Information Sciences 424 (2018) 104–117

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Partially monotonic decision trees

Shenglei Pei[a,b], Qinghua Hu[a,*]

[a] School of Computer Science and Technology, Tianjin University, Tianjin 300350, China
[b] School of Physics and Electronic Information Engineering, Qinghai University for Nationalities, Xining 810007, China

ARTICLE INFO

Article history:
Received 8 December 2016
Revised 19 June 2017
Accepted 3 October 2017
Available online 3 October 2017

Keywords:
Decision tree
Partially monotonic
Rank inconsistency
Monotonic directions

ABSTRACT

In multicriteria decision tasks, certain features are linearly ordered according to the decision and are called criteria, whereas others, called regular attributes, are not. In practice, regular attributes and criteria coexist in most classification tasks. In this paper, we propose a rank-inconsistent rate that distinguishes attributes from criteria. Furthermore, it represents the directions of the monotonic relationships between criteria and decisions. We design a partially monotonic decision tree algorithm to extract decision rules for partially monotonic classification tasks. Experimental results show that the proposed algorithm is effective and efficient.

1. Introduction

Monotonic classification is used in many applications, such as multicriteria decision making [3,37,39,40], credit rating [11,14,17], the customer satisfaction analysis [21], and the house pricing problem [42]. There is a wide range of monotonic problems, in which the decision values must increase with the feature values. For instance, the price of a house typically increases with the house's size. Smoking increases the probability of lung cancer. In these classification tasks, monotonic classification functions guarantee monotonicity between attributes and decisions. That is, objects with better feature values should not be assigned worse decision values.

Researchers in monotonic classification have proposed several methods with which to learn and extract decision rules for generating decision models. These studies can be roughly classified into those that involve constructing a theoretical framework with monotonicity constraints, and those that consist of a model-based approach. The dominance-based rough set approach (DRSA) builds a formal framework in which to discuss monotonic classification. This model was proposed by Greco et al. [19,20,22] and applied to monotonic tasks. DRSA was subsequently extensively discussed by other researchers. Błaszczyński et al. [5,6] proposed the variable- consistency dominance-based rough set approach (VC-DRSA) based on extended lower approximation. Soon afterward, they designed a transformation method for discovering partially monotonic relationships. Moreover, the VC-DomLEM algorithm was proposed based on this method [7,8]. This method was also applied to DRSA and VC-DRSA for nonmonotonic decision tasks as well [4]. However, this method is complicated, as it clones attributes with unknown monotonicity. In 2015, Wang et al. [43] proposed a refined method to improve the performance of VC-DomLEM algorithms. The preprocessing method in this approach does not clone all attributes with unknown monotonicity to improve performance. The monotonic *k*-nearest neighbors (kNN) algorithm was proposed in [15]. It relabels training data and predicts class labels by utilizing modified nearest-neighbor rules. Some hybrid approaches with monotonicity constraints have also been proposed to extract rules in recent years [10,30].

* Corresponding author.
E-mail addresses: peishenglei@qhmu.edu.cn (S. Pei), huqinghua@tju.edu.cn (Q. Hu).

https://doi.org/10.1016/j.ins.2017.10.006
0020-0255/

Table 1
An example of overdue evaluation.

Objects	Housing loans	Amount	Usage rates	History overdue	Overdue rank
$\mathbf{x}_1$	Yes	10	60	No	0
$\mathbf{x}_2$	Yes	10	80	No	1
$\mathbf{x}_3$	No	5	100	No	1
$\mathbf{x}_4$	No	5	60	Yes	2
$\mathbf{x}_5$	Yes	1	80	Yes	2

The model-based approach is an alternative for solving classification tasks with monotonicity constraints. Support vector machines (SVMs) and other methods based on kernel learning have recently been applied to monotonic classification problems as well. In 2005, Pelckmans et al. [36] designed monotonic kernel regression based on Least Squares SVM (LS-SVM) regression. This algorithm mainly solves ordinal regression problems. Doumpos et al. [14] proposed a linear SVM with an $L_1 - \infty$ norm for credit risk evaluation, which implied the monotonicity of credit risk. The monotonic SVM model is built by adding monotonicity constraints to the conventional SVM model. An extension of the monotonic SVM was studied in [32]. In [12], monotonic neural networks were proposed for partially monotonic classification. This method contains all available features needed to achieve superior performance over standard neural networks. Decision tree models have been widely applied to monotonic classification tasks. In 1995, Ben-David extended ID3 algorithms to monotonic classification [2]. Since then, monotonic decision trees have been designed to solve monotonic problems of various kinds. A postpruning classification tree was proposed by Feelders [16]. This algorithm prunes the parent node of a nonmonotone leaf, but yields only slightly better performance than standard algorithms. Moreover, Hu et al. designed a monotonic decision tree algorithm based on rank mutual information (RMI) in 2012 [25] called REMT. Inspired by this idea in [26], Marsala and Petturiti [35] defined rank Gini impurity (RGI) and proposed a binary decision tree classifier (RGMT). Some recent ensemble algorithms have been adapted for monotonic classification [18,23]. These methods have solved some important problems in monotonic classification tasks. In 2015, Qian et al. proposed the fusing monotonic decision tree [38]. Moreover, they discussed attribute reduction and fusion principles. However, most algorithms assume that all features are monotonic with the decision.

