Learning rule sets and Sugeno integrals for monotonic classification problems

Quentin Brabant, Miguel Couceiro, Didier Dubois, Henri Prade, Agnès Rico

Learning rule sets and Sugeno integrals for monotonic classification problems

Quentin Brabant[a,1], Miguel Couceiro[a], Didier Dubois[b], Henri Prade[b], Agnès Rico[c]

[a] *Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France*
[b] *IRIT, CNRS, Université Paul Sabatier, F-31062 Toulouse, France*
[c] *ERIC, Université Claude Bernard Lyon 1, F-69100 Villeurbanne, France*

Abstract

In some variants of the supervised classification setting, the domains of the attributes and the set of classes are totally ordered sets. The task of learning a classifier that is nondecreasing w.r.t. each attribute is called monotonic classification. Several kinds of models can be used in this task; in this paper, we focus on decision rules. We propose a method for learning a set of decision rules that optimally fits the training data while favoring short rules over long ones. We give new results on the representation of sets of if-then rules by extensions of Sugeno integrals to distinct attribute domains, where local utility functions are used to map attribute domains to a common totally ordered scale. We study whether such qualitative extensions of Sugeno integral provide compact representations of large sets of decision rules.

Keywords: Monotonic classification, monotonicity constraint, decision rules, Sugeno integral, decomposable model, MCDA

1. Introduction

The standard classification task can be formulated as follows: for a given set of classes (or labels), several attributes describing objects to be classified and given a set of already classified objects, try to learn a classifier that predicts the classes of a new observation for which only the attribute values are known. In some real world applications of the classification task, like in decision-support or recommender systems, the attribute domains and the set of classes are totally ordered (classes correspond to levels of satisfaction, for instance). The task of learning a nondecreasing classifier is called *monotonic*

*Corresponding author:

classification (e.g., the higher the attribute values, the better the object). The prior knowledge about the data is thus typically that the class of observations "increases with" their attribute values. In this case, one can require that the learned classifier conforms with this prior knowledge. A brief review of monotonic classification methods is provided in the first section of the Appendix. For a more comprehensive survey see, e.g., [13].

This task can be performed through different approaches, for instance, through sets of decision rules. In the monotonic case, rules often take special forms giving conditions under which the class of an object is at least as good as a given class (selection rules) or at most as good as a given class (rejection rules). Unsurprisingly, sets of selection rules and sets of rejection rules define models that are structurally monotonic and have the advantage of relying on ordinal information only: they can deal with qualitative attribute domains (e.g., scales such as $low < medium < high$) without having to map them into numerical values. Rule sets are often considered as interpretable models. However, rule sets learned from empirical data are sometimes very large, and some rules may involve many conditions. Hence, in some cases, rule sets may provide little "high-level" understanding of the data they are learned from.

An important remark for monotonic classifiers is that they can be viewed as aggregation functions [36], provided that any object having the worst (resp. best) possible ratings on all attributes is assigned to the worst (resp. best) class. However, they are *discrete* aggregation functions, defined on scales where only the minimum and maximum operators make sense. A noteworthy class of discrete aggregation functions that only use the minimum and maximum operators is that of Sugeno integrals. However, Sugeno integrals can be defined only if the attribute scales are all the same, and in one-to-one correspondence with the totally ordered set of classes. In fact, Greco *et al.* [37, 51] had already observed that Sugeno integrals have limited expressive power. They can only represent selection and rejection rules involving a single threshold for all attributes. This problem can be partially circumvented using qualitative decomposable models based on Sugeno integrals, that we call Sugeno Utility Functionals (SUFs), first introduced in [38] and extensively studied in [17, 19, 20, 21, 18], where local mappings from attribute scales to the set of classes are used to make the former commensurate with latter.

In this paper, which is an extended version of the conference paper [8], we address two issues relative to sets of selection and rejection rules. Firstly, we clarify the question of representing monotonic aggregation functions via SUFs, showing that more than one SUF is usually needed to represent a monotonic rule set. This work complements results obtained in [38], where it is shown that any monotonic aggregation function represents a set of selection

rules, and Sugeno integrals encode a set of single-threshold selection rules. Secondly, we propose a non-parametric algorithm that learns a rule set that optimally fits the training data, while favoring short rules over long ones, as well as an algorithm that minimizes the number of SUFs needed to represent a set of rules. The original motivation for this work was to take advantage of the theoretical result concerning SUFs for learning compact representations of qualitative monotonic classification data and also to describe such data in terms of selection rules, thus obtaining a concise, possibly approximate, user-friendly representation. However, in the paper we shall start with the problem of representing selection rules with multiple thresholds before finding the combination of SUFs they correspond to. Our method does not clearly fit in the taxonomy of methods for monotonic classification proposed in [13]. It uses a fuzzy integral which is a qualitative counterpart of Choquet integral, but at the same time, it generates rules that however do not form a decision tree.

The paper is organized as follows. In Section 2 we formalize the task of monotonic classification. In Section 2.2 we present monotonic rules and rule-sets as well as the monotonic classifiers they induce. In Section 3, we present Sugeno integrals and their extensions, called *Sugeno Utility Functionals* (SUFs). In Section 4, we show how Sugeno integrals and SUFs can be translated to rules and, conversely, how to identify rule sets that can be represented by SUFs. We also present the SUF-set models, whose expressiveness is the same as that of monotonic rule-sets. Section 5 initiates the experimental part of the paper. We introduce a new rule-set learning algorithm for monotonic classification with qualitative data. We evaluate its predictive accuracy on empirical datasets. Our method is compared to some existing methods that are recalled in the first section of the Appendix. Section 6 deals with the problem of representing a monotonic rule-set by a SUF set of minimal size. In Section 7, we first give a method for interpolating monotonic datasets with SUF-sets of minimal cardinality. We also exploit our interpolation methods in a learning algorithm that combines rule extraction and SUF set construction from rules, and we evaluate it on empirical datasets. In the conclusion, we discuss the use of sets of SUFs for compact represention of qualitative data, as well as their role in bridging the gap between numerical aggregation and rule-based models.

The following table recalls the notation employed throughout the paper.

	Symbols	Meaning
Section 2	$[n]$	set of integers $\{1, \dots, n\}$
	$X_1, \dots, X_n$	domains of the attributes
	$\mathbf{X}$	$X_1 \times \cdots \times X_n$, set of descriptions
	L	set of classes (or labels)
	0, 1	respectively: lowerbound and upperbound of a set
	$\mathbf{x}$	description, i.e., n-tuple $(x_1, \dots x_n) \in \mathbf{X}$
	$(\mathbf{x}, y)$	observation, element of $\mathbf{X} \times L$
	$\mathcal{D}$	set of available observations, subset of $\mathbf{X} \times L$
	$\eta_{\mathcal{D}}$	$\eta_{\mathcal{D}} : \mathcal{D} \to \mathbb{N}$, number of times an observation occurs
	$\mathbf{X}_{\mathcal{D}}$	set of all descriptions appearing in $\mathcal{D}$
Section 2.2	$(\alpha_1, \dots, \alpha_n) \to \delta$	selection rule $\big[x_1 \geq \alpha_1$ and $\dots$ and $x_n \geq \alpha_n\big] \Rightarrow y \geq \delta$
	A^r	active thresholds in the rule r
	f_r	function specified by r
	R	set of selection rules
	f_R	function specified by R
	$\mathrm{Eq}(R)$	equivalence class of R
	R^*	smallest element of $\mathrm{Eq}(R)$
Section 3	μ	capacity
	S_μ	Sugeno-integral w.r.t. μ
	$\varphi_1, \dots, \varphi_n$	local QNFs, $\varphi_i : X_i \to L$ for each $i \in [n]$
	φ	global QNF, $\varphi = \varphi_1 \times \cdots \times \varphi_n$
	$S_{\mu,\varphi}$	SUF defined by μ and ϕ, SUF defined by ν and ϕ
	$\mathcal{R}_{\mathbf{X},L}$	family of all rule-sets specifying classifiers from $\mathbf{X}$ to L
	$\mathcal{S}_{\mathbf{X},L}$	family of all SUFs from $\mathbf{X}$ to L
	$\mathbb{S}$-$\mathbb{SET}$	function translating SUFs into rule-sets
	$\mathbb{SUF}$	function translating rule-sets into SUFs
Section 5	$\lambda^-_{\mathcal{D}}$, $\lambda^+_{\mathcal{D}}$	smallest and greatest classifiers that fit a monotonic $\mathcal{D}$
	$\mathcal{L}$	relabeling function
	$\mathcal{M}$	relabeled dataset
	$r \backslash I$	rule r where the thresholds of I are deactivated

Table 1: Notation used throughout the paper together with the section in which it is introduced.

2. Rules for monotonic classification

In this section, we provide the formal framework for the monotonic classification of qualitative data and then we describe the links between monotonic rules and monotonic functions.

Throughout this paper, for any natural number n, we will denote the set of natural numbers $\{1, \ldots, n\}$ by $[n]$.

2.1. Monotonic classification

We consider the following setting. Let $X_1, \ldots, X_n$ be bounded totally ordered sets called *attribute domains*, and $\mathbf{X} = X_1 \times \cdots \times X_n$, their Cartesian product. An element of $\mathbf{X}$ is denoted by $\mathbf{x}$. Let L be a bounded totally ordered set of *classes* (also called *labels*). An *observation* is a pair $(\mathbf{x}, y) \in \mathbf{X} \times L$. The n-tuple $\mathbf{x} \in \mathbf{X}$ is called the *description* of the observation, and y is called its class (or label). The components of $\mathbf{x}$ are denoted by $x_1, \ldots, x_n$; for each $i \in [n]$, x_i belongs to X_i and is called an *attribute value*. Since the set of classes is finite and totally ordered, we represent it as an ordered set $L = \{a_1 = 0 < a_2 < \cdots < a_{|L|} = 1\}$. Since $X_1, \ldots, X_n, L$ are finite, we denote their lower bounds by 0 and their upper bounds by 1.

A function $f : \mathbf{X} \to L$ whose aim is to predict the class of an observation from its description is called a *classifier*. A monotonically nondecreasing function f is here called a *nondecreasing classifier*. A *dataset* is a multiset of available observations $\mathcal{D} = \{(x_1^i, \ldots, x_n^i, y^i)\}_{i=1}^m \subseteq \mathbf{X} \times L$. Let a function $\eta_{\mathcal{D}} : \mathbf{X} \times L \to \mathbb{N}$ indicate the number of times each pair $(\mathbf{x}, y)$ is present in $\mathcal{D}$ (for all $(\mathbf{x}, y) \in \mathcal{D}$ we have $\eta_{\mathcal{D}}(\mathbf{x}, y) > 0$, and 0 otherwise). The total number of available observations is denoted by $|\mathcal{D}|$, and defined as

$$|\mathcal{D}| = \sum_{(\mathbf{x},y) \in \mathbf{X} \times L} \eta_{\mathcal{D}}(\mathbf{x}, y).$$

Finally, we denote by $\mathbf{X}_{\mathcal{D}}$ the set of descriptions appearing in $\mathcal{D}$, i.e.,

$$\mathbf{X}_{\mathcal{D}} = \big\{ \mathbf{x} \;\big|\; (\mathbf{x}, y) \in \mathcal{D} \big\}.$$

The purpose of the paper is to learn a nondecreasing classifier from $\mathbf{X}$ to L.

Remark 1. In some practical cases, one could want to learn a classifier that is nondecreasing on certain attributes and non increasing on others. However, this problem can be reduced to monotonic classification by a suitable reordering of the attribute domains.

Whether we want to measure the fitness of a classifier on available data or its predictive accuracy on new observations, we need empirical error functions. The empirical error of a classifier f on a dataset $\mathcal{D}$ is the average loss on each observation of $\mathcal{D}$, given by

$$E_\ell(f, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},y)\in \mathbf{X}\times L} \eta_{\mathcal{D}}(\mathbf{x}, y) \cdot \ell\big(f(\mathbf{x}), y\big)$$

where $\ell : L^2 \to \mathbb{R}$ is a *loss function* that gives the "cost" $\ell(a_i, a_j)$ of predicting that an observation belongs to class i when it actually belongs to class j. As usual, we assume that any loss function ℓ verifies $\ell(a_i, a_j) = 0$, whenever $i = j$. A widely used loss function is the 0-1 loss ℓ_{0-1} (also known as the *Kronecker delta*) defined by

$$\ell_{0-1}(a_i, a_j) = \begin{cases} 0 \text{ if } i = j, \\ 1 \text{ if } i \neq j. \end{cases}$$

The empirical error of f on $\mathcal{D}$ with respect to ℓ_{0-1} is called *misclassification error rate (MER)*.

Since the classes are ordered, it is reasonable to associate greater losses to predictions that are "far" from the actual class. More precisely, it is natural to require that $\ell(a_i, a_j) < \ell(a_i, a_k)$ in all cases where $i < j < k$. Loss functions that satisfy $\ell(a_i, a_j) \leq \ell(a_i, a_k)$ for all classes $i < j < k$ are called *V-shaped loss functions* [44]. When there is no information allowing to define a specific distance between classes, the ℓ_1 loss defined by

$$\ell_1(a_i, a_j) = |i - j|$$

is often used by default. The empirical error with respect to this loss is equivalent to the *mean absolute error (MAE)*, when the classes in L are treated as quantitative values.

2.2. Sets of selection rules

In general, a decision rule is an implication of the form

$$\text{If} \quad x_1 \in A_1 \text{ and } \ldots \text{ and } x_n \in A_n, \quad \text{then} \quad y \in B,$$

where $A_i \subseteq X_i$ for each $i \in \{1, \ldots, n\}$ and $B \subseteq L$. However, since we assume that the class increases with the attribute values, we can consider particular types of rules, of the form (see for instance [37]):

$$\text{If} \quad x_1 \geq \alpha_1 \text{ and } \ldots \text{ and } x_n \geq \alpha_n, \quad \text{then} \quad y \geq \delta \quad (\textit{selection rules}),$$

or of the form

$$\text{If} \quad x_1 \leq \alpha_1 \text{ and } \ldots \text{ and } x_n \leq \alpha_n, \quad \text{then} \quad y \leq \delta \quad (\textit{rejection rules}).$$

where $\delta \in L$ and, for all $i \in [n]$, $\alpha_i \in X_i$. In what follows, we focus on selection rules and we call a set of selection rules a *monotonic rule set* (or just a rule set, whenever the meaning is clear from the context). We use the shorthand notation $r = (\alpha_1^r, \ldots, \alpha_n^r) \to \delta^r$ or even $\boldsymbol{\alpha}^r \to \delta^r$ for selection rules. Note that any threshold $\alpha_i^r = 0$ corresponds to the trivial condition $x_i \geq 0$, which is always verified. Such a condition does not provide information and can be removed from the left-hand side of any rule. We say that an attribute $i \in [n]$ is *active* in r if $\alpha_i^r > 0$. The set of all active attributes in a rule r is denoted by A^r and defined by

$$A^r = \{i \in [n] \mid \alpha_i^r > 0\}.$$

The rule r can be written as a condition on active attributes only:

$$\forall i \in A^r,\ x_i \geq \alpha_i^r \quad \Longrightarrow \quad y \geq \delta^r.$$

A monotonic rule-set defines a nondecreasing function from $\mathbf{X}$ to L. For any rule r, we define the function $f_r : \mathbf{X} \to L$ by

$$f_r(x_1, \ldots, x_n) = \begin{cases} \delta^r \text{ if } \forall i \in [n],\ x_i \geq \alpha_i^r, \\ 0 \text{ otherwise.} \end{cases}$$

For any rule-set R we define the function $f_R : \mathbf{X} \to L$ by

$$f_R(\mathbf{x}) = \bigvee_{r \in R} f_r(\mathbf{x}).$$

Such rule-sets are quite expressive: any monotonic dataset $\mathcal{D}$ can be interpolated by a rule-set, as shown by Greco *et al.* [38]. These authors made early contributions to the representation of monotonic datasets by rules, using the so-called Dominance-based Rough Set Approach (DRSA) [37].

In fact any nondecreasing function $\mathbf{X} \to L$ such that $f(1, \ldots, 1) = 1$ and $f(0, \ldots, 0) = 0$ can be represented by a set D_f of selection rules of the form $D_f = \{(x_1, \ldots, x_n) \to f(x_1, \ldots, x_n) : x \in \mathbf{X}\}$. We say that a monotonic rule-set R is *equivalent* to a nondecreasing function f if $f_R = f$. It is easy to see that $f_{\mathcal{D}_f} = f$.

Let R and R' be two rule-sets. We say that R and R' are *equivalent* if $f_R = f_{R'}$, and we denote by $\mathrm{Eq}(R)$ the family of rule-sets equivalent to R. The fact that some sets of rules are equivalent comes from the fact that

certain rules bring more information than some others. Let r and s be two rules. We say that s is *redundant* with respect to r (denoted by $r \Rightarrow s$) if and only if

$$\delta^s \leq \delta^r \text{ and } \forall i \in [n],\ \alpha_i^r \leq \alpha_i^s.$$

Note that $r \Rightarrow s$ is equivalent to $f_s \leq f_r$ and to $f_{\{r,s\}} = f_{\{r\}}$. Let R be a set of rules. We say that R is *non-redundant* if $r \Rightarrow s$ is verified for no $r, s \in R$. We define the set R^* by

$$R^* = \left\{ s \in R \;\middle|\; \forall r \in R,\ \left[r \not\Rightarrow s \text{ or } s = r \right] \right\}.$$

The following proposition shows its uniqueness. Its proof can be found in the Appendix.

Proposition 1. *For any rule-set R, R^* is the smallest element of* $\mathrm{Eq}(R)$. *This is to say, R^* is the unique non-redundant rule-set equivalent to R.*

Remark 2. All notions that we just introduced have a dual counterpart for rejection rules. Indeed, a rejection rule from $\mathbf{X}$ to L can be seen as a selection rule from $\mathbf{X}^{-1}$ to L^{-1}, where $\mathbf{X}^{-1}$ and L^{-1} denote the sets $\mathbf{X}$ and L with reversed orders. Therefore, notions pertaining to rejection rules similar to the above ones for selection rules can be defined, by swapping $\geq$ and $\leq$, $\wedge$ and $\vee$, 0 and 1, etc. in the corresponding definitions. For instance, each rejection rule induces a function

$$f^r(\mathbf{x}) = \begin{cases} \delta^r \text{ if } \forall i \in [n],\ x_i \leq \alpha_i^r, \\ 1 \text{ otherwise.} \end{cases}$$

and a set of rejection rules R yields the function $f^R(\mathbf{x}) = \bigwedge_{r \in R} f^r(\mathbf{x})$. For the sake of brevity we do not explain the results concerning rejection rules.

