gamma$:*

$$R(h) \leqslant R^* + M\left(2(K-1)\sqrt{\frac{\log\frac{2(K-1)}{\delta}}{n}} + \sqrt{\frac{m}{n}}\sum_{k=2}^{K}\frac{1}{\gamma_k}\right).$$

for some universal constant M.

Proof. We start with a transformation of rules (which can abstain from the response) to classifiers with output in $\{-1,1\}$ and with well-defined VC dimension. Let $f(\mathbf{x}) = \sum_{t=1}^{T} a_t \Phi_t(\mathbf{x})$. With each rule $\Phi_t(\mathbf{x}), t = 1,\ldots,T$, we associate a *cone* $g_t(\mathbf{x}) = 2\Phi_t(\mathbf{x}) - 1$ for upward rules ($\Phi(\mathbf{x}) \geqslant 0$) and $g_t(\mathbf{x}) = 2\Phi_t(\mathbf{x}) + 1$ for downward rules ($\Phi(\mathbf{x}) \leqslant 0$). Notice that $g_t(\mathbf{x}) \in \{-1,1\}$ and either the set $\{g_t(\mathbf{x}) = 1\}$ or the set $\{g_t(\mathbf{x}) = -1\}$ is an axis-parallel cone, i.e. set of the form $\{\mathbf{x}\colon \mathbf{x} \succeq \mathbf{x}_0\}$ or of the form $\{\mathbf{x}\colon \mathbf{x} \preceq \mathbf{x}_0\}$ for some $\mathbf{x}_0$. The class of axis-parallel cones has VC dimension m (Devroye et al., 1996). We add to the ensemble one more cone $g_0(\mathbf{x}) = \operatorname{sgn}\left(\sum_{t\in T^+} a_t - \sum_{t\in T^-} a_t\right)$ which covers the whole X and the subsets T^+ and T^- correspond to the upward and downward rules, respectively. Then, one can easily show that:

$$\sum_{t=0}^{T} c_t g_t(\mathbf{x}) = \sum_{t=1}^{T} a_t \Phi_t(\mathbf{x}),$$

when $c_t = \frac{a_t}{2}$ for $t \geqslant 1$ and $c_0 = \frac{1}{2}|\sum_{t\in T^+} a_t - \sum_{t\in T^-} a_t|$. Hence, we see that each rule ensemble has an equivalent cone ensemble. Now, suppose rule ensemble is L_1-normalized, i.e. $\sum_{t=1}^{T} a_t = 1$. The corresponding cone ensemble must be divided by the sum of weights $\ell = \sum_{t=0}^{T} c_t$ to be normalized; let us denote such ensemble by $f'(\mathbf{x}) = f(\mathbf{x})/\ell$. Notice that:

$$\ell = \frac{1}{2}\left|\sum_{t\in T^+} a_t - \sum_{t\in T^-} a_t\right| + \sum_{t=1}^{T} \frac{\alpha_t}{2} \leqslant \sum_{t=1}^{T} a_t = 1,$$

so that if $yf'(\mathbf{x}) \leqslant \gamma$, then also $yf(\mathbf{x}) \leqslant \gamma$. In other words, the fraction of margin errors for the normalized cone ensemble upperbounds the fraction of margin errors for the normalized rule ensemble. Thus, we can prove the theorem for an ensemble of cones, which are ordinary classifiers with well-defined VC dimension, and the theorem will also hold for a rule ensemble.

Let $R^{\gamma}_{\text{emp}}(f_k)$ be defined as before in (5.15), as the weighted fraction of margin violations for the k-th monotone ensemble with normalized weights. Consider the k-th ensemble $f_k(\mathbf{x})$ trained on the monotonized data D'_k. The ensemble makes no margin errors on D'_k, i.e. for all i, it holds $y'_{ik} f_k(\mathbf{x}_i) \geqslant \gamma_k w_{y'_{ik}}$. But since the sets D_k and D'_k differ only on the relabeled objects, the ensemble makes on D_k a margin error $R^{\gamma_k}_{\text{emp}}$ not greater than:

$$R^{\gamma_k}_{\text{emp}}(f) \leqslant \Gamma = \frac{1}{n}\sum_{i=1}^{n} w_{y_{ik}} 1_{y_{ik} \neq y'_{ik}}.$$

But the upper bound Γ equals to the objective function of the binary monotone approximation problem (3.16), hence it is minimized by values y'_{ik} in the class of all monotone functions. From the assumption about the monotonically constrained distribution it follows, however, that the Bayes classifier $h^*_k(\mathbf{x})$ is monotone, which means that:

$$R^{\gamma_k}_{\text{emp}}(f) \leqslant \Gamma \leqslant \frac{1}{n}\sum_{i=1}^{n} w_{y_{ik}} 1_{y_{ik} \neq h^*_k(\mathbf{x}_i)} = \frac{1}{n}\sum_{i=1}^{n} L(y_{ik}, h^*_k(\mathbf{x}_i)),$$

where L stands for the linear loss. Notice that $\mathbb{E}[L(y_k, h^*_k(\mathbf{x}))] = R^*_k$, the Bayes risk in the k-th binary problem, so using Hoeffding's bound (Devroye et al., 1996) we can state that for every $\delta' \in (0,1)$:

$$P\left(\frac{1}{n}\sum_{i=1}^{n} L(y_{ik}, h^*_k(\mathbf{x}_i)) - R^*_k \geqslant \sqrt{\frac{\log\frac{1}{\delta'}}{2n}}\right) \leqslant \delta'.$$

From Theorem 5.5 we have with probability at most δ'':

$$R(f_k) \geqslant \inf_{\gamma\in(0,1]}\left[R^{\gamma}_{\text{emp}}(f_k) + M\left(\sqrt{\frac{m}{n\gamma^2}} + \sqrt{\frac{\log\frac{1}{\delta''}}{n}}\right)\right].$$

By setting $\delta' = \delta'' = \frac{\delta}{2(K-1)}$, we have with probability at most $\delta' + \delta'' = \frac{\delta}{K-1}$:

$$R(f_k) \geqslant R_k^* + M\left(\sqrt{\frac{m}{n\gamma_k^2}} + 2\sqrt{\frac{\log\frac{2(K-1)}{\delta}}{n}}\right). \tag{5.34}$$

To bound the risk in a general K-class case, notice that with probability at most δ there exists $k \in \{2, \ldots, K\}$ such that (5.34) holds. But this means that with probability at most δ:

$$\sum_{k=2}^{K} R(f_k) \geqslant \sum_{k=2}^{K} R_k^* + M\sum_{k=2}^{K}\left(\sqrt{\frac{m}{n\gamma_k^2}} + 2\sqrt{\frac{\log\frac{2(K-1)}{\delta}}{n}}\right).$$

We take advantage of the fact that:

$$R^* = \mathbb{E}[L(y, h^*(\mathbf{x})] = \sum_{k=2}^{K}\mathbb{E}[L(y_k, h_k^*(\mathbf{x}))] = \sum_{k=2}^{K} R_k^*,$$

and from Theorem 5.2 we have:

$$R(h) = \mathbb{E}[L(y, h(\mathbf{x})] \leqslant \sum_{k=2}^{K}\mathbb{E}[L(y_k, h_k(\mathbf{x}))] = \sum_{k=2}^{K} R(f_k).$$

Thus, we conclude that with probability at most δ:

$$R(h) \geqslant R^* + M\left(2(K-1)\sqrt{\frac{\log\frac{2(K-1)}{\delta}}{n}} + \sqrt{\frac{m}{n}}\sum_{k=2}^{K}\frac{1}{\gamma_k}\right).$$

□

Chapter 6

Computational Experiment

In this chapter, we verify the efficiency of our methods in the computational experiment. We also compare our methods with already existing approaches to ordinal classification with monotonicity constraints and analyze the results with use of the nonparametric statistical tests. The experiment is conducted on several real datasets, for which it is known that there exist monotone relationships between attribute values and class labels.

As the accuracy is not the only criterion of assessing the quality of the learning method, we also consider the interpretability of the models. The interpretabilty was actually the major point for which we deal with decision rules, therefore an analysis of this issue is conducted in this chapter.

6.1 Design of the Experiment

6.1.1 Datasets

We found 12 datasets, for which it is known that monotonicity constraints are present. 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 from Ben-David (1992, 1995) and were accompanied by the following descriptions:

- `ESL` (*employee selection*) dataset contains profiles of applicants for certain industrial jobs.
- `SWD` (*social workers decisions*) dataset contains real-world assessments of qualified social workers regarding the risk facing children if they stayed with their families at home.
- `LEV` (*lecturers evaluation*) dataset contains examples of anonymous lecturer evaluations, taken at the end of MBA courses.
- `ERA` (*Employee Rejection/Acceptance*) dataset was originally gathered during an academic decision-making experiment aiming at determining which are the most important qualities of candidates for a certain type of jobs.

Three datasets are related to the problem of house pricing:

- `Housing` dataset comes from the UCI repository (Asuncion and Newman, 2007) and concerns housing values in suburbs of Boston.
- `Windsor` dataset first appeared in (Koop, 2000) and concerns housing values in Windsor, Canada.

Table 6.1: Description of data sets 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

- `DenBosch` dataset was taken from (Daniels, 1999) and concerns housing values in small Dutch city Den Bosch; see (Daniels and Feelders, 2000) for overview.

From the 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 `Housing` and `Windsor` into four levels containing equal number of objects (i.e. quartiles of the price distribution), similarly as in (Feelders and Pardoel, 2003).

There are five other datasets taken from the UCI repository:

- `Wisconsin` breast cancer dataset.
- `Ljubljana` breast cancer dataset, in which some of the non-monotone attributes and attributes containing most of missing values, have been removed.
- `Car` evaluation data.
- `CPU` performance data, for which the class attribute was discretized into four levels, containing equal number of objects.
- `Balance` scale dataset.

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 to handle the missing values, which is included in our implementation, however, its description is beyond the scope of this thesis). The quantitative characteristics of the datasets are shown in Table 6.1.

6.1.2 Algorithms

In the experiment, we used ten algorithms in total, among which three were invented by us and introduced in this thesis. Four other methods are well known in the field of ordinal classification with monotonicity constraints and can be though of as the state of the art in this domain. The last three methods are ordinary classifiers which are not suited to the ordinal classification and do not take monotonicity constraints into account. We include these methods to assess whether incorporating domain knowledge about order and monotonicity gives any improvement in accuracy. Up to our knowledge, there was no such an extensive comparison of so many algorithms ever before in this field.

State-of-the-art methods. We considered four already existing methods to ordinal classification with monotonicity constraints:

- Ordinal Learning Model (`OLM`) with implementation obtained from Weka (Witten and Frank, 2005).
- Ordinal Stochastic Dominance Learner (`OSDL`), implementation was obtained from Weka, balanced version was used (as it was recommended by Cao-Van (2003) to give better results). The internal cross-validation for tuning the interpolation parameter was turned on, because algorithm was quite fast anyway.
- Isotonic Separation (`IsoSep`) was implemented by us, using Weka and linear programming solver *lp_solve* (Berkelaar, 2005), according to (Chandrasekaran et al., 2005), with absolute error cost matrix. As suggested, all the attributes were normalized before calculating the distance in the classification procedure.
- Rule induction with `VCDomLEM` algorithm obtained from J. Błaszczyński and M. Szeląg, as a part of the *Java Rough Sets* (JRS) library. The classification procedure used in the algorithm was the one described in (Błaszczyński et al., 2007). A variable consistency version was used, with consistency level 0.9.

The methods were described in detail in Section 1.3.

Our methods. Apart from already existing approaches, we tested the following three methods, introduced in this thesis:

- Multiple isotonic regression (`IsoReg`), using the heuristic algorithm of Burdakov et al. (2006). In order to estimate the probabilities outside the training set, we must use some extension of isotonic regresion (see Section 3.1.4). We used the extension defined by (3.7) with $\lambda = 0.5$. The classification was performed by taking the median of the distribution.
- Linear Programming Rule Ensemble (`LPRules`), using our implementation in Weka and linear programming solver *lp_solve*. `LPRules` does not have any parameters to set apart from a technical parameter κ (the smallest value of object's weight), which maintains the stability of the optimization steps. We observed that the performance of the algorithm remains the same when we change κ provided we keep it small but nonzero. Thus, we set $\kappa = 02$.
- Sigmoid-loss Monotone Rule Ensembles (`MORE`), using our implementation in Weka. Since we did not optimize other algorithms, we decided to choose standard parameters, setting the number of rules M to 50 (per each binary subproblem) and the scale (step length) a to 0.5. Notice that in the real applications, those values would be obtained by cross-validation, probably leading to better results.

Ordinary methods. We decided to use three ordinary classifiers in order to check whether incorporating domain knowledge about order and monotonicity gives any improvement in accuracy:

- `j48` is the Weka's implementation of famous tree induction algorithm C4.5 (Quinlan, 1993). Unfortunately, `j48` is designed to minimize 0-1 error and does 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 by Frank and Hall (2001).

This approach divides K-class ordinal problem into $K-1$ binary problems in a similar way as it was done in this thesis. Then, for each binary problem, a base classifier (`j48` in this case) is learned and the conditional probability of upward class union $P(y \geqslant k|\mathbf{x})$ is estimated. Plugging those estimates (instead of real probabilities) into the expression defining the Bayes classifier allows us to minimize any ordinal loss function. For instance, if we use the absolute error loss function, then the Bayes classifier is the median of conditional distribution, so the output of our classifier will be the median of the estimated distribution.

- `SVM` (Support Vector Machine) classifier proposed by Boser et al. (1992); Vapnik (1998). We used its implementation in Weka. A standard linear kernel was used and the complexity parameter was set to default value. Similarly as in the case of `j48`, we used a simple approach to ordinal classification, because `SVM` in its basic version is not suited to capture the order between the labels and minimizes 0-1 error rather than the mean absolute error.

- `NB` (Naïve Bayes), which estimates the probabilities $P(\mathbf{x}|y)$ and uses Bayes' rule to obtain $P(y|\mathbf{x})$. We plug those probabilities into the expression defining the Bayes classifier to output the predicted label. For instance, when dealing with absolute error loss function, our prediction is the median of the estimated distribution.

6.2 Experimental Results

Error estimation. The measure of error was chosen to be the mean absolute error (MAE). It reflects the ordinal nature of the problem by penalizing the classifier according to the difference between predicted and observed class labels. The error of each classifier was estimated by a 10-fold cross validation, repeated 10 times to improve the replicability of the experiment (so that the results of the experiment do not depend on particular train/test folds splits). Apart from the average error, its standard deviation was estimated from the measurements on each fold and each trial (100 measurements in total). To avoid underestimation, the standard deviation of error was estimated conservatively, by taking into account the dependence between the subsequent testing samples in the repeated cross-validation, as described in (Nadeau and Bengio, 2003). The results of both average error and standard deviation are given in Table 6.2.