3. Qualitative aggregation operators for monotonic classification

In this section we introduce Sugeno integrals and Sugeno Utility Functionals, and we explain how they are related to decision rules.

Sugeno integrals, like Choquet integralq, are aggregation functions that rely on the notion of *capacity* [36]. These two aggregation functions have been studied in Multiple Criteria Decision Aid (MCDA), where they are used to aggregate local utility values (i.e., evaluations of an alternative w.r.t. several attributes or several points of view) into a single utility value [34, 35]. Note however that Sugeno integral is the aggregation function of choice on purely ordinal and non-numerical scales [28].

3.1. The Sugeno integral

Let L be a bounded ordered set. A set function $\mu : 2^{[n]} \to L$ is a *capacity* if

$$\forall I \subseteq J \subseteq [n], \quad \mu(I) \leq \mu(J)$$

and if $\mu(\emptyset) = 0$ and $\mu([n]) = 1$. For each $I \subseteq [n]$, the value $\mu(I)$ can be interpreted as an importance value given to the subset of attributes I. The Sugeno integral [52] w.r.t. μ is the function $S_\mu : L^n \to L$ defined by

$$S_\mu(\mathbf{x}) = \bigvee_{I \subseteq [n]} \mu(I) \wedge \bigwedge_{i \in I} x_i.$$

The *focal sets* of μ are the sets $F \subseteq [n]$ such that

$$\mu(F) > \bigvee_{I \subset F} \mu(I).$$

We denote by $\mathcal{F}(\mu)$ the family of focal sets of μ. The value of μ on each subset of $[n]$ is entirely determined by its values on the focal sets, and we obtain a more compact expression of Sugeno integral:

$$S_\mu(\mathbf{x}) = \bigvee_{I \in \mathcal{F}(\mu)} \mu(I) \wedge \bigwedge_{i \in I} x_i.$$

3.2. Sugeno Utility Functionals

Sugeno integrals allow to aggregate values belonging to the same scale. The so-called Sugeno Utility Functionals (SUF) generalize Sugeno integrals and enable the fusion of values coming from different scales. They seem to have been considered for the first time in [38]. The authors of [19, 20] define a SUF as the composition of a Sugeno integral with functions $\varphi_i : X_i \to L$, one for each attribute, where

$$\varphi_i(0) = 0 \quad \text{and} \quad \varphi_i(1) = 1.$$

Here, we consider SUFs for which $\varphi_1, \ldots, \varphi_n$ are nondecreasing (see [17]).

A function $\varphi_i : X_i \to L$ is called *local qualitative normalization* function (local QNF) if it is nondecreasing and if it satisfies $\varphi_i(0) = 0$ and $\varphi_i(1) = 1$. A function $\varphi : \mathbf{X} \to L^n$ is called *global qualitative normalization* function (global QNF) if it is the Cartesian product of n local QNFs, i.e., if

$$\varphi(\mathbf{x}) = (\varphi_1(x_1), \ldots, \varphi_n(x_n)),$$

where, for each $i \in [n]$, the function $\varphi_i : X_i \to L$ is a local QNF. Remark that the product of local QNFs characterizing a global QNF is unique; thus, for

any $\varphi : \mathbf{X} \to L$, we will denote by $\varphi_1, \ldots, \varphi_n$ the local QNFs whose product equals φ.

A *Sugeno Utility Functional (SUF)* from $\mathbf{X}$ to L consists in the composition of a Sugeno integral from L^n to L with a global QNF from $\mathbf{X}$ to L^n. Let $\mu : 2^{[n]} \to L$ be a capacity, $\varphi : \mathbf{X} \to L$ be a global QNF. The SUF defined w.r.t. μ and φ is the function $S_{\mu,\varphi} : \mathbf{X} \to L$ defined by

$$\begin{aligned} S_{\mu,\varphi}(x_1, \ldots, x_n) &= S_\mu\big(\varphi_1(x_1), \ldots, \varphi_n(x_n)\big) \\ &= \bigvee_{I \in \mathcal{F}(\mu)} \mu(I) \wedge \bigwedge_{i \in I} \varphi_i(x_i). \end{aligned}$$

Remark 3. All Sugeno integrals are SUFs (where $X_1 = \cdots = X_n = L$ and where $\varphi_1, \ldots, \varphi_n$ are identities).

Since Sugeno integrals are expressed using the operators $\wedge$ and $\vee$, they can easily be translated into decision rules [37, 51]; obviously, SUFs also possess this interesting property. Therefore, each prediction made by a SUF can be explained by at least one decision rule, and the local interpretability of SUFs is at least as good as that of decision rules.

Moreover, a SUF being the composition of a global QNF and a capacity, it constitutes a more compact form of a large rule-set than its usual expression as a list of rules. Note that the extent to which this is the case certainly depends on several parameters: the size of the rule-set, the number of focal sets, etc.

3.3. The expressiveness of SUFs

Let us illustrate the notion of SUF with a simple example.

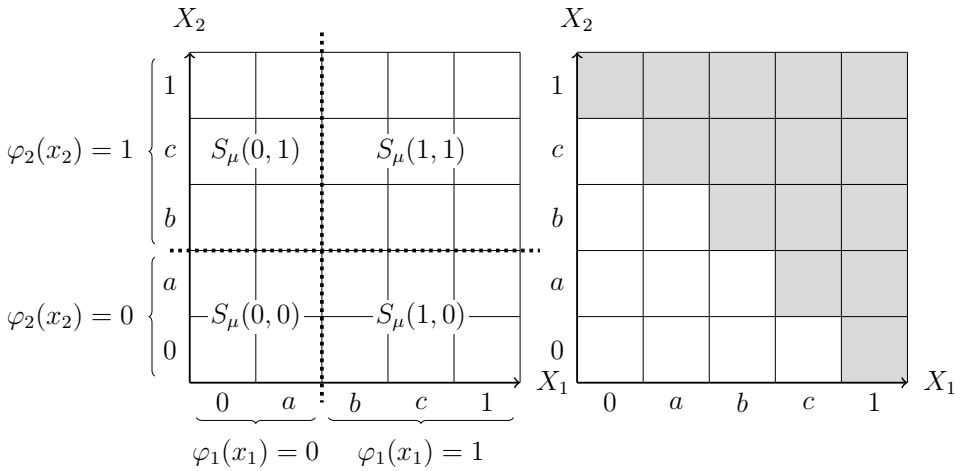


Figure 1: The figure on the left depicts local QNFs φ_1 and φ_2. These two functions divide $\mathbf{X}$ in four parts, and map each part to a point of L^2, to which Sugeno integral S_μ is applied. The figure on the right depicts a "stair-shaped" function, which is a typical example of a function that cannot be expressed in the form of a SUF.

Example 1. *Let $L = \{0, 1\}$, and $\boldsymbol{X} = X_1 \times X_2$, with $X_1 = X_2 = \{0, a, b, c, 1\}$, where $0 < a < b < c < 1$. We define the mappings $\varphi_1 : X_1 \to L$ and $\varphi_2 : X_2 \to L$ by*

$$\varphi_1(x_1) = \begin{cases} 0 \text{ if } x_1 \leq a, \\ 1 \text{ otherwise} \end{cases} \qquad \text{and} \quad \varphi_2(x_2) = \begin{cases} 0 \text{ if } x_2 \leq a, \\ 1 \text{ otherwise.} \end{cases}$$

When the functions φ_1 and φ_2 are specified, the definition of $S_{\mu,\varphi}$ depends on the choice of the values of μ, as shown in Figure 1. It shows in particular that there exist nondecreasing functions that cannot be expressed as SUFs, such as the function on the right. The latter corresponds to the monotonic rule set $\{(0,1) \to 1, (a,c) \to 1, (b,b) \to 1, (c,a) \to 1, (1,0) \to 1\}$. □

The last example highlights the fact that SUFs are less expressive than rule-sets. In fact, SUFs suffer from several limitations, which come from the loss of information induced by QNFs and from the poor expressiveness of Sugeno integrals, first highlighted by Greco et al. [37, 51, 38]. Firstly, for any SUF $S_{\mu,\varphi}$, the global QNF φ associates to each description $\mathbf{x} \in \mathbf{X}$ a value $\varphi(\mathbf{x}) \in L^n$, and we have

$$S_{\mu,\varphi}(\mathbf{x}) = S_\mu(\varphi(\mathbf{x})).$$

We can see L^n as an intermediary description space, to which $\mathbf{X}$ is mapped through $\varphi_1 \times \cdots \times \varphi_n$. Thus, the size of L^n is determined by the number

of classes, which is quite restrictive when L has a small size, in particular, in the case of binary classification, as we illustrated in the last example. Secondly, Sugeno integrals constitute a more restricted class of functions than the aggregation functions on L, see [15] and references therein. For instance it does not include associative operations studied by Fodor [33]. The limitations of SUFs follow from the following characterization [17, 21]:

Proposition 2. *A nondecreasing function $f : \boldsymbol{X} \to L$ is a SUF if and only if, for all $\mathbf{x}, \mathbf{x'} \in \boldsymbol{X}$, $i \in [n]$ and $a \in X_i$ we have*

$$f(\mathbf{x}|_i^0) < f(\mathbf{x}|_i^a) \text{ and } f(\mathbf{x'}|_i^a) < f(\mathbf{x'}|_i^1) \quad \Longrightarrow \quad f(\mathbf{x}|_i^a) < f(\mathbf{x'}|_i^a)$$

where $\mathbf{x}|_i^a$ denotes the tuple which is equal to $\mathbf{x}$ on every component except the i-th one, which equals a.

In the next section we show how SUFs can be translated into sets of selection rules. For the converse, We need a model introduced in [16], that consists in the maximum (or minimum) of several SUFs.

4. From SUFs to rule-sets and back

For any SUF S, there is exactly one non-redundant set of selection rules R such that $S = f_R$ and a non-redundant set of rejection rules R' such that $S = f^{R'}$. In this Section we define the function $\mathbb{S}\text{-}\mathbb{SET} : \mathcal{S}_{\mathbf{X},L} \to \mathcal{R}_{\mathbf{X},L}$, which associates to each SUF its equivalent non-redundant rule-set.

4.1. Translating a Sugeno integral into a rule-set

Let $S_\mu : L^n \to L$ be the Sugeno integral w.r.t. the capacity μ. This Sugeno integral is of the form:

$$S_\mu(\mathbf{x}) = \bigvee_{F \in \mathcal{F}(\mu)} g_F(\mathbf{x}), \quad \text{where} \quad g_F(\mathbf{x}) = \mu(F) \wedge \bigwedge_{i \in F} x_i.$$

Each term g_F can be represented by a rule-set as follows. For each $\delta \in L$, the inequality

$$\mu(F) \wedge \bigwedge_{i \in F} x_i \geq \delta$$

holds if and only if

$$\mu(F) \geq \delta \quad \text{and} \quad \forall i \in F,\ x_i \geq \delta.$$

Therefore, g_F is equivalent to the following non-redundant rule-set

$$\mathbb{S}\text{-}\mathbb{SET}(g_F) = \Big\{ \big[\forall i \in F,\ x_i \geq \delta\big] \Rightarrow y \geq \delta \ \Big|\ \delta \in L,\ \delta \leq \mu(F) \Big\}.$$

and S_μ is equivalent to the non-redundant rule-set

$$\mathbb{S}\text{-}\mathbb{SET}(S_\mu) = \left(\bigcup_{F \in \mathcal{F}(\mu)} \mathbb{S}\text{-}\mathbb{SET}(g_F) \right)^*.$$

where, as previously, for any rule-set R, R^* denotes its smallest equivalent rule-set. See [26] for a concrete example of a qualitative dataset represented by a Sugeno integral from which a rule-set is derived.

4.2. Translating a SUF into a rule-set

For any capacity μ and any global QNF φ, the SUF $S_{\mu,\varphi}$ verifies

$$S_{\mu,\varphi}(\mathbf{x}) = S_\mu(\varphi_1(x_1), \ldots, \varphi_n(x_n))$$

for all $\mathbf{x} \in L$. Consequently, $S_{\mu,\varphi}$ can be expressed by the following set of rules

$$\bigcup_{F \in \mathcal{F}(\mu)} \Big\{ \big[\forall i \in F,\ \varphi_i(x_i) \geq \delta\big] \Rightarrow y \ \Big|\ \delta \in L,\ \delta \leq \mu(F) \Big\}.$$

Strictly speaking, this set is not a rule-set, since its rules are formulated w.r.t. the values of $\varphi_1(x_1), \ldots, \varphi_n(x_n)$. However, for any $i \in [n]$, the inequality

$$\varphi_i(x_i) \geq \delta$$

is equivalent to

$$x_i \geq \alpha(i, \delta) = \bigwedge \big\{ a_i \in X_i \ \big|\ \varphi_i(a_i) \geq \delta \big\}.$$

The SUF $S_{\mu,\varphi}$ should therefore equivalent to the following non-redundant rule-set:

$$\mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi}) = \left(\bigcup_{F \in \mathcal{F}(\mu)} \Big\{ \big[\forall i \in F,\ x_i \geq \alpha(i, \delta)\big] \Rightarrow y \geq \delta \ \Big|\ \delta \in L,\ \delta \leq \mu(F) \Big\} \right)^*$$

as we shall see later.

4.3. Translating a rule-set into a SUF

We say that a rule-set R is *SUF-representable* if it is equivalent to a SUF, that is, the function f_R induced by R in Section 2.2 is a SUF. We define the function

$$\mathbb{SUF} : \mathcal{R}_{\mathbf{X},L} \to \mathcal{S}_{\mathbf{X},L}$$

that associates a SUF to each rule-set as follows: if R is a non-redundant rule-set,

$$\mathbb{SUF}(R) = S_{\mu,\varphi}$$

where μ is the capacity defined by

$$\mu(I) = \bigvee\{\delta^r \mid r \in R,\ A^r \subseteq I\}, \tag{1}$$

for all $\emptyset \subset I \subset [n]$, and where φ is the global QNF defined by

$$\varphi_i(x_i) = \bigvee\{\delta^r \mid r \in R,\ 0 < \alpha_i^r \leq x_i\} \tag{2}$$

for all $i \in [n]$ and $x_i \in X_i \backslash \{0, 1\}$. Note that we always have $\mu(\emptyset) = 0$, $\mu([n]) = 1$, and $\varphi_i(0) = 0$ and $\varphi_i(1) = 1$ for all $i \in [n]$.

More precisely, we can also see $\mathbb{SUF}(R)$ as the SUF iteratively defined by the following steps.

1. We set

$$\mu(I) = \begin{cases} 1 \text{ if } I = [n], \\ 0 \text{ otherwise,} \end{cases}$$

and, for all $i \in [n]$,

$$\varphi_i(x_i) = \begin{cases} 1 \text{ if } x_i = 1, \\ 0 \text{ otherwise.} \end{cases}$$

The SUF $S_{\mu,\varphi}$ so-defined is the smallest SUF from $\mathbf{X}$ to L.

2. For each rule $r \in R$

   (a) we increase $\mu(A^r)$ to δ^r,

   (b) for each $i \in A^r$, we increase $\varphi_i(\alpha_i)$ to δ^r.

Remark 4. In the previous procedure, "we increase $g(x)$ to y" means that, for each x' belonging to the domain of g and verifying $x' \geq x$, if $g(x') < y$ then we set $g(x')$ to y.

Note that for any rule-set R we have $\mathbb{SUF}(R^*) \geq R$. We first prove that if a rule-set R is obtained from a SUF $S_{\mu,\varphi}$ as explained in the previous subsection, then $\mathbb{SUF}(R) = S_{\mu,\varphi}$.

Proposition 3. *For any SUF S we have*

$$S = f_{\mathbb{S}\text{-}\mathbb{SET}(S)} = \mathbb{SUF}(\mathbb{S}\text{-}\mathbb{SET}(S)).$$

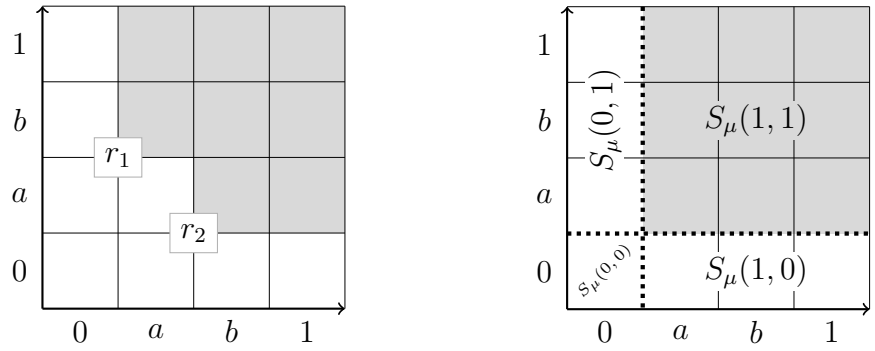


Figure 2: Representation of the function defined by the rules $r_1 = \big[(a, b) \to 1\big]$ and $r_2 = \big[(b, a) \to 1\big]$ (left) and the function $\mathbb{SUF}(\{r_1, r_2\})$ (right). The dashed lines show how $\mathbf{X}$ is partitioned by φ_1 and φ_2.

Corollary 4. *A rule-set R is SUF-representable if and only if $\mathbb{SUF}(R^*) = f_R$.*

This corollary allows to check whether a rule-set R is SUF-representable by testing if $R^* = \mathbb{S}\text{-}\mathbb{SET}(\mathbb{SUF}(R^*))$. The proof of Proposition 3 and its corollary are given in the Appendix.