Testing statistical significance. It was stressed (Bengio and Grandvalet, 2004; Nadeau and Bengio, 2003) that there is no good (unbiased) way of estimating the standard error of accuracy and using such estimates in significance test may give misleading, unreliable results. Therefore, we perform nonparametric significant test. To compare multiple classifiers on the multiple datasets, we follow (Demšar, 2006), and apply the Friedman test, which uses ranks of each algorithm to check whether all the algorithms perform equally well (null hypothesis). The ranks are calculated for each dataset and the lowest rank (1) means that an algorithm performed the best on this dataset, while the highest rank (10) means that an algorithm performed the worst. Then, the ranks are averaged for each classifier and Friedman statistics is computed:

$$\chi_F^2 = \frac{12d}{c(c+1)} \left(\sum_{j=1}^{c} \bar{r}_j^2 - \frac{c(c+1)^2}{4} \right),$$

where d is the number of datasets, c is the number of compared classifiers and $\bar{r}_j$ is the average rank of j-th classifier; χ_F^2 is distributed approximately according to χ^2 with $c-1$ degrees of freedom. Notice that Friedman statistics depends on results of the experiment only through the average ranks of the classifiers – the particular values of the mean absolute error do not matter.

Table 6.2: Results of the experiment. For each dataset (row) and each classifier (column) two values are given: average mean absolute error (above) and standard deviation of error (below, starting with ±).

Dataset	OLM	OSDL	IsoSep	DOMLEM	LPRules	MORE	IsoReg	J48	SVM	NB
DenBosch	0.282	0.157	0.183	0.125	0.168	0.133	0.165	0.172	0.202	0.126
	±0.039	±0.039	±0.037	±0.034	±0.034	±0.03	±0.038	±0.032	±0.036	±0.031
Wisconsin	0.17	0.039	0.03	0.038	0.041	0.031	0.04	0.046	0.03	0.037
	±0.014	±0.008	±0.007	±0.008	±0.009	±0.007	±0.008	±0.009	±0.007	±0.007
ESL	0.371	0.353	0.328	0.432	0.323	0.344	0.328	0.369	0.355	0.333
	±0.024	±0.025	±0.023	±0.024	±0.024	±0.023	±0.023	±0.022	±0.023	±0.024
SWD	0.452	0.44	0.442	0.449	0.435	0.441	0.44	0.442	0.435	0.457
	±0.017	±0.017	±0.018	±0.017	±0.017	±0.016	±0.017	±0.016	±0.016	±0.016
LEV	0.427	0.405	0.398	0.513	0.396	0.413	0.393	0.415	0.444	0.441
	±0.018	±0.016	±0.017	±0.015	±0.016	±0.016	±0.017	±0.018	±0.016	±0.017
ERA	1.256	1.271	1.271	1.393	1.263	1.269	1.261	1.217	1.271	1.227
	±0.031	±0.033	±0.034	±0.04	±0.033	±0.03	±0.033	±0.032	±0.029	±0.031
Housing	0.527	0.775	0.286	0.337	0.274	0.288	1.187	0.332	0.314	0.506
	±0.032	±0.021	±0.02	±0.025	±0.021	±0.023	±0.032	±0.023	±0.025	±0.033
CPU	0.29	0.215	0.099	0.086	0.073	0.065	0.232	0.1	0.371	0.18
	±0.035	±0.028	±0.02	±0.019	±0.018	±0.017	±0.033	±0.019	±0.03	±0.033
Balance	0.224	0.111	0.19	0.221	0.063	0.126	0.207	0.271	0.137	0.085
	±0.02	±0.014	±0.017	±0.015	±0.009	±0.015	±0.016	±0.021	±0.017	±0.013
Ljubljana	0.325	0.274	0.241	0.289	0.25	0.251	0.24	0.259	0.299	0.252
	±0.033	±0.027	±0.024	±0.029	±0.026	±0.022	±0.021	±0.021	±0.023	±0.025
Windsor	0.576	0.512	0.52	0.519	0.516	0.538	0.534	0.565	0.491	0.509
	±0.028	±0.025	±0.028	±0.028	±0.026	±0.025	±0.026	±0.025	±0.026	±0.029
Car	0.084	0.031	0.045	0.023	0.03	0.061	0.038	0.09	0.078	0.177
	±0.01	±0.005	±0.006	±0.005	±0.004	±0.006	±0.005	±0.008	±0.007	±0.008

Friedman statistics gives 26.73 which exceeds the critical value 15.51 (for confidence level 05), so we can reject the null hypothesis and state that classifiers are not equally good. Next, we proceed to a post-hoc analysis and calculate the *critical difference* (CD) according to the Nemeneyi procedure:

$$CD = q_\alpha \sqrt{\frac{c(c+1)}{6d}},$$

where critical value q_α is based on Studentized range statistics divided by $\sqrt{2}$ (see e.g. Demšar (2006) for the table of critical values). We obtain $CD = 3.91$ which means that algorithms with difference in average ranks more than 3.91 are significantly different.

In Figure 6.2, average ranks were marked on a line, and groups of classifiers that are not significantly different were connected. This shows that algorithms such as `LPRules` and `MORE` are significantly different only to `OLM`. No other significant differences were obtained. This is mostly due to the fact that we used only 12 datasets and applied the weak and conservative nonparametric test. The difference in ranks must be very high in order to state that one algorithm outperforms another. Using some parametric test (such as ANOVA) would probably lead to stronger results, but we prefer to be as prudent as possible in drawing conclusions about the statistical significance.

Table 6.3: Ranks of each classifier for each dataset. Rank 1 corresponds to the best average error for a given dataset, while rank 10 corresponds to the worst. In the last row, the average ranks across the datasets are given for each classifiers.

Dataset	OLM	OSDL	IsoSep	VCDomLEM	LPRules	MORE	IsoReg	J48	SVM	NB
DenBosch	10	4	8	1	6	3	5	7	9	2
Wisconsin	10	6	1	5	8	3	7	9	2	4
ESL	9	6	3	10	1	5	2	8	7	4
SWD	9	3	7	8	1	5	4	6	2	10
LEV	7	4	3	10	2	5	1	6	9	8
ERA	3	8	7	10	5	6	4	1	9	2
Housing	8	9	2	6	1	3	10	5	4	7
CPU	9	7	4	3	2	1	8	5	10	6
Balance	9	3	6	8	1	4	7	10	5	2
Ljubljana	10	7	2	8	3	4	1	6	9	5
Windsor	10	3	6	5	4	8	7	9	1	2
Car	8	3	5	1	2	6	4	9	7	10
avg. rank	8.5	5.25	4.5	6.25	3.0	4.42	5.0	6.75	6.17	5.17

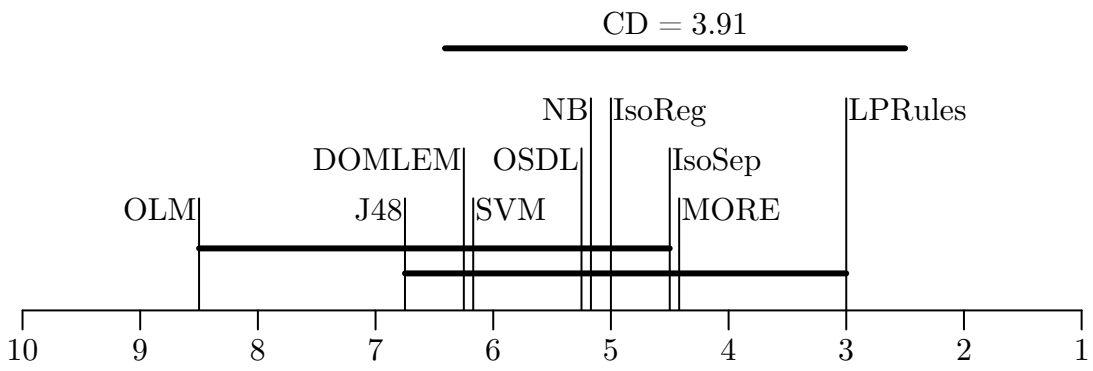

Figure 6.1: Critical difference diagram

Conclusions. Although statistical significance could not be confirmed, some important remarks can still be given:

- On the "top" of the algorithms' list there are two methods introduced in this thesis (`LPRules` and `MORE`), followed by `IsoSep`. Isotonic Separation is known to behave very well in practice (Chandrasekaran et al., 2005; Jacob et al., 2007; Ryu and Yue, 2005) and our methods perform not worse. This means that `LPRules` and `MORE` can probably be used successfully in practice and make predictions with high accuracy.