Example 2. *Consider $L = \{0, 1\}$, $X_1 = X_2 = \{0, a, b, 1\}$ with $0 < a < b < 1$, and the rules $r_1 = \big[(a, b) \to 1\big]$ and $r_2 = \big[(b, a) \to 1\big]$. We will compute φ_1, φ_2 and μ such that $\mathbb{SUF}(\{r_1, r_2\}) = S_{\mu,\varphi}$. The rule r_1 implies*

$$\varphi_1(a) = 1 \quad \textit{and} \quad \varphi_2(b) = 1.$$

The rule r_2 implies

$$\varphi_1(b) = 1 \quad \textit{and} \quad \varphi_2(a) = 1.$$

Those two rules imply the trivial condition $\mu(\{1, 2\}) = 1$. Therefore we have

$$\varphi_1(a) = 1, \quad \varphi_2(a) = 1 \quad \textit{and} \quad \mu(\{1, 2\}) = 1.$$

The functions $f_{\{r_1,r_2\}}$ and $\mathbb{SUF}(\{r_1, r_2\})$ are depicted in Figure 2. The translation of $\mathbb{SUF}(\{r_1, r_2\})$ into rules reduces to the single rule $(a, a) \to 1$. We have $\mathbb{SUF}(\{r_1, r_2\}) > f_{\{r_1,r_2\}}$, and thus $\{r_1, r_2\}$ is not SUF-representable, for the same reason as for the function on the right of Figure 1. □

For any non-redundant rule-sets R and R' such that $R \subseteq R'$, we have $\mathbb{SUF}(R) \leq \mathbb{SUF}(R')$. However, for two rule-sets R and R', it is possible that $f_R < f_{R'}$ while $\mathbb{SUF}(R') < \mathbb{SUF}(R)$, as we show in the following example.

Example 3. *Consider* $X_1 = X_2 = L = \{0, a, b, c, 1\}$*, and the non-redundant rule-sets*

$$R = \{(b, a) \to 1,\ (0, b) \to 1,\ (c, 0) \to 1\}$$

and

$$R' = \{(b, 0) \to 1,\ (0, b) \to 1\}.$$

Note that $f_R < f_{R'}$*. Moreover, the SUF* $S_{\mu,\varphi}$ *defined as follows is equal to* $\mathbb{SUF}(R)$*. We have* $\mu(\{1\}) = \mu(\{2\}) = 1$ *and:*

x	*0*	*a*	*b*	*c*	*1*
$\varphi_1(x)$	*0*	*0*	*1*	*1*	*1*
$\varphi_2(x)$	*0*	1	1	1	*1*

Similarly, the SUF $S_{\mu',\varphi'}$ *defined as follows is equal to* $\mathbb{SUF}(R')$*. We have* $\mu'(\{1\}) = \mu'(\{2\}) = 1$ *and:*

x	*0*	*a*	*b*	*c*	*1*
$\varphi'_1(x)$	*0*	*0*	*1*	*1*	*1*
$\varphi'_2(x)$	*0*	*0*	1	1	*1*

□

Thus we have $f_R < f_{R'}$ and $\mathbb{SUF}(R') < \mathbb{SUF}(R)$. This phenomenon is due to the expression of QNFs (2), in which non-active attributes are ignored. Fewer attributes are active in the rule $(b, 0) \to 1$ than in $(b, a) \to 1$, even though $f_{(b,a)\to 1} < f_{(b,0)\to 1}$.

4.4. Translating monotonic rule-sets into SUF-sets

As pointed out earlier, any aggregation function $f : \mathbf{X} \to L$, namely nondecreasing and $f(0, \ldots, 0) = 0 \quad \text{and} \quad f(1, \ldots, 1) = 1$ can be represented by a set of selection rules $\mathcal{D}_f$. Each rule $r^i = \boldsymbol{\alpha}_i \to \delta_i \in (\mathcal{D}_f)^*$ can be represented by a min-term of the form $f_i = \min(\min_{j\in A_i} \varphi_{ij}(x_j), \delta_i)$ where A_i contains the active attributes of $r^i, \varphi_{ij}(\alpha_j) = \delta_i$, each mapping $\varphi_{i,j}$ being a local QNF, for the min-term indexed by i.

Let $\mathbf{S}$ a set of SUFs. We call $\bigvee \mathbf{S}$ the function defined by

$$\bigvee \mathbf{S}(\mathbf{x}) = \bigvee_{S\in\mathbf{S}} S(\mathbf{x}).$$

The following proposition lays bare the expressive power of functions of the form $\bigvee \mathbf{S}$ as shown in [16].

Proposition 5. *For any nondecreasing function $f : \boldsymbol{X} \to L$ such that*

$$f(0,\ldots,0) = 0 \quad \textit{and} \quad f(1,\ldots,1) = 1,$$

there exists a set of SUFs $\boldsymbol{S}$ such that $f = \bigvee \boldsymbol{S}$.

Proof. We express f as

$$f(x_1,\ldots x_n) = \bigvee_{i\in I}\Big(\delta_i \wedge \bigwedge_{j\in A_i} \varphi_{ij}(x_j)\Big),$$

translating each rule in $(\mathcal{D}_f)^*$. Note that the domain L^n, can be partitioned into subsets where $f(x_1,\ldots x_n) = \varphi_{ij}(x_j)$ or is a constant $\delta_i \in L$. Moreover, as $f(1,1,\ldots,1) = 1$, there exists $i \in I$ such that $\delta_i = 1$, and $\forall j \in A_i, \varphi_{ij}(1) = 1$ and $\varphi_{ij}(0) = 0$. It is clear that we can rewrite f as

$$f(x_1,\ldots,x_n) = \bigvee_{i\in I}\left[\Big(\delta_i \wedge \bigwedge_{j\in A_i} \varphi_{ij}(x_j)\Big) \vee \Big(\bigwedge_{k\in[n]} \varphi_{ik}(x_k)\Big)\right],$$

provided that $\forall k \notin A_i, \varphi_{ik}(1) = 1$ and $\varphi_{ik}(x_k) = 0$ if $x_k < 1$. The inner expression

$$S_i(\mathbf{x}) = \left[\delta_i \wedge \bigwedge_{j\in A_i} \varphi_{ij}(x_j)\right] \vee \left[\bigwedge_{k\in[n]} \varphi_{ik}(x_k)\right]$$

is a SUF with respect to the capacity μ_i with focal sets A_i such that $\mu_i(A_i) = \delta_i$, and $[n]$ such that $\mu_i([n]) = 1$ in the case $A_i \neq [n]$. So we have $f(\mathbf{x}) = \bigvee \mathbf{S}(\mathbf{x}) = \bigvee_{i\in I} S_i(\mathbf{x})$. □

This result completes those in [38], where it is shown that monotonic functions correspond to selection rules (Th. 1), and Sugeno integrals to single threshold selection rules (Th. 3).

The representation obtained in the proof of Proposition 5 is not parsimonious. Some min-terms can be grouped into a single SUF with respect to a more complex capacity by unifying some local QNFs for each attribute into a single one. To do so, the idea is that we extract a maximal number of subsets $A_i \subset [n]$, such that

- whenever $A_i \cap A_{i'} \neq \emptyset$, the utility functions φ_{ij} and $\varphi_{i'j}$ for all $j \in A_i \cap A_{i'}$ must be equal.
- whenever $A_i \subset A_{i'}$, we have that $\delta_i < \delta_{i'}$.

In order to determine a set of SUFs equivalent to a given rule-set, we rely on the notion of *SUF-cover*. A family $\mathbf{P}$ of rule-sets is a *cover* of a rule-set R if

$$\bigcup_{P \in \mathbf{P}} P = R.$$

Moreover, a cover $\mathbf{P}$ of R is a *SUF-cover* of R if the SUF-set

$$\mathbf{S} = \{\mathbb{SUF}(P) \mid P \in \mathbf{P}\}$$

is equivalent to R, that is to say if $\bigvee \mathbf{S} = f_R$. In fact, it is easy to show that $\mathbf{P}$ is a SUF-cover of R if and only if

$$\forall P \in \mathbf{P}, \quad \mathbb{SUF}(P) \leq f_R. \tag{3}$$

Indeed, this condition is necessary. It is also sufficient, because when it holds we have

$$f_R \geq \bigvee_{P \in \mathbf{P}} \mathbb{SUF}(P) \geq \bigvee_{P \in \mathbf{P}} f_P = \bigvee_{P \in \mathbf{P}} \bigvee_{r \in P} f_r = f_R.$$

The main difficulty is to find a minimal SUF cover that accounts for a monotonic dataset $\mathcal{D}$ or the corresponding bunch of selection rules R, so as to have $f_{\mathcal{D}} = f_R = \bigvee \mathbf{S}$ where $\mathbf{S}$ contains a minimal number of SUFs.

5. A rule set learning algorithm

The theoretical results obtained above motivates new algorithms for monotonic classification, exploiting the synergy between rule sets and SUFs. In this section, we introduce and evaluate a non-parametric monotonic rule-set learning algorithm that we call SRL.

5.1. Interpolation of a monotonic dataset by short rules

Let $\mathcal{D}$ be a monotonic dataset, i.e., a dataset in which all pairs $\{(\mathbf{x}, y), (\mathbf{x'}, y')\} \subseteq \mathcal{D}$ verify

$$\mathbf{x} \leq \mathbf{x'} \quad \Longrightarrow \quad y \leq y'.$$

We define functions $\lambda^-_{\mathcal{D}} : \mathbf{X} \to L$ and $\lambda^+_{\mathcal{D}} : \mathbf{X} \to L$ by

$$\lambda^-_{\mathcal{D}}(\mathbf{x}) = \bigvee \left\{ y' \;\middle|\; (\mathbf{x'}, y') \in \mathcal{D} \text{ and } \mathbf{x'} \leq \mathbf{x} \right\}$$
$$\lambda^+_{\mathcal{D}}(\mathbf{x}) = \bigwedge \left\{ y' \;\middle|\; (\mathbf{x'}, y') \in \mathcal{D} \text{ and } \mathbf{x} \leq \mathbf{x'} \right\}.$$

They are (respectively) the lowest and the highest classifiers that make no error on $\mathcal{D}$. For all $\mathbf{x} \in \mathbf{X}$ we have $\lambda^-_{\mathcal{D}}(\mathbf{x}) \leq \lambda^+_{\mathcal{D}}(\mathbf{x})$, and for all $(\mathbf{x}, y) \in \mathcal{D}$ we have

$$\lambda^-_{\mathcal{D}}(\mathbf{x}) = \lambda^+_{\mathcal{D}}(\mathbf{x}) = y.$$

We say that a function f *interpolates* $\mathcal{D}$ if $f(\mathbf{x}) = y$ for all $(\mathbf{x}, y) \in \mathcal{D}$. The interval $[\lambda^-_{\mathcal{D}}(\mathbf{x}), \lambda^+_{\mathcal{D}}(\mathbf{x})]$ contains all nondecreasing functions that interpolate $\mathcal{D}$.

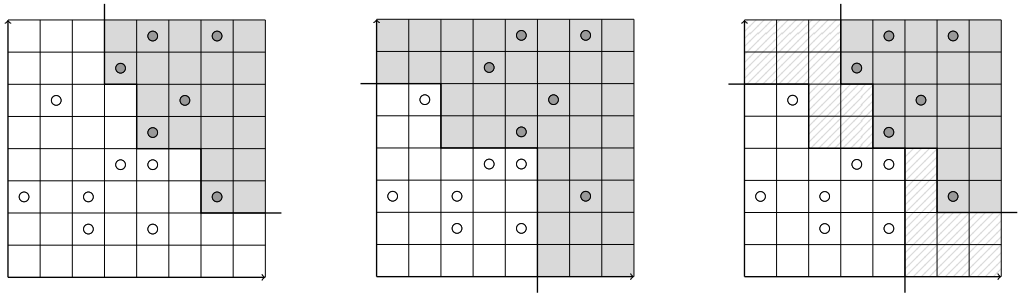


Figure 3: Example of monotonic data; two attributes with discrete domains, two classes (0 and 1). Each possible description is represented by a square of the grid. Observations of class 0 (resp. 1) are depicted by white (resp. gray) circles. The first schema depicts $\lambda^-_{\mathcal{D}}$ as a frontier separating all $\mathbf{x} \in \mathbf{X}$ such that $\lambda^-_{\mathcal{D}}(\mathbf{x}) = 0$ (in the white zone) from all $\mathbf{x} \in \mathbf{X}$ such that $\lambda^-_{\mathcal{D}}(\mathbf{x}) = 0$ (in the gray zone). The second schema depicts $\lambda^+_{\mathcal{D}}$ in an analogous manner. The third schema depicts the interval $[\lambda^-_{\mathcal{D}}, \lambda^+_{\mathcal{D}}]$: white (resp. gray) zones correspond to all $\mathbf{x} \in \mathbf{X}$ such that $f(\mathbf{x}) = 0$ (resp. such that $f(\mathbf{x}) = 1$) for all $f \in [\lambda^-_{\mathcal{D}}, \lambda^+_{\mathcal{D}}]$. The shaded zone contains all $\mathbf{x} \in \mathbf{X}$ for which $\lambda^-_{\mathcal{D}}(\mathbf{x}) < \lambda^+_{\mathcal{D}}(\mathbf{x})$.

We denote by R^- the non-redundant set of selection rules equivalent to $\lambda^-_{\mathcal{D}}$, and by R^+ the non-redundant set of rejection rules equivalent to $\lambda^+_{\mathcal{D}}$. The functions $f_{R^-} = \lambda^-_{\mathcal{D}}$ and $f^{R^+} = \lambda^-_{\mathcal{D}}$ are respectively the least and greatest interpolations of $\mathcal{D}$. Let us focus on R^-. We have

$$R^- = \left\{ \mathbf{x} \to y \;\middle|\; (\mathbf{x}, y) \in \mathcal{D} \right\}^*.$$

It is likely that most rules in R^- have many active attributes. In order to prevent overfitting and to improve readability, the rules in R^- should be simplified (which may cause a possible information loss). For any $r \in R^-$, a rule s such that $\delta^s = \delta^r$, $A^s \subset A^r$ and such that $\forall i \in A^s$, $\alpha^s_i = \alpha^r_i$, is called a simplification of r.

The SRL algorithm returns a rule-set containing simplifications of the rules in R^-. We say that a rule $\boldsymbol{\alpha} \to \delta$ is compatible with $\mathcal{D}$ if for all $(\mathbf{x}, y) \in \mathcal{D}$

$$\mathbf{x} \geq \boldsymbol{\alpha} \quad \Longrightarrow \quad y \geq \delta.$$

For each rule $r \in R^-$, we search for a subset $A \subset A^r$ (as small as possible) such that the rule

$$\forall i \in A,\ x_i \geq \alpha_i^r \quad \Longrightarrow \quad y \geq \delta^r$$

is compatible with $\mathcal{D}$. For any rule r and any $I \subseteq [n]$, we denote by $r\backslash I$ the rule defined by

$$r\backslash I = (\alpha_1, \dots \alpha_n) \to \delta^r,$$

where, for each $i \in [n]$,

$$\alpha_i = \begin{cases} 0 \text{ if } i \in I, \\ \alpha_i^r \text{ otherwise.} \end{cases}$$

In other words, the rule $r\backslash I$ corresponds to the rule r in which the attributes of I are deactivated.

For each rule r in R^-, we search for a set $I \subseteq [n]$ (as large as possible) such that $r\backslash I$ is compatible with $\mathcal{D}$. We use a greedy approach, where attributes are iteratively added to I. Before adding an attribute i to I, we test whether $r\backslash I \cup \{i\}$ is compatible with $\mathcal{D}$. Since it can happen that $r\backslash I \cup \{i\}$ and $r\backslash I \cup \{j\}$ are compatible with $\mathcal{D}$ while $r\backslash I \cup \{i, j\}$ is not, the order in which attributes are added to I affects the result. We choose this order w.r.t. an evaluation of the discrimination power of each threshold $\alpha_1, \dots, \alpha_n$ of the considered rule. We call discrimination power of a threshold α_i the number of observations for which knowing whether $x_i \geq \alpha_i$ allows to determine whether $y \geq \delta$. It is given by the function $u_i : X_i \times L \to \mathbb{N}$ defined by

$$u_i(\alpha_i, \delta) = \left| \left\{ (\mathbf{x}, y) \in \mathcal{D} \;\middle|\; \left[y \geq \delta \text{ and } x_i \geq \alpha_i \right] \text{ or } \left[y < \delta \text{ and } x_i < \alpha_i \right] \right\} \right|.$$

The process of simplifying rules is formalized by Algorithm 1.

Algorithm 1: Simplification of R^-.

input : A monotonic dataset $\mathcal{D}$ and a rule-set R^- equivalent to $\lambda_{\mathcal{D}}^-$
output: A rule-set compatible with $\mathcal{D}$

```
function SIMPLIFICATION(D, R⁻)
    R ← {}
    for each r ∈ R⁻ do
        I ← {}
        for i ∈ [n] in increasing order of u(α_i, i) do
            if r\(I ∪ {i}) is compatible with D then
                I ← I ∪ {i}
        R ← R ∪ {r\I}
    return R*
```

Running SRL on a dataset where the ranking of values in each set of $X_1, \ldots, X_n$ and L has been reversed is equivalent to running, on the original dataset, a dual method which returns a rule-set containing simplifications of the rejection rules in R^+. We call this dual method SRL^{-1}.

5.2. Handling non-monotonic datasets

Now suppose that $\mathcal{D}$ is not monotonic. This is to say: there exist some pairs of data $\{(\mathbf{x}, y), (\mathbf{x'}, y')\} \subseteq \mathcal{D}$ that are decreasing, i.e., such that

$$\mathbf{x} \leq \mathbf{x'} \quad \text{and} \quad y' < y,$$

while other pairs are increasing. For a decreasing pair, we have

$$\lambda^+_{\mathcal{D}}(\mathbf{x}) < \lambda^-_{\mathcal{D}}(\mathbf{x}) \quad \text{and} \quad \lambda^+_{\mathcal{D}}(\mathbf{x'}) < \lambda^-_{\mathcal{D}}(\mathbf{x'}).$$

In that case, the interval $[\lambda^-_{\mathcal{D}}, \lambda^+_{\mathcal{D}}]$ is empty (no nondecreasing function interpolates $\mathcal{D}$). The lack of monotonicity makes interpolation impossible. Moreover, this defect is often interpreted as the result of noise and, for this reason, several pre-processing methods for restoring monotonicity from data have been proposed.