- `OSDL` and `IsoReg` perform very similar to each other and worse than the top-of-the-list methods. Notice that both `OSDL` and `IsoReg` are purely based on the dominance relation $\succeq$, i.e. they classify by comparing the test object with training objects using $\succeq$. The classification procedure is practically the same in both cases, the main difference is how those algorithms deal with inconsistencies: `IsoReg` uses statistically justified multiple isotonic regression, while `OSDL` uses simpler, but also faster procedure. Since the amount of inconsistencies is usually not very high, their total accuracy is thus very similar.

  The performance of `IsoReg` and `OSDL` is poor for some datasets, which shows that using only the dominance relation is not enough – one must use some "parametric" approach, e.g. by

restricting to the class of decision rules.

- `OLM` perform very poorly in almost every case. Note, however, that `OLM` was the first algorithm, proposed already in 1992, to deal with monotone problems.
- `VCDomLEM` has a very high variance in classification error: on some datasets the error is low, while on the other datasets the algorithm works best among all the algorithms. We do not know how to explain this behavior, but we noticed that `VCDomLEM` usually works worse on the data with high level of inconsistency.
- The ordinary classifiers which do not take monotonicity constraints into account, perform worse than our methods. This was anticipated in the previous chapters, as there is a theoretical justification for using monotone classifiers for such problems. However, the more important fact is that the ordinary classifiers can be inconsistent with the domain knowledge about the monotonicity. This leads to the problems with interpretation of the model; moreover, decision maker may want the model to be consistent with the knowledge she or he possesses about the problem. Those issues are usually much more substantial than a small gain in accuracy.

  Although neglecting the monotonicity does not deteriorate much the accuracy of the classifier, neglecting the order between the class labels do. When we first used `J48` and `SVM` directly, without ordinal setting, the results were very bad. The only exception was Naïve Bayes, which works well without taking the order into account, but its good performance follows from the fact that it is a generative method (in contrary to `J48` and `SVM`, which are discriminative), i.e. it estimates the joint distribution $P(\mathbf{x}, y)$.

6.3 Interpretability

Let us consider the intelligibility of the considered methods, i.e. how easy or difficult is the interpretation of resulting models. Since it is not possible to strictly measure and compare the intepretability, we will briefly comment on each algorithm, indicating its pros and cons with respect to this aspect. Moreover, we will estimate the size of the model (description length), defined as the amount of memory needed to save the model. This rather corresponds to the compactness of the model, but there is no other measurable property which is closer to interpretability.

Instance-based methods. Our main competitor, `IsoSep`, is an instance-based method of classification and "nearest neighbor" (with respect to the asymmetric "distance") is found each time an object is to be classified. Although there exists a reduction method (Chandrasekaran et al., 2005) which allows us to store only the Pareto and the anti-Pareto frontier of each class (the set of objects not dominated by other objects from the class and the set of objects which do not dominate other objects from the class), those sets can be very large, especially when the dimensionality m is high. In general, they grow linearly with n. This makes Isotonic Separation quite slow in the classification phase. Moreover, when classifying new objects, the nearest neighbor chosen with respect to the specific, asymmetric distance function does not explain clearly why a given prediction was made.

`OLM` stores the model by saving a subset of the training set D called "rule base". `OSDL` and Isotonic Regression (`IsoReg`) keep all of the objects, but a similar reduction as for `IsoSep` can be used to store only a subset of D. Additionally, `IsoReg` stores probabilities instead of class labels. All those models have a very similar size and interpretability properties to `IsoSep`. The main difference is that, contrary to `IsoSep`, they do not use nearest-neighbor procedure to classify new objects, they use only the dominance relation to this end, which is a much clearer and comprehensible way of classification.

Table 6.4: Average numbers of rules (per class union) for LPRules, MORE and VCDomLEM.

Data set	# objects	# rules		
		LPRules	MORE	VCDomLEM
ESL	448	3	25	5.0
SWD	1000	6.7	24.9	16.8
LEV	1000	3.2	25	10.7
ERA	1000	2.6	25	10.2
Housing	506	26.6	25	32.2
Windsor	546	19.1	25	22.3
DenBosch	119	4.7	25	6.1
Wisconsin	699	9.7	25	9.3
Ljubljana	286	8.6	25	11.2
Car	1728	5.2	24.8	10.2
CPU	209	4.1	25	5.0
Balance	625	32.2	25	44.2

Rule-based methods. MORE and LPRules build a model which consists of a set of rules with assigned weights (respones). VCDomLEM is similar but it does not assign weights to the rules. It is well known that decision rules are one of the most interpretable forms of knowledge representation. Rule weight expresses the importance and strength of the rule vote, hence its meaning is very clear. Thus, the intepretability of the whole model mostly depend on the number of rules in the ensemble. Table 6.4 contains average numbers of rules per class union for the datasets used in the experiment.

MORE produces M rules in each binary subproblem, which is $M/2$ rules per class union (there are upward and downward class unions in each subproblem). In our case, it was 25 rules per class union which is usually much smaller than the number of objects. It may sometimes (but rarely) happen that less rules are produced when no rule decreasing the current loss function can be found.

LPRules generates rules until the optimum (the largest margin) is reached. Sometimes it can take even up to 100 iteration, but at the optimum most of the generated rules have zero weights, because the solution of the linear program is usually very sparse. This means that the ensembles produced by LPRules are usually very small, much smaller then those produced by MORE, especially for the problems in which attributes have purely ordinal scale and finite domains. We believe the sparsity of the ensemble is the most important advantage of LPRules.

VCDomLEM generates rule sets larger than LPRules, but significantly smaller than MORE.

Summary. The algorithms developed by us proved to be efficient and competitive with the best existing approaches to ordinal classification with monotonicity constraints (they obtained the best results actually, but the difference was not found to be statistically significant due to a small number of datasets). What distinguishes our approaches from the main competitor, isotonic separation, is the simplicity and interpretability of the decision rule model. Moreover, isotonic separation is a "lazy learning" algorithms, therefore it needs to keep a large part of the dataset to classify unseen objects. This, in turn, slows down the classification procedure. In case of monotone rule ensembles, all we need is a small set of rules combined by a weighted sum in an ensemble, which captures the information contained in the dataset.

Summary of the Contribution

In this thesis, we dealt with the problem of ordinal classification with monotonicity constraints. As stated in Section 1.4, our goal was to provide a comprehensive and consistent statistical theory for this problem, along with an efficient and accurate method for solving it. In our opinion, the goal has been achieved. To support this claim, we provide below a summary of our contributions to the field of ordinal classification with monotonicity constraints.

List of Main Results

Our results can be considered from both theoretical and practical point of view. The theoretical results of the thesis can be found mainly in Chapters 2, 3 and 4, and they are related to the development of a statistical learning methodology for ordinal classification with monotonicity constraints. The practical results were presented in Chapters 5 and 6, and they are related to introducing novel monotone classification methods, which are verified in course of a computational experiment.