One approach consists in selecting a subset of the dataset that does not contain any decreasing pair [49]. A more popular approach tries to modify the class of certain observations and is called monotonic relabeling. Its aim is to find a monotonic function $\pounds : D \to L$ called *relabeling function*, which determines a new label for each observation, according to its attribute values. A relabeling function is said to be *optimal* w.r.t. a given loss function ℓ if its empirical error on $\mathcal{D}$ w.r.t. ℓ is minimal.

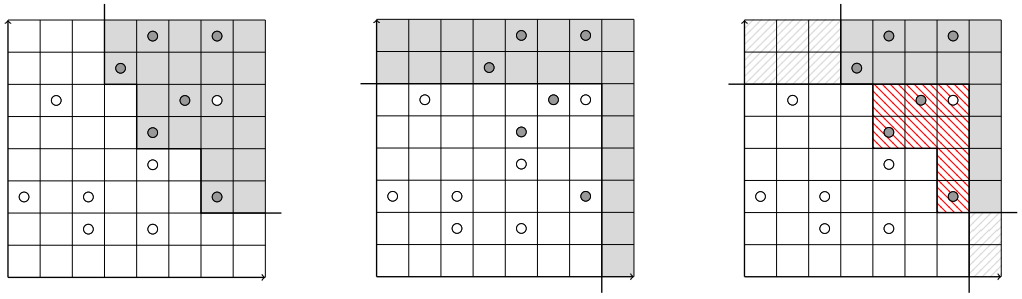


Figure 4: Example of non monotonic data. The two first schemas depict $\lambda^-_{\mathcal{D}}$ and $\lambda^+_{\mathcal{D}}$. In the third schema, we have: $\lambda^-_{\mathcal{D}}(\mathbf{x}) = \lambda^+_{\mathcal{D}}(\mathbf{x}) = 0$ in the white zone, $\lambda^-_{\mathcal{D}}(\mathbf{x}) = \lambda^+_{\mathcal{D}}(\mathbf{x}) = 1$ in the gray zone, $\lambda^-_{\mathcal{D}}(\mathbf{x}) < \lambda^+_{\mathcal{D}}(\mathbf{x})$ in the gray-shaded zone, and $\lambda^+_{\mathcal{D}}(\mathbf{x}) < \lambda^-_{\mathcal{D}}(\mathbf{x})$ in the red-shaded zone.

An optimal relabeling function for any convex loss function (such as ℓ_1, whose associated empirical error is the mean absolute one (MAE)), can be

obtained using an algorithm with complexity in $O(m^3|L|)$ that can be found in [30]. This algorithm will be used in the next sections. Note, however, that numerous relabeling methods exist. An optimal relabeling function in $O(m^3|L|^3)$, for any V-shaped loss is proposed in [49]. Not necessarily optimal re-labelings can be found more efficiently by means of the methods presented in [24, 47].

Several studies [23, 24, 31, 43], relying on empirical or noisy artificial data, show that in certain cases, relabeling the data improves model accuracy. In our case, relabeling $\mathcal{D}$ allows to subsequently perform interpolation, in the sense of [18].

5.3. The rule-set learning algorithm

Relying on the notions and algorithms presented in the previous subsections, we define the rule-set learning algorithm SRL as follows.The first step is a relabeling step that produces a monotonic dataset $\mathcal{M}$. Note that, since we obtain $\mathcal{M}$ via an optimal relabeling of $\mathcal{D}$, the interpolation of $\mathcal{M}$ that is returned by the algorithm is also optimally fit to $\mathcal{D}$ (w.r.t. the MAE).

SRL($\mathcal{D}$)**.** *Let $\mathcal{D}$ be a dataset.*

1. *Relabel $\mathcal{D}$ optimally w.r.t. MAE. We denote by $\mathcal{M}$ the relabeled dataset.*
2. *Express $\lambda^-_{\mathcal{M}}$ by a rule-set R^-:*

$$R^- \leftarrow \big\{\mathbf{x} \to y \;\big|\; (\mathbf{x}, y) \in \mathcal{M}\big\}^*.$$

3. *Simplify the rules of R^-:*

$$R \leftarrow \text{SIMPLIFICATION}(\mathcal{M}, R^-).$$

The result is the rule-set R.

Example 4. *We now illustrate Steps 2 and 3 of SRL. Let $L = \{0, 1\}$ and $X_1 = X_2 = \{0, a, b, 1\}$ such that $0 < a < b < 1$, and let*

$$\mathcal{M} = \big\{\big((0,1),0\big),\ \big((a,b),0\big),\ \big((a,a),0\big),\ \big((1,b),1\big),\ \big((a,1),1\big)\big\}$$

After Step 2, we get the rule-set $R^- = \{r_1, r_2\}$ with

$$r_1 = (1,\ b) \to 1 \quad \textit{and} \quad r_2 = (a,\ 1) \to 1.$$

Note that observations whose class is 0 only produce rules whose right-hand side is $y \geq 0$; those rules do not carry any information.

We now apply Algorithm 1. We compute the discrimination power u_i for observations whose class is 1. For $((1, b), 1)$ we have $u_1(1, 1) = 4$ and $u_2(b, 1) = 3$ and for $((a, 1), 1)$ we have $u_1(a, 1) = 3$ and $u_2(1, 1) = 3$.

We start with the rule $(1, b) \to 1$: since $u_1(1, 1) > u_2(b, 1)$, we first consider the 2nd threshold. This threshold can be decreased to 0, because the rule $(1, 0) \to 1$ is compatible with all observations in $\mathcal{M}$. Then, the first threshold cannot be decreased to 0, because the rule $(0, 0) \to 1$ is incompatible with every observation of class 0. We then consider the rule $(a, 1) \to 1$. No threshold can be decreased to 0, because $(0, 1) \to 1$ is incompatible with $((0, 1), 0)$, and because $(a, 0) \to 1$ is incompatible with $((a, b), 0)$ and $((a, a), 0)$. The function SIMPLIFICATION therefore returns $R = \{r'_1, r_2\}$ with

$$r'_1 = \big[(1,\ 0) \to 1\big] \quad \textit{and} \quad r_2 = \big[(a,\ 1) \to 1\big].$$

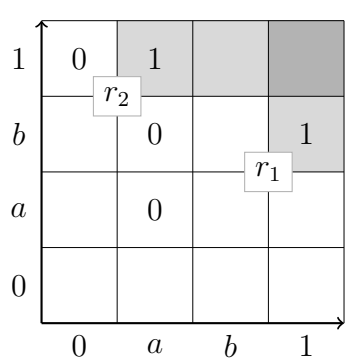


Figure 5: $\mathcal{M}$ and rules of R^-.

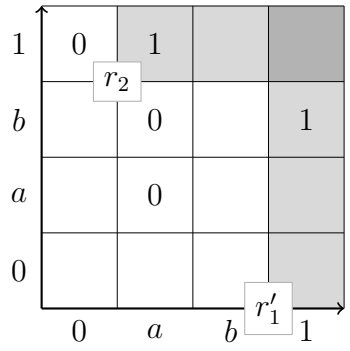


Figure 6: $\mathcal{M}$ and rules returned by $SIMPLIFICATION(\mathcal{M}, R^-)$.

5.4. Practical evaluation of SRL

We evaluate the predictive accuracy of SRL and SRL^{-1} on the datasets listed in Table 2.

ID	Name	# obs.	# att.	# class	Description, source
1	breast-c	286	8	2	Breast-cancer Ljubljana, UCI
2	breast-w	699	9	2	Breast-cancer Wisconsin, UCI
3	car	1296	6	4	Car evaluations
4	CPU	209	6	4	CPU performance evaluation
5	bank-g	1411	16	2	Greek banks evaluation
6	fame	1328	10	5	Firm financial evaluation
7	denbosch	119	8	2	House pricing [22]
8	ERA	1000	4	9	Employees rejection/acceptance [5][1]
9	ESL	488	4	9	Employees selection [5][1]
10	LEV	1000	4	5	Lectures evaluations [5][1]
11	SWD	1000	10	4	Social workers decisions [5][1]
12	windsor	546	10	4	House pricing [2]
13	haberman	306	3	2	Patient survival, UCI
14	balance	325	4	3	Balance state (left, center, right), UCI
15	pima	768	8	2	Pima indians diabetes
16	car	1728	6	4	Car evaluations, UCI
17	auto-MPG	392	7	2	Car fuel consumption, UCI
18	churn	5000	18	2	Client churning prediction
19	german	1000	24	2	German credit data, UCI
20	contraception	1473	9	3	Contraceptive method choice, UCI

Table 2: Description of the datasets.

All datasets that we used are available in `https://github.com/QGBrabant/SUF4OC/`. The first twelve datasets have already been used in [11].[3] Data-sets 13 to 20 are equivalent to those used in [55], with minor modifications: we removed 6 rows in the auto-MPG dataset because of missing attribute values, and we removed one categorical attribute from the churn dataset. When it was not clear whether the class should increase or decrease with a specific attribute value, we used the Spearman's rank correlation coefficient.

In what follows, we evaluate the accuracy of a learning algorithm $\mathscr{A}$ on a dataset $\mathcal{D}$ via the following test.

Evaluation of a learning algorithm$(\mathscr{A}, \mathcal{D})$**.** *Let $\mathcal{D}$ be a dataset and $\mathscr{A}$ be a learning algorithm. The test consists in 10 tenfold cross-validation steps. For each cross-validation, we measure the average accuracy of the result of $\mathscr{A}$ (MER and MAE) obtained on the 10 bins extracted from $\mathcal{D}$.*

The test returns, for MAE and MER, the mean and standard deviation of the results of the 10 tenfold cross-validation steps.

We apply this test to the following 4 algorithms (on each dataset of the table):

[1] `https://www.cs.waikato.ac.nz/ml/weka/data-sets.html`

[3] We acknowledge Roman Slowinski for providing them.

- SRL,
- SRL^{-1},
- a learning method consisting of the two first steps of SRL (which returns $\lambda^{-}_{\mathcal{M}}$),
- a learning method consisting of the two first steps of SRL^{-1} (which returns $\lambda^{+}_{\mathcal{M}}$).

We compare the results to two state-of-the art rule-based monotonic algorithms: VC-DomLEM [7] and RULEM [55] (see their brief description in Appendix 1). Table 3 displays the results obtained by these algorithms and by VC-DomLEM on the first 12 datasets (whose results are reported from [11]).

The results show that the simplification step of SRL has a positive effect on the accuracy. Interestingly, although the error scores of SRL^{-1} and SRL are similar on average, they differ strongly on several datasets.

To verify that Step 3 of SRL and SRL^{-1} improves the accuracy of the rule-set in most cases, we used the Wilcoxon signed-rank test. We compared the accuracy of $\lambda^{-}_{\mathcal{M}}$ (resp. $\lambda^{+}_{\mathcal{M}}$) with that of SRL (resp. SRL^{-1}) on the 13 datasets. When the accuracy measure considered is the misclassification error rate (MER), the tests yielded respectively the p-values 0.003 and 0.005, whereas when the mean absolute error (MAE) is considered, they yielded 0.003 and 0.004. Hence the differences are significant in both cases.

	$\lambda^-_{\mathcal{M}}$		$\lambda^+_{\mathcal{M}}$		SRL		SRL^{-1}		VC-DomLEM	
	MER	MAE	MER	MAE	MER	MAE	MER	MAE	MER	MAE
breast-c	0.253 ± 0.010	0.253 ± 0.010	0.276 ± 0.011	0.276 ± 0.011	0.256 ± 0.011	0.256 ± 0.011	0.278 ± 0.007	0.278 ± 0.007	**0.233** ± 0.003	**0.232** ± 0.003
breast-w	0.12 ± 0.006	0.12 ± 0.006	0.055 ± 0.002	0.055 ± 0.002	0.042 ± 0.003	0.042 ± 0.003	0.04 ± 0.003	0.04 ± 0.003	**0.037** ± 0.002	**0.037** ± 0.002
car	0.038 ± 0.002	0.045 ± 0.002	0.04 ± 0.002	0.045 ± 0.002	**0.023** ± 0.001	**0.026** ± 0.002	0.034 ± 0.003	0.038 ± 0.003	0.028 ± 0.001	0.034 ± 0.001
CPU	0.209 ± 0.008	0.225 ± 0.012	0.22 ± 0.004	0.246 ± 0.005	**0.064** ± 0.008	**0.067** ± 0.008	0.099 ± 0.008	0.102 ± 0.011	0.083 ± 0.014	0.083 ± 0.015
bank-g	0.141 ± 0.001	0.141 ± 0.001	0.311 ± 0.006	0.311 ± 0.006	0.083 ± 0.003	0.083 ± 0.003	0.059 ± 0.001	0.059 ± 0.001	**0.046** ± 0.001	**0.046** ± 0.001
fame	0.547 ± 0.004	0.682 ± 0.004	0.61 ± 0.005	0.803 ± 0.007	0.344 ± 0.004	0.367 ± 0.003	0.343 ± 0.005	0.367 ± 0.005	**0.334** ± 0.005	**0.341** ± 0.002
denbosch	0.306 ± 0.018	0.306 ± 0.018	0.23 ± 0.014	0.23 ± 0.014	0.168 ± 0.013	0.168 ± 0.013	0.146 ± 0.008	0.146 ± 0.008	**0.123** ± 0.010	**0.123** ± 0.010
ERA	**0.735** ± 0.010	**1.26** ± 0.011	0.766 ± 0.007	1.3 ± 0.014	**0.73** ± 0.006	**1.251** ± 0.011	0.76 ± 0.002	1.295 ± 0.009	**0.731** ± 0.004	1.307 ± 0.002
ESL	**0.326** ± 0.012	**0.348** ± 0.012	0.344 ± 0.014	0.388 ± 0.014	**0.33** ± 0.010	**0.355** ± 0.013	0.34 ± 0.006	**0.36** ± 0.007	**0.333** ± 0.013	0.37 ± 0.014
LEV	0.371 ± 0.009	0.402 ± 0.007	0.384 ± 0.005	0.42 ± 0.006	**0.364** ± 0.004	**0.394** ± 0.004	0.384 ± 0.004	0.418 ± 0.004	0.444 ± 0.004	0.481 ± 0.004
SWD	**0.425** ± 0.008	**0.443** ± 0.009	**0.419** ± 0.005	**0.443** ± 0.006	**0.418** ± 0.008	**0.435** ± 0.009	**0.422** ± 0.005	0.445 ± 0.006	0.436 ± 0.005	0.454 ± 0.004
windsor	0.513 ± 0.011	0.589 ± 0.011	0.486 ± 0.011	0.587 ± 0.013	0.486 ± 0.014	0.551 ± 0.018	0.472 ± 0.009	0.526 ± 0.010	**0.454** ± 0.008	**0.502** ± 0.006
mean	0.332	0.401	0.345	0.425	0.276	0.333	0.281	0.339	0.274	0.334

Table 3: Comparison of the errors scores of SRL, unsimplified rule-sets, and VC-DomLEM. On each line, the scores that are at most one standard deviation away from the best score are in bold font.

For the datasets of Table 2 that have been used in [55], we compare the results of SRL and SRL^{-1} to that of three combinations of RULEM with a non-monotonic classification method: Ripper and RULEM, Antminer+ and RULEM, and C4.5 and RULEM. The results of RULEM are reported from [55]. Note that RULEM was evaluated using random split-ups of the data into 2/3 for training and 1/3 for test data. Therefore, the comparison might be slightly biased in favor of SRL, since more training data are used in the tenfold cross-validation.

We used the Wilcoxon signed-rank test corrected with Bonferroni method to investigate whether significant differences in the overall performances of those algorithms could be established. We compared the accuracies of SRL, SRL^{-1}, VC-DomLEM, Ripper+RULEM, Ant+RULEM and C4.5+RULEM on 8 common datasets. We performed a Wilcoxon test for each pair of those methods on the 8 datasets. Each test on a pair of methods yielded a p-

	SRL		SRL^{-1}		Ripper+RULEM		Ant+RULEM		C4.5+RULEM	
	MER	MAE	MER	MAE	MER	MAE	MER	MAE	MER	MAE
breast-c	**0.256**	**0.256**	0.278	0.278	0.289	0.289	0.267	0.267	0.291	0.291
ERA	0.73	1.251	0.76	1.295	0.555	0.78	0.619	0.88	**0.553**	0.78
ESL	0.33	0.355	0.34	0.36	**0.292**	0.32	–	–	0.3	0.3
LEV	**0.364**	0.394	0.384	0.418	0.396	0.45	0.544	0.61	0.391	0.44
SWD	**0.418**	0.435	0.422	0.445	0.438	0.438	0.602	0.602	0.447	0.447
haberman	0.266	0.266	0.288	0.288	**0.25**	**0.25**	0.273	0.273	0.273	0.273
balance-scale	0.187	0.205	**0.186**	0.202	0.187	0.3	0.225	0.37	0.188	0.29
pima	0.255	0.255	0.285	0.285	**0.251**	**0.25**	0.293	0.3	0.253	**0.25**
car	**0.023**	0.026	0.034	0.038	0.051	0.08	–	–	0.03	0.03
auto-mpg	**0.08**	**0.08**	0.084	0.084	0.19	0.19	0.165	0.165	0.175	0.175
churn	0.104	0.104	0.292	0.292	**0.063**	**0.063**	0.112	0.112	–	–
german	0.289	0.289	**0.263**	**0.263**	0.283	0.283	0.32	0.32	–	–
contraception	0.504	0.668	0.492	0.661	**0.48**	0.76	0.576	0.91	–	–
mean	0.293	0.353	0.316	0.378	0.287	0.343	–	–	–	–

Table 4: Comparison of the errors scores of SRL compared to RULEM. On each line, the best score is give in bold font.

value, and we applied a Bonferroni correction to the set of obtained p-values. We did not observe any significant difference ($p < 0.05$) between methods, regardless of the measure considered, i.e., misclassification error rate (MER) or mean absolute error (MAE).

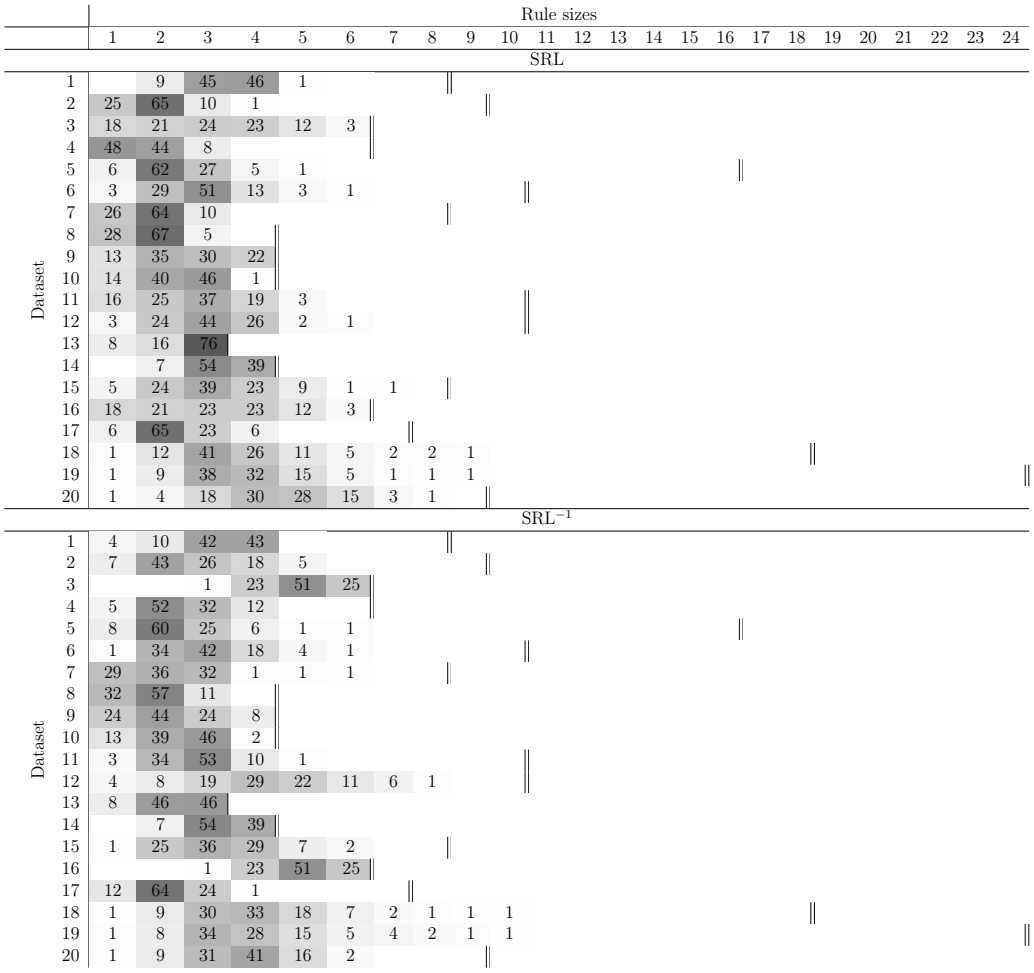


Table 5: Distributions of the length of rules obtained by SRL and SRL^{-1}. Each line is the distribution obtained on one dataset (percentage, rounded-up). The number of attributes in each dataset is indicated by a vertical double line.

The average distributions of rule lengths (number of active attributes) obtained by SRL and SRL^{-1} are given, respectively, in Table 5. These distributions can be compared to those reported in [7]; the great majority of rules generated by SRL, SRL^{-1} and VC-DomLEM are of size from 1 to 6. Such rule lengths are reasonable. However, certain rule-sets contain many rules, as shown in Table 6.

dataset	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
# obs.	286	683	1728	209	1411	1328	119	1000	488	1000	1000	546	306	325	768	1728	392	5000	1000	1473
R^-	14	65	38	41	173	623	23	36	45	39	72	114	13	93	126	38	28	573	430	210
R^+	14	26	39	45	389	691	16	38	48	40	62	104	19	93	192	39	32	2871	206	225
SRL	12	24	33	19	119	449	11	23	39	25	37	87	12	89	104	33	18	393	297	178
SRL^{-1}	13	14	38	23	94	476	7	25	41	25	38	73	15	89	129	38	21	2163	180	188

Table 6: Average number of rules.

Interestingly, it seems that when SRL produces fewer rules than SRL^{-1}, it has a better accuracy, and vice-versa. However, we see that, for several dataset, the obtained rule-sets are very large, and therefore not readable.

6. An algorithm for minimizing SUF-covers

We now want to know the number of SUFs that are necessary to express a rule-set that has been learned from empirical data. We define the following problem.

Minimal SUF-cover problem. *Let R be a rule-set. Find a SUF-cover of R of minimal size.*

Indeed, the size of the smallest SUF-cover of a rule-set R is the minimal size of any SUF-set $\mathbf{S}$ verifying

$$\bigvee \mathbf{S} = f_R.$$

In order to see the difficulty of this problem, consider the following example.

Example 5. *Let $n = 4$, $L = X_1 = X_2 = X_3 = X_4 = \{0, a, b, 1\}$ be a rule-set $R = \{r^1, r^2, r^3, r^4\}$ whose domain is L^4 and whose codomain is L, with*

$$\begin{aligned} r^1 &= (a, a, 0, 0) \rightarrow b, \\ r^2 &= (a, 0, a, 0) \rightarrow b, \\ r^3 &= (a, 0, 0, a) \rightarrow b, \\ r^4 &= (0, b, b, b) \rightarrow 1. \end{aligned}$$

We can check that any subset of R of size 3 is SUF-representable. However, R is not SUF-representable. To see this, we compute $\varphi_1, \varphi_2, \varphi_3, \varphi_4$ and μ such that $S_{\mu,\varphi} = \mathbb{SUF}(R)$. We obtain the following local QNFs:

x	*0*	a	b	*1*
$\varphi_1(x)$	*0*	b	b	*1*
$\varphi_2(x)$	*0*	b	b	*1*
$\varphi_3(x)$	*0*	b	b	*1*
$\varphi_4(x)$	*0*	b	b	*1*

and also the function μ*, whose focal sets are* $\{1,2\}$*,* $\{1,3\}$*,* $\{1,4\}$*,* $\{2,3,4\}$ *with*

$$\mu(\{1,2\}) = \mu(\{1,3\}) = \mu(\{1,4\}) = b \quad \mu(\{2,3,4\}) = 1.$$

If we translate $S_{\mu,\varphi}$ *into a rule-set we get* $R' = \{r^1, r^2, r^3, r^4, r^5\}$ *where*

$$r^5 = (0, a, a, a) \to b.$$

We have $\mathbb{SUF}(R) > f_R$*. The rule* r^5 *in* R' *comes from the fact that* $\mu(\{2,3,4\}) = 1$ *and* $\varphi_2(a) = \varphi_3(a) = \varphi_4(a) = b$*.*

This example illustrates the fact that there exist rule-sets R for which every subset of size $n-1$ is SUF-representable, while $\mathbb{SUF}(R) > f_R$. This fact does not prove that the minimal SUF-cover problem is NP-hard, but we do not know any algorithm for solving it in polynomial time in terms of n and the size of the considered rule-set. We therefore propose an approximate method.

The procedure SUF-INTERPOLATION (Algorithm 2) is a greedy approach that treats a more general problem than the minimal SUF-cover of a rule-set. In what follows, this procedure will be abbreviated by the function S-I. For a given rule-set R^-, represented by a list of rules $\mathtt{l}_{R^-}$, and a function λ^+ such that $f_{R^-} \leq \lambda^+$, S-I($\mathtt{l}_{R^-}, \lambda^+$) returns a partition $\mathbf{P}$ of R^- such that

$$f_{R^-} \leq \bigvee \{\mathbb{SUF}(P) \mid P \in \mathbf{P}\} \leq \lambda^+,$$

by iteratively building a partition $\mathbf{P}$ of R^-, that verifies the constraint

$$\bigvee \{\mathbb{SUF}(P) \mid P \in \mathbf{P}\} \leq \lambda^+,$$

or equivalently, for all $P \in \mathbf{P}$, $\mathbb{SUF}(P) \leq \lambda^+$. The algorithm aims at minimizing the size of $\mathbf{P}$. When $\lambda^+ = f_{R^-}$, SUF-INTERPOLATION returns a SUF-cover of R^-.

The basic idea of the algorithm is based on the following principle: the list $\mathtt{l}_{\boldsymbol{P}}$ contains disjoints subsets of R^- and is initialized only with the empty set. Then we iterate over the set of rules R (**for each** loop): each rule r is added to the largest set $P \in \mathtt{l}_{\boldsymbol{P}}$ such that $\mathbb{SUF}(P) \leq \lambda^+$. If no such set exists, then $\{r\}$ is appended to $\mathtt{l}_{\boldsymbol{P}}$.

Algorithm 2: Greedy search of a minimal partition $\mathbf{P}$ of R^- such that $\bigvee \left\{ \mathbb{SUF}(P) \;\middle|\; P \in \mathbf{P} \right\} \leq \lambda^+$.

input : A list $\mathtt{l}_{R^-}$ containing the rules of the rule-set R^- and a function λ^+ such that $f_{R^-} \leq \lambda^+$.
output: A list $\mathtt{l}_{\boldsymbol{P}}$ which represents a partition of R^-.

```
function SUF-INTERPOLATION(l_{R^-}, λ^+)
    l_P ← empty list
    for each r in l_{R^-} do
        i ← 1
        alone ← true
        while alone and i ≤ size(l_P) do
            if SUF(l_P[i] ∪ {r}) ≤ λ^+ then
                l_P[i] ← l_P[i] ∪ {r}
                sort l_P in decreasing order of cardinality
                alone ← false
            i ← i + 1
        if alone then
            add {r} to l_P
    return l_P
```

We denote the dual counterpart of S-I by S-I^{-1}. The function S-I^{-1} takes as input a set R^+ of rules of the dual form. Moreover, S-I^{-1} applied to R^+ and λ^- returns a SUF-cover of R^+.

Proposition 6. *Let R^- be a rule-set and λ^+ be a function such that $f_{R^-} \leq \lambda^+$. There necessarily exists a rule-set R such that*

$$f_{R^-} \leq f_R \leq \lambda^+$$

and a list $\mathtt{l}_R$ containing all elements of R such that the SUF-set $\boldsymbol{S}$ defined by

$$\boldsymbol{S} = \left\{ \mathbb{SUF}(P) \mid P \in \text{S-I}(\mathtt{l}_R, \lambda^+) \right\}$$

is one of the smallest SUF-sets verifying

$$f_{R^-} \leq \bigvee \boldsymbol{S} \leq \lambda^+. \tag{4}$$

Sketch of proof. We consider one of the smallest SUF-set $\mathbf{S}$ verifying (4). This SUF-set can be translated into an equivalent rule-set R. The intuition

behind the proof is that, provided the elements of R are in the right order, the function S-I returns a SUF-cover that corresponds to $\mathbf{S}$. In fact, we prove the following weaker statement: it is possible to order the elements of R in such a way that the returned partition yields a SUF set that verifies (4) and that is of the same size as $\mathbf{S}$. The complete proof is given in the Appendix.

Such a rule-set R is called an *optimal argument* of the function S-I. The proof of Proposition 6 is given in the Appendix. When $f_{R^-} = \lambda^+$, the optimal argument is necessarily R^-. We therefore have the following corollaries.

Corollary 7. *Let R be a rule-set. There necessarily exists a list $\mathtt{l}_R$ containing all elements of R, such that $\bigvee\{\mathbb{SUF}(P) \mid P \in \text{S-I}(\mathtt{l}_R, f_R)\}$ is a minimal SUF-cover of R.*

However, if $f_{R^-} \neq \lambda^+$, the rule-set R^- is not necessarily an optimal argument of S-I. This comes from the fact that, for two rule-sets R and R' such that $f_R < f_{R'}$, it is possible that $\mathbb{SUF}(R') < \mathbb{SUF}(R)$.

We now rely on S-I to estimate the number of SUFs that are necessary to express rule-sets of various sizes. To this end, we learn rule-sets from the 12 datasets of Table 2, via SRL. Then, for each obtained rule-set, we perform the following test.

SUF-cover test(R)**.** *For a given rule-set R, we repeat 1000 times the following steps:*

1. *Shuffle $\mathtt{l}_R$.*
2. *Compute* S-I$(\mathtt{l}_R, f_R)$ *(see Algorithm 2).*

The result of this test is composed of the mean, the minimum and the standard deviation of $|\text{S-I}(\mathtt{l}_R, f_R)|$*, obtained during these 1000 loops.*

dataset	1	2	3	4	5	6	7	8	9	10	11	12
Number of observations	286	683	1728	209	1411	1328	119	1000	488	1000	1000	546
Size of R^-	13	38	36	23	178	557	14	25	36	26	38	101
Avg. nb. of SUFs	6	11	10	6	126	214	8	8	11	9	7	36
std.	0	0.78	0.75	0.59	0	2.92	0.45	0.73	0.81	0.81	0.85	1.07
min.	6	9	9	6	126	205	8	7	10	8	6	34
Size of R^+	13	23	38	33	297	618	6	28	46	23	48	83
Avg. nb. of SUFs	5	12	16	14	149	206	4	9	11	7	11	34
std.	0.5	0.72	0.97	0.95	0.16	2.83	0	0.76	0.92	0.69	0.82	1.2
min.	5	11	15	12	149	198	4	8	9	6	10	31

Table 7: Results of the SUF-cover test, for the original and reversed order.

Since our algorithm is not optimal, we cannot guarantee that the sizes of SUF-sets reported in Table 7 are representative of the sizes of the optimal

solutions. However, it is reasonable to assume that they are good approximations, relying on Corollary 7, and on the fact that obtained standard deviations are small.

7. Rules vs. sets of SUFs

In this section we try to compare the extraction of rule sets from monotonic data, and the extraction of sets of SUFS, considering synergies between both, using algorithms SRL and S-I.

7.1. Interpolation of a dataset by a SUF-set

In practice, it is not very interesting to translate a rule-set into a SUF-set. Indeed, the rule-set is not a perfect model of the data; therefore, it is useless to require that the SUF-set is perfectly equivalent to a given rule-set. In this subsection, we take a different approach: we define a method for interpolating a (monotonic) dataset by a SUF-set. This method takes a greedy approach to minimize the number of SUFs in the interpolating SUF-set.

A SUF interpolation of a dataset $\mathcal{M}$ is a SUF-set $\mathbf{S}$ verifying

$$\lambda^-_{\mathcal{M}} \leq \bigvee \mathbf{S} \leq \lambda^+_{\mathcal{M}}.$$

In order to estimate the number of SUFs needed to interpolate a relabeled dataset, we start with the following problems.

Minimal SUF interpolation. *Let $\mathcal{M}$ be a monotonic dataset. Return one of the smallest SUF interpolation of $\mathcal{M}$.*

Again, we rely on function S-I. Let R^- be a rule-set such that $f_{R^-} = \lambda^-_{\mathcal{M}}$. Then S-I($\mathtt{l}_{R^-}, \lambda^+_{\mathcal{M}}$) gives a (non-necessarily optimal) solution to the minimal SUF-interpolation problem.

7.2. Experiment 1

We designed a first experiment, based on the datasets of the previous section. This time, for each dataset, we ran the following test.

SUF interpolation test($\mathcal{D}$).

1. *Relabel $\mathcal{D}$ optimally w.r.t. MAE. We denote by $\mathcal{M}$ the monotonic relabeled dataset.*
2. *Create a list $\mathtt{l}_{R^-}$ representing the rule-set R^- such that $f_{R^-} = \lambda^-_{\mathcal{M}}$.*
3. *Repeat 1000 times:*

(a) Shuffle $\mathtt{l}_{R^-}$.

(b) Compute S-I$(\mathtt{l}_{R^-}, \lambda^+_{\mathcal{M}})$ *(see Algorithm 2).*

The result of this test is composed of the size of $\mathtt{l}_{R^-}$*, and of the mean, minimum and std of* $|\text{S-I}(\mathtt{l}_{R^-}, \lambda^+_{\mathcal{M}})|$*, over the 1000 loops.*

dataset	1	2	3	4	5	6	7	8	9	10	11	12
Number of observations	286	683	1728	209	1411	1328	119	1000	488	1000	1000	546
Size of R^-	17	71	36	50	456	811	35	37	46	49	74	128
Avg. nb. of SUFs	7	9	8	6	19	46	8	3	9	6	5	18
std.	0.42	1.09	0.64	0.71	1.39	1.73	0.67	0.53	0.69	0.59	0.63	1.03
min.	7	7	8	5	15	42	8	3	8	5	4	15
Size of R^+	16	54	40	56	610	818	35	41	54	38	61	149
Avg. nb. of SUFs	6	5	16	3	17	50	4	5	11	8	5	20
std	0.31	0.61	0.82	0.54	1.24	2.02	0.5	0.59	0.85	0.68	0.83	1.32
min.	6	4	15	3	14	44	4	4	9	7	4	16

Table 8: Results of the first SUF-interpolation test for each dataset, for original and reversed order.

In the previous section, we relied on the corollary of Proposition 6 to state that the results of the SUF-cover test is a reasonable approximation of the number of SUFs required to translate the rule-sets that were learned via SRL. We can not make such a statement in the case of SUF-interpolation, which is a more general problem, unless R^- is an optimal argument of S-I.

Consequently, the average number of SUFs in Table 8 should be thought of as upper bounds on the number of SUFs required for SUF-interpolation.

7.2.1. Experiment 2

We do not know any algorithm that returns an optimal argument of function S-I in reasonable time. However, note that most rules in R^- have a lot of active attributes. Thus, it follows from the definition of $\mathbb{SUF}$ that most focal sets of S-I$(\mathtt{l}_{R^-}, \lambda^+_{\mathcal{M}})$ have a great cardinality. Thus, using rules with fewer active attributes could be a way to decrease the number of SUFs used for interpolation.

We ran a test that is similar to the previous one, but where R^- is replaced by the result of SRL.

2nd SUF-interpolation test$(\mathcal{D})$**.**

1. *Relabel* $\mathcal{D}$ *optimally w.r.t. MAE. We denote by* $\mathcal{M}$ *the monotonic relabeled dataset.*

2. $R \leftarrow SRL(\mathcal{M})$*; create a list* $\mathtt{l}_R$ *that contains all elements of* R*.*

3. *Repeat 1000 times:*

(a) Shuffle $\mathtt{l}_R$ *and* $\mathtt{l}_{R^{-1}}$.