Below, we list our contributions in the chronological order:

Statistical framework for ordinal classification with monotonicity constraints. We proposed a definition of the problem from probabilistic point of view. This was done by introducing monotonically constraint (with respect to the stochastic dominance) probability distribution. Then, we also showed that some more constraints must be imposed on the loss function in order to assure that the Bayes classifier is monotone.

Multiple isotonic regression. We considered the problem of probability estimation. We proposed a nonparametric method, taking into account only the monotonicity constraints expressed by the dominance relation. Our method is based on isotonic regression and is equivalent to MLE in binary-class case. Although isotonic regression has already been used to this end in binary-class case with linear preorder relation, our approach for any number of classes and partial preorder is new.

Monotone approximation. We proposed a general method for removing the inconsistencies and "monotonizing" the data in a statistically safe way. The method is called *monotone approximation* and is based on relabeling the training objects. This is done in a nonparametric way, by empirical risk minimization within the class of all monotone functions. Although the problem of relabeling has already been used, we are first who derived this problem from the statistical perspective and who showed its connection with isotonic regression. We also first raised and solved the problem of non-unique optimal solution and we gave a detailed analysis of possible problem reduction to speed up the computations. Finally, we proved the convergence of our method to the Bayes classifier for a wide class of probability distributions.

Stochastic Dominance-based Rough Sets. We extended DRSA into the stochastic setting by defining the lower approximations (or, equivalently, generalized decision) in the probabilistic way. Notice that this was done in a different manner than in the previous variable consistency approaches, which were also based on probabilistic reasoning. After defining the model, we proposed to estimate the probabilities (which are unknown) by multiple isotonic regression. Our extension has classical DRSA in the deterministic limit.

Then, we considered the problem of monotone confidence interval estimation, which can be though of as a generalization of monotone approximation into the abstaining classifiers. We introduced to this end a specific, interval loss function and showed that stochastic DRSA is equivalent to minimizing this loss within the class of all monotone functions. Since the problem is linear, we obtain a fast algorithm for calculating the lower approximations without estimation of probabilities (i.e. isotonic regression does not need to be solved).

Our theory extends well beyond DRSA and can be though of as a statistical base for explanation of all rough set approaches.

Monotone Rules Ensembles. We introduced new algorithms based on decision rules for ordinal classification with monotonicity constraints. Our algorithms are based on the so called two-phase procedure: in the first phase, the monotone approximation (or, equivalently, Stochastic DRSA) is applied to the training data in order to get rid of the inconsistencies. In the second phase, the rule ensemble is built on the monotonized data. We explained why monotonization is made and how it is connected with separability of the data by a set of rules. We proved that the accuracy of the ensemble learned on the monotonized data without errors, and having a large margin, remains close to the accuracy of the optimal classifier. We proposed a method of reducing the main K-class problem into $K-1$ binary subproblems and showed the necessary conditions for monotonicity of the classifier. This constitutes to a general theory of monotone rule ensembles.

Next, we proposed two particular rule induction algorithms for dealing with a linear loss function. The first algorithm, LPRules (Linear Programming Rule Ensemble), is based on the linear programming boosting which is known to directly maximize the minimal margin on the dataset. LPRules produces very small and accurate ensembles. The second algorithm, MORE (Sigmoid Loss Monotone Rule Ensemble) is based on greedily minimizing a differentiable approximation of the linear loss (sigmoid loss), using the boosting strategy to learning. MORE also achieves high accuracy on the real datasets.

Our rule induction methods produce compact and interpretable sets of rules, consistent with the domain knowledge about the order and monotonicity, maintaining very good prediction performance.

Future Research

There still remain many unsolved questions in the investigated research area. We mention below some problems which we would like to study in the future:

Mining monotonicity constraints. It is not always clear whether the monotone relationship holds for our data, although the expert may say so. Moreover, even if no domain knowledge is available, it can be worth considering automatic way of mining the monotonicity constraints from data. This improves our knowledge about the problem and thus mining the monotonicity constraints can be regarded as a part of the knowledge discovery process.

Reduction of dimensionality. The nonparametric methods (isotonic regression, monotone approximation) are based only on dominance relation. As the dimensionality of the problem (number of attributes m) increases, the dominance relation becomes sparse and the nonparametric methods become less reliable, since less objects are comparable by a dominance relation. This is one of the manifestations of the famous curse of dimensionality (Bellman, 1961). Generally, if m is too high, one should decrease the dimension of the space by removing some of the attributes. This is in fact a feature selection process but now is driven not only by choosing the most informative attributes but also the most monotone ones. There is a need for the research in this area.

More extensive experimental evaluation. We plan to extend the computational experiment into more datasets. We are currently at the stage of gathering the data.

Ranking with monotonicity constraints. The final and most important issue planned for our future research is the extension of the theory proposed in this thesis into the ranking problems, where the monotonicity constraints are present in the data. In particular, we would like to develop a statistical framework based on the rank loss formulation of the ranking problem.

Bibliography

J. A. Anderson. Regression and ordered categorical variables. *Journal of the Royal Statistical Society*, 46:1–30, 1984.

A. Asuncion and D. Newman. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/~mlearn/{MLR}epository.html`.

M. Ayer, H. D. Brunk, G. M. Ewing, W. T. Reid, and E. Silverman. An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 26(4): 641–647, 1955.

M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, 2006.

R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

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

A. Ben-David. Monotonicity maintenance in information-theoretic machine learning algorithms. *Machine Learning*, 19(1):29–43, 1995.

A. BenDavid, L. Sterling, and Y.-H. Pao. Learning and classification of monotonic ordinal concepts. *Computational Intelligence*, 5(1):45–49, 1989.

Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research*, 5:1089–1105, 2004.

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

M. Berkelaar. LP_SOLVE: Linear programming code, 2005. URL `http://sourceforge.net/projects/lpsolve/`.

J. C. Bioch and V. Popova. Monotone decision trees and noisy data. Technical report, Erasmus University Rotterdam, 2002.

J. Błaszczyński and R. Słowiński. Incremental induction of decision rules from dominance-based rough approximations. *Electronic Notes in Theoretical Computer Science*, 82(4), 2003.

J. Błaszczyński, K. Dembczyński, W. Kotłowski, R. Słowiński, and M. Szeląg. Ensembles of decision rules for solving binary classification problems in the presence of missing values. In *Rough Sets and Current Trends in Computing*, volume 4259 of *Lecture Notes in Computer Science*, pages 224–234, Kobe, Japan, 2006a. Springer-Verlag.

J. Błaszczyński, K. Dembczyński, W. Kotłowski, R. Słowiński, and M. Szeląg. Ensemble of decision rules. *Foundations of Computing and Decision Sciences*, 31(1), 2006b.

J. Błaszczyński, S. Greco, and R. Słowiński. Multi-criteria classification — a new scheme for application of dominance-based decision rules. *European Journal of Operational Research*, 181: 1030–1044, 2007.

J. Błaszczyński, S. Greco, R. Słowiński, and M. Szeląg. Monotonic variable consistency rough set approaches. In *Rough Sets and Knowledge Technology*, volume 4481 of *Lecture Notes in Computer Science*, pages 126–133, Toronto, Canada, 2007.

J. Błaszczyński, S. Greco, R. Słowiński, and M. Szeląg. Monotonic variable consistency rough set approaches. *Journal of Approximate Reasoning*, to appear, 2008.

E. Boros, P. L. Hammer, and J. N. Hooker. Boolean regression. *Annals of Operations Research*, 58 (3), 1995.

E. Boros, P. L. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik. An implementation of logical analysis of data. *IEEE Transactions on Knowledge and Data Engineering*, 12:292–306, 2000.

B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *COLT*, pages 144–152, 1992.

L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26:801–849, 1998.

L. Breiman. Prediction games and arcing algorithms. *Neural Computation*, 11(7):1493–1517, 1999.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

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

O. Burdakov, O. Sysoev, A. Grimvall, and M. Hussian. An $O(n^2)$ algorithm for isotonic regression. In *Large-Scale Nonlinear Optimization*, volume 83 of *Nonconvex Optimization and Its Applications*, pages 25–33. Springer-Verlag, 2006.

K. Cao-Van. *Supervised Ranking, from semantics to algorithms*. Ph.D. dissertation, Ghent University, 2003.

K. Cao-Van and B. De Baets. Growing decision trees in an ordinal setting. *International Journal of Intelligent Systems*, 18:733–750, 2003.

K. Cao-Van and B. De Baets. An instance-based algorithm for learning rankings. In *Proceedings of Benelearn*, pages 15–21, 2004.

R. Chandrasekaran, Y. U. Ryu, V. S. Jacob, and S. Hong. Isotonic separation. *INFORMS Journal on Computing*, 17(4):462–474, 2005.

G. Choquet. Theory of capacities. *Annales de L'Institut Fourier*, 5:131—-295, 1953.

C. Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16:41–46, 1970.

N. Christopeit and G. Tosstorff. Strong consistency of least-squares estimators in the monotone regression model with stochastic regressors. *Annals of Statistics*, 15(2):568–586, 1987.

P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.

S. Clémençon, G. Lugosi, and N. Vayatis. Ranking and empirical minimization of U-statistics. `arXiv:math/0603123v1`, 2006.

W. W. Cohen. Fast effective rule induction. In *International Conference of Machine Learning*, pages 115–123, 1995.

W. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *National Conference on Artificial Intelligence*, pages 335–342, 1999.

M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. In *COLT '00: Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, pages 158–169, 2000.

H. Daniels and A. Feelders. Combining domain knowledge and data in datamining systems. Discussion Paper 63, Tilburg University, Center for Economic Research, 2000. URL `http://ideas.repec.org/p/dgr/kubcen/200063.html`.

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

K. Dembczyński, R. Pindur, and R. Susmaga. Dominance-based rough set classifier without induction of decision rules. *Electronic Notes in Theoretical Computer Science*, 82(4), 2003.

K. Dembczyński, S. Greco, W. Kotłowski, and R. Słowiński. Quality of rough approximation in multi-criteria classification problems. In *Rough Sets and Current Trends in Computing*, volume 4259 of *Lecture Notes in Computer Science*, pages 318–327, Kobe, Japan, 2006a. Springer-Verlag.

K. Dembczyński, W. Kotłowski, and R. Słowiński. Additive preference model with piecewise linear components resulting from dominance-based rough set approximations. In *Artificial Intelligence and Soft Computing - ICAISC 2006*, volume 4029 of *Lecture Notes in Computer Science*, pages 499–508, Zakopane, Poland, 2006b. Springer-Verlag.

K. Dembczyński, S. Greco, W. Kotłowski, and R. Słowiński. Optimized generalized decision in dominance-based rough set approach. In *Rough Sets and Knowledge Technology*, volume 4481 of *Lecture Notes in Computer Science*, pages 118–125, Toronto, Canada, 2007a.

K. Dembczyński, S. Greco, W. Kotłowski, and R. Słowiński. Statistical model for rough set approach to multicriteria classification. In *Knowledge Discovery in Databases: PKDD 2007*, volume 4702 of *Lecture Notes in Computer Science*, pages 164–175, Warsaw, Poland, 2007b. Springer-Verlag.

K. Dembczyński, W. Kotłowski, and R. Słowiński. Ordinal classification with decision rules. In *Mining Complex Data 2007*, volume 4944 of *Lecture Notes in Artificial Intelligence*, pages 169–181, Warsaw, Poland, 2008a. Springer-Verlag.

K. Dembczyński, W. Kotłowski, and R. Słowiński. Ensemble of decision rules for ordinal classification with monotonicity constraints. In *Rough Sets and Knowledge Technology 2008*, volume 5009 of *Lecture Notes in Artificial Intelligence*, pages 260–267, Chengdu, China, 2008b. Springer-Verlag.

K. Dembczyński, W. Kotłowski, and R. Słowiński. A general framework for learning an ensemble of decision rules. In *LeGo '08: From Local Patterns to Global Models, ECML/PKDD 2008 Workshop*, 2008c.

K. Dembczyński, W. Kotłowski, and R. Słowiński. Maximum likelihood rule ensembles. pages 224–231, 2008d.

K. Dembczyński, W. Kotłowski, and R. Słowiński. Solving regression by learning an ensemble of decision rules. In *International Conference on Artificial Intelligence and Soft Computing, 2008*, volume 5097 of *Lecture Notes in Artificial Intelligence*, pages 533–544, Zakopane, Poland, 2008e. Springer-Verlag.

A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning Journal*, 46(1–3):225–254, 2001.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1st edition, 1996.

M. Doumpos and F. Pasiouras. Developing and testing models for replicating credit ratings: A multicriteria approach. *Computational Economics*, 25(4):327–341, 2005.

H. Drucker and C. Cortes. Boosting decision trees. In *Advances in Neural Information Processing Systems 8, NIPS*, pages 479–485, 1996.

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

I. Düntsch and G. Gediga. *Rough set data analysis: A road to non-invasive knowledge discovery*. Methodos Publishers (UK), Bangor, 2000. URL `http://www.cosc.brocku.ca/~duentsch/archive/nida.pdf`.

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

A. Feelders and M. Pardoel. Pruning for monotone classification trees. In *Advances in Intelligent Data Analysis V*, volume 2810 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

E. Frank and M. Hall. A simple approach to ordinal classification. In *Proc. 12th European Conference on Machine Learning*, volume 2167 of *Lecture Notes in Computer Science*, pages 145–157, Freiburg, Germany, 2001. Springer.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

J. H. Friedman and B. E. Popescu. Gradient directed regularization. Technical report, Dept. of Statistics, Stanford University, 2004.

J. H. Friedman and B. E. Popescu. Predictive learning via rule ensembles. Technical report, Dept. of Statistics, Stanford University, 2005.

J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Stanford university technical report, Dept. of Statistics, 1998.

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

J. Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, 1996.

S. Giove, S. Greco, B. Matarazzo, and R. Słowiński. Variable consistency monotonic decision trees. In *Rough Sets and Current Trends in Computing*, volume 2475 of *Lecture Notes in Artificial Intelligence*, pages 247–254, Berlin, 2002. Springer-Verlag.

M. Grabisch. The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Research*, 89:445–456, 1996.

S. Greco, B. Matarazzo, and R. Słowiński. A new rough set approach to evaluation of bankruptcy risk. In C. Zopounidis, editor, *Operational Tools in the Management of Financial Risks*, pages 121–136. Kluwer Academic Publishers, Dordrecht, 1998.

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

S. Greco, B. Matarazzo, and R. Słowiński. The use of rough sets and fuzzy sets in multiple-criteria decision making. In T.Gal, T.Stewart, and T.Hanne, editors, *Advances in Multiple Criteria Decision Making*, chapter 14, pages 14.1–14.59. Kluwer Academic Publishers, 1999b.

S. Greco, B. Matarazzo, and R. Słowiński. Extension of the rough set approach to multicriteria decision support. *INFOR*, 38(3):161–196, 2000.