(b) Compute S-I($\mathtt{l}_R, \lambda^+_{\mathcal{M}}$*) (see Algorithm 2).*

The result of the test consists in the sizes of R, and of the mean minimum and std of |S-I($\mathtt{l}_R, \lambda^+_{\mathcal{M}}$*)|, over 1000 loops.*

dataset	1	2	3	4	5	6	7	8	9	10	11	12
Size of $\mathcal{D}$	286	683	1728	209	1411	1328	119	1000	488	1000	1000	546
Size of SRL($\mathcal{M}$)	13	26	36	21	134	493	15	26	44	25	36	96
Avg. nb. of SUFs	3	4	10	4	16	60	6	5	11	5	4	17
std	0.51	0.35	0.55	0.6	0.88	1.98	0	0.57	0.58	0.52	0.48	0.95
min.	3	4	10	4	13	54	6	5	11	4	4	14
Size of SRL^{-1}($\mathcal{M}$)	14	15	38	25	104	532	8	25	47	27	42	69
Avg. nb. of SUFs	4	4	15	8	16	70	4	4	10	6	4	17
std	0	0.6	0.56	0.48	0.99	2.13	0	0.55	0.8	0.56	0.53	0.9
min.	4	4	15	8	15	64	4	4	9	6	3	16

Table 9: Results of the second test of SUF-interpolation for each dataset, with original and reversed order.

We see that there is no overall improvement compared to the previous test. However, using rule-sets obtained via SRL had a clear influence (positive or negative, depending on the dataset) on the sizes of SUF-sets. Therefore, it is probable that knowing the optimal arguments of S-I would enable a significant decrease in the number of SUFs used for interpolation.

Improvements towards more compact representations seem feasible, even if the minimal SUF-interpolation problem is NP-hard [18]:

- It could be useful to heuristically search for optimal arguments of the SUF-INTERPOLATION procedure.
- A minimally interpolating SUF-set could also be searched via a meta-heuristic method. This solution has already been proposed for learning Sugeno integrals (using genetic algorithms [57] and particle swarms [56]).

7.3. SUF-sets for monotonic classification

In the previous section we computed the SUF-interpolations of relabeled datasets. We will now evaluate the predictive accuracy of the obtained classifiers. We propose the following learning algorithm.

RL-SUF($\mathcal{D}$)**.** *Let* $\mathcal{D}$ *be a given dataset.*

1. *Relabel* $\mathcal{D}$ *optimally w.r.t. MAE. We denote by* $\mathcal{M}$ *the monotonic relabeled dataset.*

2. $R \leftarrow \mathrm{SRL}(\mathcal{M})$.

3. Return S-I$(\mathtt{1}_R, \lambda^+_{\mathcal{M}})$.

This algorithm searches for a SUF-set $\mathbf{S}$ such that

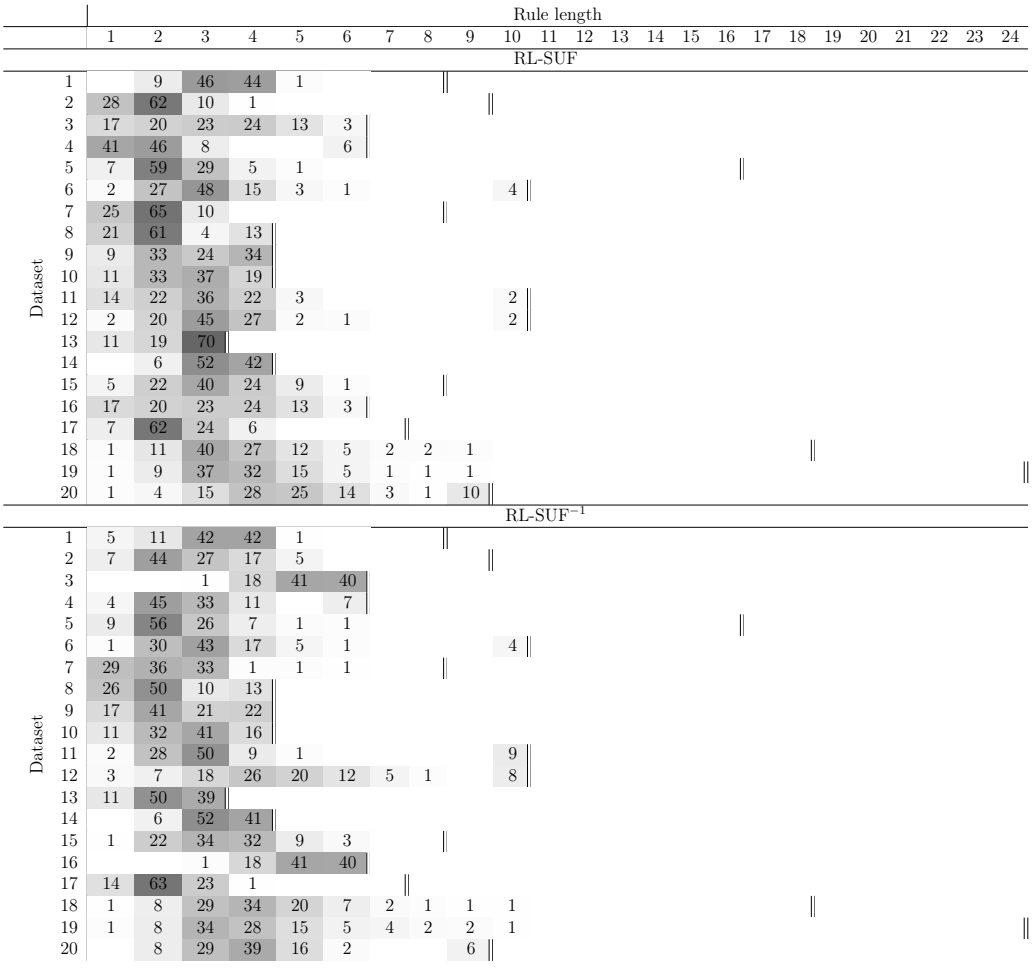
$$\lambda^-_{\mathcal{M}} \leq \bigvee \mathbf{S} \leq \lambda^+_{\mathcal{M}}.$$

and whose cardinality is as small as possible.

We compare the accuracy of RL-SUF and of its dual counterpart RL-SUF^{-1} to that of SRL and SRL^{-1}. We evaluate each method according to the test procedure defined in Subsection 5.4, on each dataset of Table 2. Table 10 displays the result of this test for each algorithm and each dataset.

	SRL		SRL^{-1}		RL-SUF		$RL\text{-}SUF^{-1}$	
	MER	MAE	MER	MAE	MER	MAE	MER	MAE
breast-c	**0.256** ± 0.011	**0.256** ± 0.011	0.278 ± 0.007	0.278 ± 0.007	**0.257** ± 0.014	**0.257** ± 0.014	0.276 ± 0.013	0.276 ± 0.013
breast-w	**0.042** ± 0.003	**0.042** ± 0.003	**0.04** ± 0.003	**0.04** ± 0.003	**0.039** ± 0.003	**0.039** ± 0.003	**0.04** ± 0.002	**0.04** ± 0.002
car	**0.023** ± 0.001	**0.026** ± 0.002	0.034 ± 0.003	0.038 ± 0.003	**0.022** ± 0.002	**0.026** ± 0.002	**0.024** ± 0.002	0.028 ± 0.002
CPU	**0.064** ± 0.008	**0.067** ± 0.008	0.099 ± 0.008	0.102 ± 0.011	**0.069** ± 0.009	**0.074** ± 0.009	0.1 ± 0.009	0.106 ± 0.010
bank-g	0.083 ± 0.003	0.083 ± 0.003	**0.059** ± 0.001	**0.059** ± 0.001	0.076 ± 0.003	0.076 ± 0.003	**0.056** ± 0.003	**0.056** ± 0.003
fame	0.344 ± 0.004	0.367 ± 0.003	0.343 ± 0.005	0.367 ± 0.005	**0.328** ± 0.005	**0.354** ± 0.004	**0.333** ± 0.006	0.362 ± 0.006
denbosch	0.168 ± 0.013	0.168 ± 0.013	**0.146** ± 0.008	**0.146** ± 0.008	0.163 ± 0.014	0.163 ± 0.014	**0.146** ± 0.012	**0.146** ± 0.012
ERA	**0.73** ± 0.006	**1.251** ± 0.011	0.76 ± 0.002	1.295 ± 0.009	**0.732** ± 0.006	**1.253** ± 0.011	0.759 ± 0.009	1.295 ± 0.015
ESL	**0.33** ± 0.010	**0.355** ± 0.013	0.34 ± 0.006	**0.36** ± 0.007	**0.324** ± 0.012	**0.35** ± 0.012	0.34 ± 0.009	**0.361** ± 0.009
LEV	**0.364** ± 0.004	**0.394** ± 0.004	0.384 ± 0.004	0.418 ± 0.004	**0.366** ± 0.005	**0.398** ± 0.006	0.384 ± 0.007	0.419 ± 0.008
SWD	**0.418** ± 0.008	**0.435** ± 0.009	**0.422** ± 0.005	0.445 ± 0.006	0.427 ± 0.009	0.445 ± 0.010	**0.417** ± 0.007	**0.441** ± 0.008
windsor	0.486 ± 0.014	0.551 ± 0.018	**0.472** ± 0.009	**0.526** ± 0.010	**0.472** ± 0.006	0.534 ± 0.009	**0.47** ± 0.012	**0.519** ± 0.010
haberman	**0.266** ± 0.074	**0.266** ± 0.074	**0.278** ± 0.007	**0.278** ± 0.007	**0.263** ± 0.070	**0.263** ± 0.070	**0.288** ± 0.076	**0.288** ± 0.076
balance-scale	0.187 ± 0.006	0.205 ± 0.006	**0.04** ± 0.003	**0.04** ± 0.003	0.206 ± 0.010	0.227 ± 0.010	0.211 ± 0.012	0.233 ± 0.012
pima	0.255 ± 0.054	0.255 ± 0.060	**0.034** ± 0.003	**0.038** ± 0.003	0.26 ± 0.046	0.26 ± 0.055	0.279 ± 0.043	0.279 ± 0.049
car	**0.023** ± 0.002	**0.026** ± 0.004	0.099 ± 0.008	0.102 ± 0.011	**0.023** ± 0.005	**0.026** ± 0.006	**0.025** ± 0.008	**0.029** ± 0.013
auto-mpg	0.08 ± 0.047	0.08 ± 0.047	**0.059** ± 0.001	**0.059** ± 0.001	0.075 ± 0.048	0.075 ± 0.048	0.078 ± 0.044	0.078 ± 0.044
churn	**0.104** ± 0.004	**0.104** ± 0.004	0.343 ± 0.005	0.367 ± 0.005	**0.108** ± 0.006	**0.108** ± 0.006	0.235 ± 0.007	0.235 ± 0.007
german	0.289 ± 0.011	0.289 ± 0.013	**0.146** ± 0.008	**0.146** ± 0.008	0.285 ± 0.010	0.285 ± 0.013	0.264 ± 0.011	0.264 ± 0.013
contraception	0.504 ± 0.001	0.668 ± 0.002	0.76 ± 0.002	1.295 ± 0.009	0.511 ± 0.001	0.67 ± 0.001	**0.49** ± 0.002	**0.652** ± 0.003
mean	0.251	0.294	0.257	0.32	0.25	0.294	0.261	0.305

Table 10: Comparison of errors obtained by SRL and RL-SUF.



RL-SUF

Dataset	Rule length 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1		9	46	44	1																			
2	28	62	10	1																				
3	17	20	23	24	13	3																		
4	41	46	8			6																		
5	7	59	29	5	1																			
6	2	27	48	15	3	1				4														
7	25	65	10																					
8	21	61	4	13																				
9	9	33	24	34																				
10	11	33	37	19																				
11	14	22	36	22	3					2														
12	2	20	45	27	2	1				2														
13	11	19	70																					
14		6	52	42																				
15	5	22	40	24	9	1																		
16	17	20	23	24	13	3																		
17	7	62	24	6																				
18	1	11	40	27	12	5	2	2	1															
19	1	9	37	32	15	5	1	1	1															
20	1	4	15	28	25	14	3	1	10															

RL-SUF^{-1}

Dataset	Rule length 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	5	11	42	42	1																			
2	7	44	27	17	5																			
3			1	18	41	40																		
4	4	45	33	11		7																		
5	9	56	26	7	1	1																		
6	1	30	43	17	5	1				4														
7	29	36	33	1	1	1																		
8	26	50	10	13																				
9	17	41	21	22																				
10	11	32	41	16																				
11	2	28	50	9	1					9														
12	3	7	18	26	20	12	5	1		8														
13	11	50	39																					
14		6	52	41																				
15	1	22	34	32	9	3																		
16			1	18	41	40																		
17	14	63	23	1																				
18	1	8	29	34	20	7	2	1	1	1														
19	1	8	34	28	15	5	4	2	2	1														
20		8	29	39	16	2			6															

Table 11: Length distributions of the rules obtained by the translation of the result of RL-SUF and RL-SUF^{-1}.

We see that the scores of RL-SUF are similar to those of SRL, and that the scores of RL-SUF^{-1} are similar to those of SRL^{-1}. Using once again the Wilcoxon test to compare the accuracy scores of SRL (resp. SRL^1) to that of RL-SUF (resp. RL-SUF^{-1}), we observed no significant difference, as the test yielded the p-value 0.75 (resp. 0.75) for MER and 0.71 (resp. 0.88) for MAE. In other words, the step of learning the SUF-set from the rule-set does not affect the accuracy much; accuracy is mainly determined by the rule-set learning step.

The SUF-sets resulting from RL-SUF and RL-SUF^{-1} can be translated into rules via the function $\mathbb{S}$-$\mathbb{SET}$. The average rule-size distributions of the rule-sets obtained in this way are given in Tables 11.

Those distributions are similar to those obtained by SRL and SRL^{-1}, respectively. However, both variants of RL-SUF produce rules of maximal size on datasets 6, 11 and 12. The presence of those rules is due to the fact that any capacity $\mu : 2^{[n]} \to L$ verifies $\mu([n]) = 1$.

7.4. SUF-sets pruning

We now want to see if the SUF-sets resulting from RL-SUF can be reduced while keeping a similar accuracy. Let $\mathbf{S}$ be a SUF-set resulting of RL-SUF, and $\mathcal{M}$ be the monotonic relabeled dataset from which $\mathbf{S}$ has been learned. The function $\bigvee \mathbf{S}$ is an interpolation of $\mathcal{M}$. Thus, removing SUFs from $\mathbf{S}$ increases the empirical error of $\mathbf{S}$ on $\mathcal{M}$. This increase of error will be used to regulate the pruning of $\mathbf{S}$. We define the *accuracy* of $\mathbf{S}$ on $\mathcal{M}$ as the ratio of observations that are correctly classified:

$$\text{accuracy}(\bigvee \mathbf{S}, \mathcal{M}) = 1 - \text{MER}(\bigvee \mathbf{S}, \mathcal{M}),$$

where MER stands for Misclassification Error Rate. We consider $\rho \in [0, 1]$, which is the minimal accuracy ratio that has to be preserved while removing a SUF. The function PRUNING (see Algorithm 3) prunes the SUF-set $\mathbf{S}$ with a greedy approach, according to the parameter ρ.

The higher the value of ρ, the more the accuracy on training data is favored over pruning. When $\rho > 1$, the SUF-set is left unchanged. By varying the value of ρ, we obtain various trade-offs between the cardinality of the resulting SUF-set and its accuracy on the training data.

Algorithm 3: Pruning of a SUF-set resulting of RL-SUF.

input : A SUF-set $\mathbf{S}$, a dataset $\mathcal{M}$, the ratio $\rho \in [0, 1]$.
output: A subset of $\mathbf{S}$.

```
function PRUNING(S, M, ρ)
    stop ← false
    while stop = false do
        stop ← true
        for S ∈ S do
            if accuracy(⋁(S\S), M) ≥ ρ * accuracy(⋁ S, M) then
                S ← S\S
                stop ← false
    return S
```

For each dataset of the table, we ran the following test procedure.

Test: pruning of SUF-set obtained via RL-SUF($\mathcal{D}$). *Let $\mathcal{D}$ be a given dataset. The test consists in 10 tenfold cross-validation steps (a total of 100 times).*

1. *For each step,*
   (a) *divide $\mathcal{D}$ in $\mathcal{D}_{\text{app}}$ and $\mathcal{D}_{\text{eval}}$,*
   (b) $\boldsymbol{S} \leftarrow \text{RL-SUF}(\mathcal{D}_{\text{app}})$,
   (c) *for each value of ρ in* $\{0.95, 0.96, \ldots, 1\}$
      i. $\boldsymbol{S'} \leftarrow \text{PRUNING}(\boldsymbol{S}, \mathcal{M}, \rho)$,
      ii. measure the MER of $\boldsymbol{S'}$ on $\mathcal{D}_{\text{eval}}$.

The result of the test is, for each value of ρ, the average number of SUFs in the SUF-set and the average MER, on the 100 cross-validation steps.

Figures 7 and 8 display the results of the pruning test. The number of SUFs that are required for reaching the best accuracy varies greatly over datasets. Sometimes, pruning allows to reduce the cardinality of the SUF-set without affecting accuracy (this is the case, e.g., for breast-c, breast-w, ESL and windsor); however, in many datasets, the number of SUFs that are required for approaching the good accuracy is clearly too large to achieve readability.