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

S. Greco, B. Matarazzo, R. Słowiński, and J. Stefanowski. Variable consistency model of dominance-based rough set approach. In Z. W. and Y. Yao, editors, *Rough Sets and Current Trends in Computing*, volume 2005 of *Lecture Notes in Artificial Intelligence*, pages 170–181, Berlin, 2001b. Springer-Verlag.

S. Greco, B. Matarazzo, R. Słowiński, and J. Stefanowski. An algorithm for induction of decision rules consistent with the dominance principle. In *Rough Sets and Current Trends in Computing*, volume 2005 of *Lecture Notes in Artificial Intelligence*, pages 304–313, Berlin, 2001c. Springer-Verlag.

S. Greco, B. Matarazzo, and R. Słowiński. Multicriteria classification. In W. Kloesgen and J. Żytkow, editors, *Handbook of Data Mining and Knowledge Discovery*, chapter 16.1.9, pages 318–328. Oxford University Press, New York, 2002a.

S. Greco, B. Matarazzo, and R. Słowiński. Rough approximation by dominance relations. *International Journal of Intelligent Systems*, 17(2):153–171, 2002b.

S. Greco, B. Matarazzo, and R. Słowiński. Axiomatic characterization of a general utility function and its particular cases in terms of conjoint measurement and rough-set decision rules. *European Journal of Operational Research*, 158:271–292, 2004a.

S. Greco, B. Matarazzo, and R. Słowiński. Dominance-based rough set approach to knowledge discovery (I) - general perspective. In N. Zhong and J. Liu, editors, *Intelligent Technologies for Information Analysis*, chapter 20, pages 513–552. Springer-Verlag, Berlin, 2004b.

S. Greco, B. Matarazzo, and R. Słowiński. Dominance-based rough set approach to knowledge discovery (II) - extensions and applications. In N. Zhong and J. Liu, editors, *Intelligent Technologies for Information Analysis*, chapter 20, pages 533–612. Springer-Verlag, Berlin, 2004c.

S. Greco, B. Matarazzo, and R. Słowiński. Decision rule approach. In J.Figueira, S.Greco, and M.Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, chapter 13, pages 507–562. Springer-Verlag, New York, 2005.

S. Greco, B. Matarazzo, and R. Słowiński. Rough set approach to customer satisfaction analysis. volume 4259 of *Lecture Notes in Computer Science*, pages 284–295. Springer, 2006.

S. Greco, B. Matarazzo, and R. Słowiński. Dominance-based rough set approach as a proper way of handling graduality in rough set theory. In *Transactions on Rough Sets VII*, volume 4400 of *Lecture Notes in Computer Science*, pages 36–52, Berlin, 2007a. Springer-Verlag.

S. Greco, R. Słowiński, and Y. Yao. Bayesian decision theory for dominance-based rough set approach. In *Rough Sets and Knowledge Technology*, volume 4481 of *Lecture Notes in Computer Science*, pages 134–141, Toronto, Canada, 2007b.

S. Greco, V. Mousseau, and R. Słowiński. Ordinal regression revisited: multiple criteria ranking with a set of additive value functions. *European Journal of Operational Research*, 191:415–435, 2008.

J. W. Grzymala-Busse. LERS — a system for learning from examples based on rough sets. In R. Słowiński, editor, *Intelligent Decision Support, Handbook of Applications and Advances of the Rough Sets Theory*, pages 3–18. Kluwer Academic Publishers, 1992.

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

V. Jacob, R. Krishnan, and Y. U. Ryu. Internet content filtering using isotonic separation on the PICS specification. *ACM Transactions on Internet Technology*, 7(1), 2007.

E. Jacquet-Lagrèze. An application of the UTA discriminant model for the evaluation of R&D projects. In P. M. Pardalos, Y. Siskos, and C. Zopounidis, editors, *Advances in Multicriteria Analysis*, pages 203–211. Kluwer Academic Publishers, Dordrecht, 1995.

E. Jacquet-Lagrèze and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision making: The UTA method. *European Journal of Operational Research*, 10:151–164, 1982.

G. Karakoulas and J. Shawe-Taylor. Optimizing classifiers for imbalanced training sets. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 253–259, 1999. ISBN 0-262-11245-0.

M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, 1994.

J. Kivinen and M. Warmuth. Boosting as entropy projection. In *COLT '99: Proceedings of the twelfth annual conference on Computational learning theory*, pages 134–144, 1999.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 2002.

V. Koltchinskii and D. Panchenko. Complexities of convex combinations and bounding the generalization error in classification. *Annals of Statistics*, 33(4):1455–1496, 2006.

G. Koop. *Analysis of Economic Data*. John Wiley and Sons, 2000.

J. Koronacki and J. Ćwik. *Statistical Learning Systems*. WNT, Warsaw, Poland, 2005.

W. Kotłowski and R. Słowiński. Statistical approach to ordinal classification with monotonicity constraints. In *Preference Learning ECML/PKDD 2008 Workshop*, 2008.

W. Kotłowski, K. Dembczyński, S. Greco, and R. Słowiński. Stochastic dominance-based rough set model for ordinal classification. *Information Sciences*, 178(21):3989–4204, 2008.

J. Leskovec and J. Shawe-Taylor. Linear programming boosting for uneven datasets. In *20th International Conference on Machine Learning (ICML'03)*, 2003.

H. Levy. *Stochastic Dominance*. Kluwer Academic Publishers, 1998.

H.-T. Lin and L. Li. Ordinal regression by extended binary classifications. *Advances in Neural Information Processing Systems*, 19:865–872, 2007.

G. Lugosi. Pattern classification and learning theory. In *Principles of Nonparametric Learning*, pages 1–56. Springer, 2002.

K. Makino, T. Suda, H. Ono, and T. Ibaraki. Data analysis by positive decision trees. *IEICE Transations on Information and Systems*, E82-D(1):76—-88, 1999.

J.-L. Marichal, P. Meyer, and M. Roubens. Sorting multi-attribute alternatives: the TOMASO method. *Computers and Operations Research*, 32(4):861–877, 2005.

L. Mason, J. Baxter, P. Bartlett, and M. Frean. *Functional gradient techniques for combining hypotheses*, pages 33–58. MIT Press, 1999.

L. Mason, P. L. Bartlett, and J. Baxter. Improved generalization through explicit optimization of margins. *Machine Learning*, 38(3):243–255, 2000.

W. L. Maxwell and J. A. Muchstadt. Establishing consistent and realistic reorder intervals in production-distribution systems. *Operations Research*, 33:1316–1341, 1985.

C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

R. S. Michalski. A theory and methodology of inductive learning. In *Machine Learning: An Artificial Intelligence Approach*, pages 83–129. Tioga Publishing, Palo Alto, 1983.

C. Nadeau and Y. Bengio. Inference for the generalization error. *Machine Learning*, 52(3):239–281, 2003.

C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization*. Dover Publications, New York, 1998.

Z. Pawlak. Rough sets and intelligent data analysis. *Information Sciences*, 147(1–4):1–12, 2002.

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

Z. Pawlak. *Rough Sets. Theoretical Aspects of Reasoning about Data.* Kluwer Academic Publishers, Dordrecht, 1991.

Z. Pawlak and A. Skowron. Rough sets. Some extensions. *Information Sciences*, 177(1):341–356, 2007.

Z. Pawlak and R. Słowiński. Rough set approach to multi-attribute decision analysis. *European Journal of Operational Research*, 72:443–459, 1994.

Z. Pawlak, J. Grzymała-Busse, R. Słowiński, and W. Ziarko. Rough sets. *Communications of the ACM*, 38(11):89–95, 1995.

T. Pietraszek. Optimizing abstaining classifiers using ROC analysis. In *Proceedings of 22th International Conference on Machine Leaning (ICML'05)*, pages 665–672, 2005.