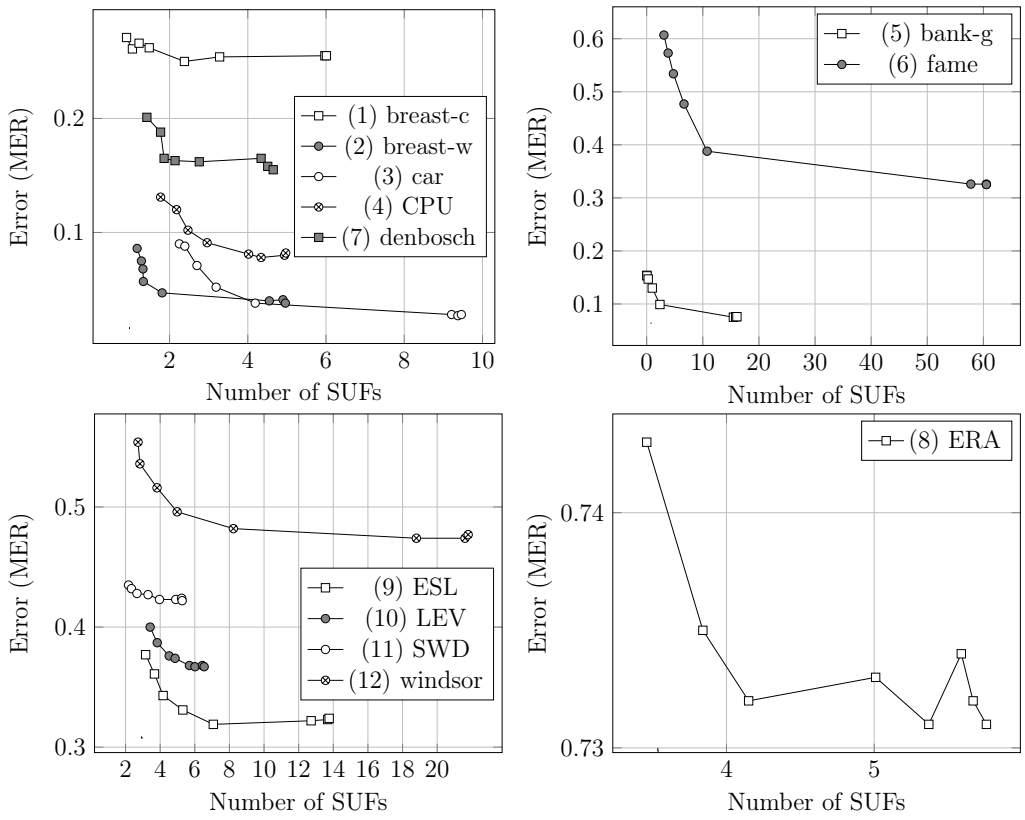


Figure 7: Averaged Error (MER) and number of SUFs obtained by pruned results of RL-SUF. Each curve corresponds to a dataset.

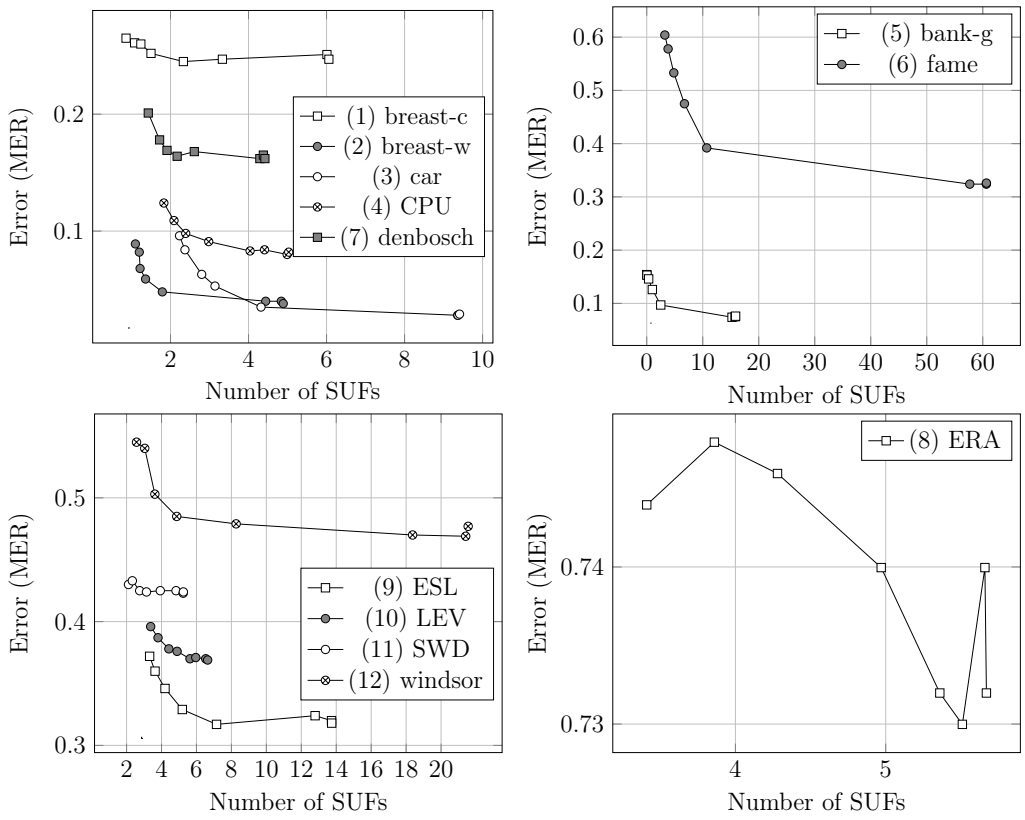


Figure 8: Averaged Error (MER) and number of SUFs obtained by pruned results of RL-SUF^{-1}. Each curve corresponds to a dataset.

The pruning methods show that on some datasets, it is possible to reduce the number of SUFs of the SUF-set without impacting greatly the accuracy on test data.

8. Conclusion

In this paper, we considered the problem of monotonic classification from the standpoint of extracting rules and finding an optimal classification function in terms of generalized Sugeno integrals (SUFs). We studied the equivalence between the two representations from a theoretical point of view. SUF sets can be seen as an alternative expression of rule-sets. Any SUF can be translated into a rule set, but a rule set generally corresponds to several SUFS. SUF sets might provide advantages in terms of conciseness, by expressing a great number of rules in a few SUFs.

Moreover, we proposed a non-parametric rule set learning algorithm (SRL), and a non-parametric SUF set learning algorithm (RL-SUF). We finally studied whether rule sets can be expressed in a more compact and readable form using SUFs. Our non-parametric rule-set learning algorithm is competitive with VC-DomLEM in terms of errors and rule sizes. We then used SRL for studying the number of SUFs required for the following tasks. We gave an empirical estimation of the number of SUFs required to express rule-sets learned from empirical data (via SRL). The results suggest that such a translation does not significantly improve conciseness. We also gave an upper bound on the number of SUFs needed, for each dataset considered. Finally, the accuracy of the learned SUF-sets does not vary significantly under our methods.

The results thus obtained are competitive with those obtained by efficient algorithms such as VC-DomLEM. As we have shown in our experiments, neither RULEM, VC-DomLEM nor SRL does better than the two other methods on all datasets. An advantage of RULEM is that it produces rule-sets that are clearly smaller than those of SRL (the sizes of the rule-sets of VC-DomLEM are unknown). The sizes of the SUF-sets obtained by RL-SUF are sometimes too large to favor interpretability.

In the case of qualitative data it seems that the capabilities of learning compact representations of large datasets are more limited than in the numerical setting. For instance, in some cases, sets of SUFs are not always more interpretable than rule sets, since many SUFs are needed to attain an accuracy comparable to that of the corresponding rule-set. As future work, we will explore a hybrid approaches that combines SUFs through decision trees. In particular, we believe that this hybrid methodology will provide more concise and interpretable representations. However, SUFs can stand as

the bridge between rule-based representations and numerical models based on weighted average and generalizations thereof. Indeed, Sugeno integral can be seen as a qualitative rendering of Choquet integral [27] as well as a closed form representation of rule sets. In practice, looking for a cognitively palatable counterpart of a dataset in terms of rules makes sense only if the underlying function is simple enough. It is in such situations that SUFs can be interpreted in terms of simple rules. If one is only interested in the classification task, without concern about interpretability, the SUF format may be of interest, since it is slightly more compact than a rule set, and this format makes predictions easier to compute than using the whole rule set.

Acknowledgements. This work is supported by ANR-11-LABX-0040-CIMI (Centre International de Mathématiques et d'Informatique) within the program ANR-11-IDEX-0002-02, project ISIPA.

Appendix

1. Overview of monotonic classification methods

In this paper, we defined monotonic classification as the task of learning a monotonic classifier; however, this term sometimes refers to a less constrained task, where the monotonicity of the classifier is seen as a measure to optimize rather than a hard constraint. Moreover, some monotonic classification methods allow to learn classifiers that are monotonic on a subset of attributes and unconstrained on the others. For a broader overview of monotonic classification methods, see, e.g. [12].

Several monotonic classification approaches have been designed by adapting classical approaches, such as k-nearest neighbors [29], Bayesian networks [1, 32, 40], or neural networks [50, 58].

Other approaches use models that rely on the notion of monotonicity. This is the case of isotonic separation [14], which relies on a notion of distance to predict the classes of observations, and of the probabilistic approaches OSDL [45] and MOCA [4, 3]. Ordinal logistic regression [46] is a monotonic classification method that generalizes the binary classification method of logistic regression. It relies on the idea that the classes can be viewed as intervals on the domain of an unobserved variable, which is can be approximated as a linear combination of the attributes. Then, the *ordinal Choquistic regression* [53] is a monotonic classification method that generalizes the ordinal logistic regression by relying on a Choquet integral instead of a linear function. Choquet integrals are aggregation functions on the interval $[0, 1]$, and are popular in the domain of preference modelling because they generalize the idea of weighted mean, by associating a weight to each

subset of attributes, and thus allow to model synergy or redundancy between attributes.

Several approaches for learning monotonic binary decision trees have also been proposed. The method of [48] requires $\mathcal{D}$ to be monotonic and returns a function from $[\lambda_{\mathcal{D}}^-, \lambda_{\mathcal{D}}^+]$. The algorithm REMT [41] relies on a generalization of Shannon's entropy that takes into account the monotonicity hypothesis, which is called *rank entropy*. When $\mathcal{D}$ is monotonic, the binary tree learned by REMT is also monotonic. Finally, the ICT [54] algorithm makes use of a variant of CART [9], followed by a step of pruning that guaranties the monotonicity of the tree. Decision trees constitute locally interpretable models; the global interpretability of a tree depends on the number of nodes it contains. However, note that small change in the data can have a strong impact on the resulting tree. Thus, a global interpretation of the tree is potentially misleading (see [39], Section 9.2).

The first rule-set learning algorithm was OLM [6]. Since then, several other approaches were proposed, such as RULEM [55], which consists in post-processing the results of nonmonotonic rule-set learning algorithms in order to obtain monotonic rule-sets, or the ant-based method of [10]. A noteworthy method is VC-DomLEM [7, 11], which is based on the DRSA [37]. This algorithm aims at fitting the learning data while favoring short rules. It depends on several hyperparameters. The authors show, via a comparison with other methods over 12 datasets, that VC-DomLEM is competitive with several state of the art approaches. This is why we use VC-DomLEM as a reference for evaluating our own learning algorithms.

Finally, note that some methods like MORE [25] or LPRules [42] consist in learning a linear combination of simple functions specified by selection or rejection rules. The class predictions of these methods tend to be more accurate than those of the "classical" rule-sets presented in the previous section. However, although the rules that are used in these methods are of the same form as those of classical rule-sets, they are combined in such a way that it is impossible to justify each class prediction by a single rule.

2. Proof of Proposition 1

Proposition 1. For any rule-set R, R^* is the smallest element of $\mathrm{Eq}(R)$.

Proof. Let R be a rule-set. The implication relation on rules can be seen as a partial order. We have $f_r \leq f_s$ if and only if $s \Rightarrow r$. The set R^* contains all maximal elements of R (w.r.t. the order $\Rightarrow$). Consequently, for any $r \in R$ there is $s \in R^*$ such that $f_r \leq f_s$, and therefore we have

$$f_{R^*} = \bigvee_{r \in R^*} f_r = \bigvee_{r \in R} f_r = f_R.$$

We now show, by contradiction, that R^* is included in any $R' \in \mathrm{Eq}(R)$. Assume that there is $R' \in \mathrm{Eq}(R)$ such that $R^* \not\subseteq R'$. Then there exists $\boldsymbol{\alpha}^1 \to \delta$ belonging to R^* and not to R'. By definition, R^* contains no rule $\boldsymbol{\alpha}^1 \to \delta'$, such that $\delta' > \delta$. Thus we have

$$f_{R^*}(\boldsymbol{\alpha}^1) = f_{\boldsymbol{\alpha}^1 \to \delta}(\boldsymbol{\alpha}^1) = \delta.$$

Since R^* and R' belongs to $\mathrm{Eq}(R)$, $f_{R^*} = f_{R'}$, and thus

$$f_{R'}(\boldsymbol{\alpha}^1) = \delta.$$

Consequently, R' contains a rule $\boldsymbol{\alpha}^2 \to \delta$ such that $\boldsymbol{\alpha}^2 \leq \boldsymbol{\alpha}^1$. If $\boldsymbol{\alpha}^2 = \boldsymbol{\alpha}^1$, the assumption that $(\boldsymbol{\alpha}^1 \to \delta) \notin R'$ is contradicted. Therefore, we should have $\boldsymbol{\alpha}^2 < \boldsymbol{\alpha}^1$. Since $f_{R'}$ is non-decreasing, we have $f_{R'}(\boldsymbol{\alpha}^2) \leq f_{R'}(\boldsymbol{\alpha}^1) = \delta$, and since R' contains $\boldsymbol{\alpha}^2 \to \delta$, we have $f_{R'}(\boldsymbol{\alpha}^2) \geq \delta$. Therefore

$$f_{R^*}(\boldsymbol{\alpha}^2) = f_{R'}(\boldsymbol{\alpha}^2) = \delta,$$

and this means that R^* contains a rule $\boldsymbol{\alpha}^3 \to \delta$, with $\boldsymbol{\alpha}^3 \leq \boldsymbol{\alpha}^2 < \boldsymbol{\alpha}^1$. This contradicts the definition of R^*, because $[\boldsymbol{\alpha}^3 \to \delta] \Rightarrow [\boldsymbol{\alpha}^1 \to \delta]$. □

3. Proof of Proposition 3 and Corollary 4

Proposition 3. For any SUF S we have

$$S = f_{\$\text{-}\$\mathbb{ET}(S)} = \$\mathbb{UF}(\$\text{-}\$\mathbb{ET}(S)).$$

Proof. Let $S_{\mu,\varphi}$ be a SUF. We will show that the following assertions always hold.

A. $S_{\mu,\varphi} \leq \$\mathbb{UF}(\$\text{-}\$\mathbb{ET}(S_{\mu,\varphi}))$ and $S_{\mu,\varphi} \leq f_{\$\text{-}\$\mathbb{ET}(S_{\mu,\varphi})}$

B. $S_{\mu,\varphi} \geq f_{\$\text{-}\$\mathbb{ET}(S_{\mu,\varphi})}$

C. $S_{\mu,\varphi} \geq \$\mathbb{UF}(\$\text{-}\$\mathbb{ET}(S_{\mu,\varphi}))$.

Proof of A. Let $\mathbf{x} \in \mathbf{X}$ and $y \in L$ such that $S_{\mu,\varphi}(\mathbf{x}) = y$. There necessarily is a focal set $F \in \mathcal{F}(\mu)$ such that

$$\mu(F) \wedge \bigwedge_{i \in F} \varphi_i(x_i) = y.$$

Consequently,

$$\mu(F) \geq y \quad \text{and} \quad \forall i \in F,\ \varphi_i(x_i) \geq y.$$

Let r be the rule such that

$$A^r = F, \quad \delta^r = y \quad \text{and} \quad \forall i \in A^r,\ \alpha_i^r = \bigwedge\{a_i \in X_i \mid \varphi_i(a_i) \geq y\}.$$

Note that for all $i \in A^r$ we have $\alpha_i^r \leq x_i$ because $\varphi_i(x_i) \geq y$. Therefore $f_r(\mathbf{x}) \geq y$. From the definition of $\mathbb{S}$-$\mathbb{SET}$, it follows that there exists $s \in \mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})$ such that $s \Rightarrow r$. Thus, we have $f_{\mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})}(\mathbf{x}) \geq y$.

Let $S_{\mu',\varphi'}$ be the SUF given by $S_{\mu',\varphi'} = \mathbb{SUF}(\mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi}))$. From the definition of $\mathbb{SUF}$, it follows that

$$\mu'(A^s) \geq y \quad \text{and} \quad \forall i \in A^s,\ \varphi_i'(x_i) \geq y,$$

and thus $\mathbb{SUF}(\mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})) \geq y$.

Proof of B. Let $\mathbf{x} \in \mathbf{X}$ and $y \in L$ be such that $f_{\mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})}(\mathbf{x}) = y$. There exists a rule $r \in \mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})$ such that

$$\forall i \in A^r,\ x_i \geq \alpha_i^r \quad \text{and} \quad \delta^r = y.$$

From the definition of $\mathbb{S}$-$\mathbb{SET}$ it follows that $\mu(A^r) \geq \delta^r$. Moreover, for all $i \in A^r$,

$$\alpha_i^r = \bigwedge\{a_i \in X_i \mid \varphi_i(a_i) \geq \delta^r\},$$

and thus $\varphi_i(x_i) \geq \varphi_i(\alpha_i^r) \geq \delta^r$. Therefore, we have

$$\mu(A^r) \wedge \bigwedge_{i \in A^r} \varphi_i(x_i) \quad \geq \quad \delta^r \quad \geq \quad y,$$

and thus $S_{\mu,\varphi}(\mathbf{x}) \geq y$.

Proof of C. Let $S_{\mu',\varphi'}$ be the SUF given by $\mathbb{SUF}(\mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi}))$. Let $\mathbf{x} \in \mathbf{X}$ and $y \in L$ be such that $S_{\mu',\varphi'}(\mathbf{x}) = y$. If $y = 0$ then $S_{\mu,\varphi}(\mathbf{x}) \geq y$. Therefore we assume that $y > 0$.

There necessarily exists $F \in \mathcal{F}(\mu')$ such that

$$\mu'(F) \wedge \bigwedge_{i \in F} \varphi_i'(x_i) = y > 0 \tag{5}$$

Consequently, $\mu'(F) \geq y$ and $\varphi_i'(x_i) \geq y$ for all $i \in F$. Since $y > 0$, we have $\mu'(F) > 0$ and thus $F \neq \emptyset$. We start by showing that $\mu(F) \geq y$.

- If $F = [n]$, then $\mu(F) = 1 \geq y$.

- If $F \neq [n]$, then from the definition of $\mathbb{SUF}$ we have

$$\mu'(F) = \bigvee\{\delta^r \mid r \in \mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi}),\ A^r \subseteq F\},$$

and thus $\mu'(F) \geq y$ implies that there exists a rule $r \in \mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})$ such that $A^r \subseteq F$ and $\delta^r \geq y$. Consequently $\mu(A^r) \geq \delta^r \geq y$.