R. Pindur and R. Susmaga. Fast rule extraction with binary-coded relations. *Intelligent Data Analysis*, 7(1):27–42, 2003.

R. Pindur, R. Susmaga, and J. Stefanowski. Hyperplane aggregation of dominance decision rules. *Fundam. Inform.*, 61(2):117–137, 2004.

V. N. Popova. *Knowledge Discovery and Monotonicity.* Ph.D. dissertation, Erasmus University Rotterdam, 2004.

R. Potharst. *Classification using decision trees and neural networks.* Ph.D. dissertation, Erasmus University Rotterdam, 1999.

R. Potharst and J. Bioch. Decision trees for ordinal classification. *Intelligent Data Analysis*, 4(2): 97—-112, 2000.

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

R. Potharst, J. C. Bioch, and T. C. Petter. Monotone decision trees. Technical report, Erasmus University Rotterdam, 1997.

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

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

G. Rätsch and M. Warmuth. Efficient margin maximization with boosting. *Journal of Machine Learning Research*, 6:2131–2152, 2005.

G. Rätsch, B. Schölkopf, A. J. Smola, K. Müller, T. Onoda, and S. Mika. ν-arc: Ensemble learning in the presence of outliers. In *Advances in Neural Information Processing Systems 12*, 2000.

T. Robertson and F. T. Wright. Consistency in generalized isotonic regression. *Annals of Statistics*, 3(2):350–362, 1975.

T. Robertson, F. T. Wright, and R. L. Dykstra. *Order Restricted Statistical Inference.* John Wiley & Sons, 1998.

B. Roy. *Multicriteria Methodology for Decision Aiding.* Kluwer Academic Publishers, Dordrecht, 1996.

C. Rudin, R. E. Schapire, and I. Daubechies. Precise statements of convergence for adaboost and arc-gv. In *Proc. AMS-IMS-SIAM Joint Summer Research Conference*, pages 131–145, 2007a.

C. Rudin, R. E. Schapire, and I. Daubechies. Analysis of boosting algorithms using the smooth margin function. *Annals of Statistics*, 35(6):2723–2768, 2007b.

Y. U. Ryu and W. T. Yue. Firm bankruptcy prediction: Experimental comparison of isotonic separation and other classification approaches. *IEEE Transactions on Systems, Man and Cybernetics*, 35(5):727–737, 2005.

Y. U. Ryu, R. Chandrasekaran, and V. Jacob. Data classification using the isotonic seeparation technique: Application to breast cancer prediction. *Eurpoean Journal of Operational Research*, 181(2):842–854, 2007.

R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

J. Sill. Monotonic networks. In *Advances in Neural Information Processing Systems*, volume 10, pages 661–667, Denver, USA, 1998. The MIT Press.

J. Sill and Abu-Mostafa. Monotonicity hints. In *Advances in Neural Information Processing Systems*, volume 9, pages 634–640, Denver, USA, 1997. The MIT Press.

Y. Siskos, E. Grigoroudis, and N. F. Matsatsinis. UTA methods. In J. Figueira, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 297–344. Springer Verlag, Boston, Dordrecht, London, 2005.

R. Słowiński, editor. *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory.* Kluwer Academic Publishers, Dordrecht, 1992.

R. Słowiński. Rough set learning of preferential attitude in multi-criteria decision making. In *Methodologies for Intelligent Systems*, volume 683 of *Lecture Notes in Artificial Intelligence.* Springer-Verlag, 1993.

R. Słowiński. Rough set analysis of multi-attribute decision problems. In *Rough Sets, Fuzzy Sets and Knowledge Discovery*, Workshops in Computing, pages 136–142. Springer-Verlag, 1994.

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

R. Słowiński, S. Greco, and B. Matarazzo. Axiomatization of utility, outranking and decision-rule preference models for multiple-criteria classification problems under partial inconsistency with the dominance principle. *Control and Cybernetics*, 31(4):1005–1035, 2002a.

R. Słowiński, S. Greco, and B. Matarazzo. Rough set analysis of preference-ordered data. In *Rough Sets and Current Trends in Computing*, volume 2475 of *Lecture Notes in Artificial Intelligence*, pages 44–59, Berlin, 2002b. Springer-Verlag.

R. Słowiński, S. Greco, and B. Matarazzo. Rough set based decision support. In E. K. Burke and G. Kendall, editors, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, chapter 16, pages 475–527. Springer-Verlag, New York, 2005.

J. Stefanowski. On rough set based approach to induction of decision rules. In L. Polkowski and A. Skowron, editors, *Rough Set in Knowledge Discovering*, pages 500–529. Physica Verlag, 1998.

J. Stefanowski and M. Żurawski. Incremental rule induction for multicriteria and multiattribute classification. In M. A. Klopotek, S. T. Wierzchon, and K. Trojanowski, editors, *Intelligent Information Systems*, Advances in Soft Computing, pages 311–319. Springer, 2003.

K. Sugeno. *Theory of fuzzy integrals and its applications.* Ph.D. dissertation, Tokyo Institute of Technology, 1974.

V. Vapnik. *Statistical Learning Theory.* Wiley, 1998.

V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1999.

M. Velikova. *Monotone Models for Prediction in Data Mining.* Ph.D. dissertation, Tilburg University, 2006.

A. Verkeyn, D. Botteldooren, B. De Baets, and G. De Tré. Modelling annoyance aggregation with choquet integrals. In B. De Baets, J. Fodor, and G. Pasi, editors, *Proceedings of the Eurofuse Workshop on Information Systems*, pages 259—-264, 2002.

S. Wang. Neural network techniques for monotonic nonlinear models. *Computers & OR*, 21(2): 143–154, 1994.

S. Wang and N. P. Archer. A neural network technique in modeling multiple criteria multiple person decision making. *Computers & OR*, 21(2):127–142, 1994.

M. Warmuth, J. Liao, and G. Rätsch. Totally corrective boosting algorithms that maximize the margin. pages 1001–1008, 2006.

M. Warmuth, K. Glocer, and G. Rätsch. Boosting algorithms for maximizing the soft margin. 2008a.

M. K. Warmuth, K. A. Glocer, and S. Vishwanathan. Entropy regularized LPBoost. In *Proceeding of the 19th International Conference on Algorithmic Learning Theory (ALT 08)*, 2008b.

S. M. Weiss and N. Indurkhya. Lightweight rule induction. In *International Conference on Machine Learning*, pages 1135–1142, 2000.

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

D. H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, 1996.

Y. Yao. Decision-theoretic rough set models. In *Rough Sets and Knowledge Technology*, volume 4481 of *Lecture Notes in Artificial Intelligence*, pages 1–12, Toronto, Canada, 2007. Springer-Verlag.

Y. Yao and S. Wong. A decision theoretic framework for approximating concepts. *International Journal of Man-machine Studies*, 37(6):793–809, 1992.

W. Ziarko. *Set approximation quality measures in the variable precision rough set model*, pages 442–452. IOS Press, 2001.

W. Ziarko. Probabilistic rough sets. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, volume 3641 of *Lecture Notes in Artificial Intelligence*, pages 283–293, Regina, Canada, 2005. Springer-Verlag.

C. Zopounidis and M. Doumpos. A preference disaggregation decision support system for financial classification problems. *European Journal of Operation Research*, 130(2):402–413, 1997.

H. Zou, J. Zhu, and T. Hastie. The margin vector, admissible losses and multi-class margin-based classifiers. Technical report, University of Minnesota, 2005.