For each $i \in F$, we show that $\varphi_i(x_i) > y$. We have $\varphi_i'(x_i) > 0$ (see (5)) and thus $x_i > 0$.

- If $x_i = 1$, then $\varphi_i(x_i) = 1 \geq y$.
- If $x_i < 1$, then it follows from the definition of $\mathbb{SUF}$ that

$$\varphi_i'(x_i) = \bigvee\{\delta^r \mid r \in \mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi}),\ 0 < \alpha_i^r \leq x_i\}$$

consequently $\varphi'(x_i) \geq y$ implies that there is a rule $r \in \mathbb{S}\text{-}\mathbb{SET}(S_{\mu,\varphi})$ such that $\delta^r = y$ and $0 < \alpha_i^r \leq x_i$. Due to the definition of $\mathbb{S}\text{-}\mathbb{SET}$, it holds:

$$\alpha_i^r = \bigwedge\{a_i \in X_i \mid \varphi_i(a_i) \geq \delta^r\}.$$

Consequently $\varphi_i(x_i) \geq \varphi_i(\alpha_i^r) \geq \delta^r = y$.

Finally we have

$$\mu(A^r) \wedge \bigwedge_{i \in A^r} \varphi_i(x_i) \geq y,$$

and thus $S_{\mu,\varphi}(\mathbf{x}) \geq y$. □

The proof of the corresponding result for rejection rules can be obtained in an analogous manner.

Corollary 4. *A set of selection (resp. rejection) rules R is SUF-representable if and only if $\mathbb{SUF}(R^*) = f_R$ (resp. $\mathbb{SUF}(R^*) = f^R$).*

Proof. Let R be a rule-set. If R is not SUF-representable, then $\mathbb{SUF}(R^*) \neq f_R$. If R is SUF-representable, we call S the SUF such that $S = f_R$. From Proposition 3 it follows that

$$f_{\mathbb{S}\text{-}\mathbb{SET}(S)} = \mathbb{SUF}(\mathbb{S}\text{-}\mathbb{SET}(S)).$$

And since $S = f_R$ we have

$$R^* = \mathbb{S}\text{-}\mathbb{SET}(S)^* = \mathbb{S}\text{-}\mathbb{SET}(S),$$

and thus

$$f_R = f_{R^*} = \mathbb{SUF}(R^*).$$

In the case where R is a reject-set, the equality $\mathbb{SUF}(R^*) = f_R$ can be proven in an analogous manner. □

4. Proof of Proposition 6

Proposition 6. *Let R^- be a rule-set and λ^+ be a function, such that $f_{R^-} \leq \lambda^+$. There necessarily exists a rule-set R such that*

$$f_{R^-} \leq f_R \leq \lambda^+$$

and a list $\mathtt{l}_R$ containing the elements of R such that the set of SUFs $\boldsymbol{S}$ defined by

$$\boldsymbol{S} = \left\{ \mathbb{SUF}(P) \mid P \in \text{S-I}(\mathtt{l}_R, \lambda^+) \right\}$$

is one of the smallest SUF-sets verifying $f_{R^-} \leq \bigvee \boldsymbol{S} \leq \lambda^+$.

Proof. During this proof, lists will be denoted as tuples.

Let $\mathbf{S} = \{S_1, \ldots, S_d\}$ be a SUF-set of minimal size verifying $f_{R^-} \leq \bigvee \mathbf{S} \leq \lambda^+$. The size of $\mathbf{S}$ is denoted by d. We will show that there exists a rule-set R such that S-I$(\mathtt{l}_R, \lambda^+)$ returns a SUF-set of size d and

$$f_{R^-} \leq \bigvee \left\{ \mathbb{SUF}(P) \mid P \in \text{S-I}(\mathtt{l}_R, \lambda^+) \right\} \leq \lambda^+.$$

Assume that there is a list $\mathtt{l}_{\mathbf{P}} = (P_1, \ldots, P_d)$ such that

$$\bigcup_{i=1}^{d} P_i = \bigcup_{i=1}^{d} \mathbb{S}\text{-}\mathbb{SET}(S_i), \tag{6}$$

$$\forall i \in [d], \quad \mathbb{SUF}(P_i) \leq \lambda^+, \tag{7}$$

$$|P_1| \geq \cdots \geq |P_d|, \tag{8}$$

$$\forall i, j \in [d] \text{ such that } |P_i| \geq |P_j|, \ \forall r \in P_j : \quad \mathbb{SUF}(P_i \cup \{r\}) \not\leq \lambda^+. \tag{9}$$

It follows from (6) and (7) that

$$f_{R^-} \leq \mathbf{S} \leq \bigvee \left\{ \mathbb{SUF}(P_i) \;\middle|\; P_i \in \mathtt{l}_{\mathbf{P}} \right\} \leq \lambda^+.$$

By relying on (8) and (9), we can define a list $\mathtt{l}_R$ such that S-I$(\mathtt{l}_R, \lambda^+) = \mathtt{l}_{\mathbf{P}}$. For this, it is sufficient that:

- $\mathtt{l}_R$ contains every element of $\bigcup_{i=1}^{d} P_i$,
- $\mathtt{l}_R$ is sorted in such a way that, for each $i \in \{2, \ldots, d\}$, all rules of P_{i-1} appear before the rules of P_i.

We still have to show that there indeed exists a list verifying (6), (7), (8) and (9). Let $\mathtt{l_Q}$ be the list defined by

$$\mathtt{l_Q} = (\mathbb{S}\text{-}\mathbb{SET}(S_1), \ldots, \mathbb{S}\text{-}\mathbb{SET}(S_d)).$$

This list necessarily verifies (6) and (7). We define the function g on lists of rule-sets of length d by

$$g(R_1, \ldots, R_d) = \begin{cases} (R_1, \ldots, R_d) \text{ if } (R_1, \ldots, R_d) \text{ verifies (9)}, \\ (R'_1, \ldots, R'_d) \text{ otherwise,} \end{cases}$$

where $R'_1, \ldots, R'_d$ are defined in the following manner. Firstly, let R_i and R_j be the smallest rule-sets of $(R_1, \ldots, R_d)$ such that $|R_i| \geq |R_j|$ and such that $\exists r \in R_j$, $\mathbb{SUF}(R_i \cup \{r\}) \leq \lambda^+$. Secondly,

$$\forall k \in [d], \quad R'_k = \begin{cases} R_k \cup \{r\} \text{ if } k = i, \\ R_k \backslash \{r\} \text{ if } k = j, \\ R_k \text{ otherwise.} \end{cases}$$

We will now use g^k to denote k successive applications of g. Formally: let $g^1 = g$ and, for any integer $k > 1$, let $g^k = g \circ g^{k-1}$, where $\circ$ denote the function composition. It is quite easy to see that:

- If $(R_1, \ldots, R_d)$ verifies (6) and (7), then $g(R_1, \ldots, R_d)$ verifies (6) and (7).
- For any list of rule-sets $(R_1, \ldots, R_d)$ there exists a positive integer k such that $g^k(R_1, \ldots, R_d)$ verifies (9).

Consequently, there is a positive integer k such that $g^k(\mathtt{l_Q})$ verifies (6), (7), and (9). Sorting $g^k(\mathtt{l_Q})$ yields a list that also verifies (8). □

References

[1] E.E. Altendorf, A.C. Restificar, and T.G. Dietterich. Learning from Sparse Data by Exploiting Monotonicity Constraints. In *Proc. 21st Conf. on Uncertainty in Artificial Intelligence*, UAI'05, pages 18–26, Arlington, Virginia, 2005. AUAI Press.

[2] P.M. Anglin and R. Gençay. Semiparametric estimation of a hedonic price function. *J. Applied Econometrics*, 11(6):633–648, 1996.

[3] N. Barile. *Studies in Learning Monotonic Models from Data.* PhD Thesis, Dutch Research School for Information and Knowledge Systems, 2014.

[4] N. Barile and A.J. Feelders. Nonparametric Monotone Classification with MOCA. In *Proc. 8th IEEE Int. Conf. on Data Mining*, pages 731–736, 2008.

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

[6] A. Ben-David, L. Sterling, and Y-H. Pao. Learning and classification of monotonic ordinal concepts. *Computational Intelligence*, 5(1):45–49, 1989.

[7] J. Blaszczyński, R. Slowiński, and M. Szelag. VC-DomLEM: Rule induction algorithm for variable consistency rough set approaches. Technical Report RA-07/09, Poznań University of Technology, 2009.

[8] Q. Brabant, M. Couceiro, D. Dubois, H. Prade, and A. Rico. Extracting decision rules from qualitative data via sugeno utility functionals. In J. Medina et al., editor, *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations (Proc. 17th IPMU Conf.), Part I*, volume 853 of *Communications in Computer and Information Science*, pages 253–265. Springer, 2018.

[9] L. Breiman. *Classification and Regression Trees.* Routledge, 2017.

[10] J. Brookhouse and F.E.B. Otero. Monotonicity in Ant Colony Classification Algorithms. In M. et al Dorigo, editor, *Swarm Intelligence*, LNCS, pages 137–148. Springer, 2016.

[11] J. Błaszczyński, R. Słowiński, and M. Szeląg. Sequential covering rule induction algorithm for variable consistency rough set approaches. *Information Sciences*, 181(5):987–1002, 2011.

[12] J-R Cano, P.A. Gutiérrez, B. Krawczyk, M. Woźniak, and S. García. Monotonic classification: an overview on algorithms, performance measures and data sets. *arXiv:1811.07155 [cs]*, 2018.

[13] J.R. Cano, P.A. Gutiérrez, B. Krawczyk, M. Wozniak, and S. García. Monotonic classification: An overview on algorithms, performance measures and data sets. *Neurocomputing*, 341:168–182, 2019.

[14] R. Chandrasekaran, Y.U. Ryu, V.S. Jacob, and S. Hong. Isotonic Separation. *INFORMS Journal on Computing*, 17(4):462–474, 2005.

[15] M. Couceiro, J. Devillet, and J.-L. Marichal. Characterizations of idempotent discrete uninorms. *Fuzzy Sets and Systems*, 334:60 – 72, 2018.

[16] M. Couceiro, D. Dubois, H. Prade, and A. Rico. Enhancing the Expressive Power of Sugeno Integrals for Qualitative Data Analysis. In *Advances in Fuzzy Logic and Technology (Proc. EUSFLAT 2017)*, Advances in Intelligent Systems and Computing, pages 534–547. Springer, Cham, 2017.

[17] M. Couceiro, D. Dubois, H. Prade, and T. Waldhauser. Decision-Making with Sugeno integrals - bridging the gap between multicriteria evaluation and decision under uncertainty. *Order*, 33:517–535, 2016.

[18] M. Couceiro, M. Maróti, T. Waldhauser, and L. Zádori. Computing version spaces in the qualitative approach to multicriteria decision aid. *Int. J. Found. Comput. Sci.*, 30(2):333–353, 2019.

[19] M. Couceiro and T. Waldhauser. Sugeno Utility Functions I: Axiomatizations. In V. Torra, Y. Narukawa, and M. Daumas, editors, *Modeling Decisions for Artificial Intelligence*, LNCS, pages 79–90. Springer Berlin Heidelberg, 2010.

[20] M. Couceiro and T. Waldhauser. Sugeno Utility Functions II: Factorizations. In V. Torra, Y. Narukawa, and M. Daumas, editors, *Modeling Decisions for Artificial Intelligence*, LNCS, pages 91–103. Springer Berlin Heidelberg, 2010.

[21] M. Couceiro and T. Waldhauser. Pseudo-polynomial functions over finite distributive lattices. *Fuzzy Sets and Systems*, 239:21–34, 2014.

[22] H. Daniels and B. Kamp. Application of MLP Networks to Bond Rating and House Pricing. *Neural Computing & Applications*, 8(3):226–234, 1999.

[23] H. Daniels and M.V. Velikova. Derivation of monotone decision models from non-monotone data. Technical Report 2003-30, Tilburg School of Economics and Management, 2003.

[24] H. Daniels and M.V. Velikova. Derivation of monotone decision models from noisy data. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 36(5):705–710, 2006.

[25] K. Dembczyński, W. Kotłowski., and R. Słowiński. Learning Rule Ensembles for Ordinal Classification with Monotonicity Constraints. *Fundamenta Informaticae*, 94(2):163–178, 2009.

[26] D. Dubois, C. Durrieu, H. Prade, A. Rico, and Y. Ferro. Extracting Decision Rules from Qualitative Data Using Sugeno Integral: A Case-Study. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 9161 of *LNCS*, pages 14–24. Springer, 2015.

[27] D. Dubois and H. Fargier. Making Discrete Sugeno Integrals More Discriminant. *International Journal of Approximate Reasoning*, 50(6):880–898, June 2009.

[28] D. Dubois, J-L. Marichal, H. Prade, M. Roubens, and R. Sabbadin. The use of the discrete sugeno integral in decision-making: a survery. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(5):539–561, 2001.

[29] W. Duivesteijn and A.J. Feelders. Nearest Neighbour Classification with Monotonicity Constraints. In *Machine Learning and Knowledge Discovery in Databases*, volume 5211 of *LNCS*, pages 301–316. Springer, Berlin, Heidelberg, 2008.

[30] A.J. Feelders. Monotone relabeling in ordinal classification. In *Proc. IEEE Int. Conf. on Data Mining*, pages 803–808, 2010.

[31] A.J. Feelders and T. Kolkman. Exploiting monotonicity constraints to reduce label noise: An experimental evaluation. In *Proc. Int. Joint Conf. on Neural Networks (IJCNN)*, pages 2148–2155, 2016.

[32] A.J. Feelders and L.C. Van der Gaag. Learning Bayesian network parameters under order constraints. *Int. J. Approximate Reasoning*, 42(1):37–53, 2006.

[33] J. C. Fodor. Smooth associative operations on finite ordinal scales. *IEEE Trans. Fuzzy Systems*, 8(6):791–795, 2000.

[34] M. Grabisch. The application of fuzzy integrals in multicriteria decision making. *Eur. J. of Operational Research*, 89(3):445–456, 1996.

[35] M. Grabisch and C. Labreuche. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1):247–286, 2010.

[36] M. Grabisch, J.-L. Marichal, R. Mesiar, and E. Pap. *Aggregation Functions*, volume 127 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, New York, NY, 2009.

[37] S. Greco, B. Matarazzo, and R. Slowinski. Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research*, 129(1):1–47, February 2001.

[38] 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. *Eur. J. of Operational Research*, 158(2):271–292, 2004.

[39] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer-Verlag, New York, 2 edition, 2009.

[40] E.M. Helsper, L.C. Van Der Gaag, A.J. Feelders, W.L.A. Loeffen, P.L. Geenen, and A.R.W. Elbers. Bringing order into Bayesian-network construction. In *Proc. 3rd Int. Conf. on Knowledge Capture (K-CAP '05)*, pages 121–128, New York, NY, 2005. ACM.

[41] Q. Hu, X. Che, L. Zhang, D. Zhang, M. Guo, and D. Yu. Rank entropy-based decision trees for monotonic classification. *IEEE Trans. on Knowledge and Data Engineering*, 24(11):2052–2064, 2012.

[42] W. Kotlowski. *Statistical Approach to Ordinal Classification with Monotonicity Constraints*. PhD thesis, Poznań University of Technology, 2008.

[43] W. Kotlowski and R. Slowinski. On Nonparametric Ordinal Classification with Monotonicity Constraints. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2576–2589, 2013.

[44] L. Li and H-T. Lin. Ordinal Regression by Extended Binary Classification. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 865–872. MIT Press, 2007.

[45] S. Lievens, B. De Baets, and K. Cao-Van. A probabilistic framework for the design of instance-based supervised ranking algorithms in an ordinal setting. *Annals of Operations Research*, 163(1):115–142, 2008.

[46] P. McCullagh. Regression Models for Ordinal Data. *J. of the Royal Statistical Society. Series B (Methodological)*, 42(2):109–142, 1980.

[47] W. Pijls and R. Potharst. Repairing non-monotone ordinal data sets by changing class labels. Technical report, Econometric Institute, Erasmus University Rotterdam, 2014.

[48] R. Potharst and J.C. Bioch. Decision trees for ordinal classification. *Intelligent Data Analysis*, 4(2):97–111, 2000.

[49] M. Rademaker, B. De Baets, and H. De Meyer. Loss optimal monotone relabeling of noisy multi-criteria data sets. *Information Sciences*, 179(24):4089–4096, 2009.

[50] J. Sill. Monotonic networks. In M. I. Jordan et al., editor, *Advances in Neural Information Processing Systems (Proc. 1997 NIPS Conference)*, pages 661–667. The MIT Press, 1998.

[51] R. Slowinski, 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):1000–1035, 2002.

[52] M. Sugeno. *Theory of fuzzy integrals and its applications*. Tokyo Institute of Technology, 1974.

[53] A.F. Tehrani and E. Hüllermeier. Ordinal Choquistic regression. In J. Montero, G. Pasi, and D. Ciucci, editors, *Proc. 8th Conf. European Society for Fuzzy Logic and Technology (EUSFLAT-13), Milano*, pages 802–809. Atlantis Press, 2013.

[54] R. Van De Kamp, A.J. Feelders, and N. Barile. Isotonic Classification Trees. In *Advances in Intelligent Data Analysis VIII*, volume 5772 of *LNCS*, pages 405–416. Springer, Berlin, Heidelberg, 2009.

[55] W. Verbeke, D. Martens, and B. Baesens. RULEM: A novel heuristic rule learning approach for ordinal classification with monotonicity constraints. *Applied Soft Computing*, 60:858–873, 2017.

[56] X-Z. Wang, Y-L. He, L-C. Dong, and H-Y. Zhao. Particle swarm optimization for determining fuzzy measures from data. *Information Sciences*, 181(19):4230–4252, 2011.

[57] Z. Wang, K-S. Leung, and J. Wang. Determining nonnegative monotone set functions based on Sugeno's integral: an application of genetic algorithms. *Fuzzy Sets and Systems*, 112(1):155–164, 2000.

[58] H. Zhu, E.C.C. Tsang, X.Z. Wang, and R. Aamir Raza Ashfaq. Monotonic classification extreme learning machine. *Neurocomputing*, 225:205–213, 2017.