Certain features that have a monotonic relationship with the decision are naturally ordered [12,28]; however, some features may be qualitative, such that their relationship with the decision is nonmonotonic. These features are effective in improving classifier performance. In general, features that are monotonically related to decisions are called criteria, whereas other features are referred to as attributes. We provide the example of evaluating whether a credit card is likely to be overdue in Table 1. There are four features and a decision here. Housing loans and overdue history are qualitative variables; the values of these attributes and the decision are not linearly ordered. These features affect overdue ranking. A cardholder with no housing loan is generally likely to be overdue, as is one with a history of being overdue. A lower credit may lead to higher overdue rank; that is, it monotonically decreases with the decision. This makes it necessary to consider the monotonic direction. In Table 1, we see that usage rates monotonically increase with overdue rank.

To evaluate problems such as these, we design partially monotonic decision trees that can improve the handling of attributes and criteria. In this paper, we propose a rank inconsistency rate (RIR) based on rank mutual information to determine whether features are monotonic. Moreover, RIR can capture monotonic directions. Partially monotonic decision trees (PMDT) generate the best split using mutual information (MI) and rank mutual information (RMI). The algorithm handles not only monotonic features, but also considers nonmonotonic features.

The remainder of this paper is organized as follows: In Section 2, we review some related concepts and formulations. In Section 3, we introduce the discriminant feature monotonicity method. In Section 4, we illustrate the construction of PMDTs. In Section 5, we present some experiments used to evaluate the effectiveness of the proposed algorithms. Finally, our conclusions and directions for future work are given in Section 6.

2. Related work

In this section, we review the concepts related to monotonic classification and measuring feature importance in the context of partially monotonic problems.

2.1. Partial monotonicity constraints

Let $\langle U, A, D\rangle$ be a decision table, where $U = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is a set of objects, $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ is a set of attributes to describe the objects, and D is a finite ordinal set of decisions. The value domain of D is $\{d_1, d_2, \ldots, d_k\}$.

Definition 1. Given a set of objects U, $\forall \mathbf{x} \in U$ and $B \subseteq A$, where $B = \{\mathbf{a}_1, \ldots, \mathbf{a}_{m'}\}$. Let $\mathbf{a}_k(\mathbf{x})$ be the attribute value of sample $\mathbf{x}$ on $\mathbf{a}_k$. The ordinal relations between samples in terms of attribute $\mathbf{a}_k$ or D is denoted by $\preceq$. Thus, the partial ordering $\preccurlyeq$ on U is defined as

$$\mathbf{x}_i \preccurlyeq_{\mathbf{B}} \mathbf{x}_j \iff \mathbf{a}_k(\mathbf{x}_i) \leqslant \mathbf{a}_k(\mathbf{x}_j), \text{ for } k = 1, \ldots, m'. \tag{1}$$

For $\forall \mathbf{a}_k \in B$, $\mathbf{a}_k(\mathbf{x}_i) \geq \mathbf{a}_k(\mathbf{x}_j)$. Then, we say that $\mathbf{x}_i$ is not worse than $\mathbf{x}_j$ regarding B, and denote this by $\mathbf{x}_i \succcurlyeq_B \mathbf{x}_j$. Similarly, if $\mathbf{a}_k(\mathbf{x}_i) \leq \mathbf{a}_k(\mathbf{x}_j)$, it is denoted by $\mathbf{x}_i \preccurlyeq_B \mathbf{x}_j$.

Given that $\mathbf{x}_i \in U$, $B \subseteq A$, we associate $\mathbf{x}_i$ with the following sets:

$$[\mathbf{x}_i]_B^{\preccurlyeq} = \{\mathbf{x}_j \in U \mid \mathbf{x}_j \preccurlyeq_B \mathbf{x}_i\},$$
$$[\mathbf{x}_i]_D^{\preccurlyeq} = \{\mathbf{x}_j \in U \mid \mathbf{x}_j \preccurlyeq_D \mathbf{x}_i\}. \tag{2}$$

Definition 2. Given a set of objects U, $\forall \mathbf{x} \in U$, $B \subseteq A$, where $B = \{\mathbf{a}_1, \ldots, \mathbf{a}_{m'}\}$. Thus, partial monotonicity constraints on U are defined as

$$\begin{aligned} &\mathbf{x}_i \preccurlyeq_B \mathbf{x}_j \ \Rightarrow D(\mathbf{x}_i) \leqslant D(\mathbf{x}_j), \\ \text{or } &\mathbf{x}_i \succcurlyeq_B \mathbf{x}_j \ \Rightarrow D(\mathbf{x}_i) \geqslant D(\mathbf{x}_j). \end{aligned} \tag{3}$$

In partially monotonic classification tasks, criteria are monotonic with class labels [27,28]. Partial monotonicity constraints monotonically depend on some attributes, but not all [4]. Henceforth, B is the set of criteria and $A - B$ is the set of regular attributes. For $\forall \mathbf{a}_k \in A$, if attribute $\mathbf{a}_k$ has a monotonic relationship with decision $\mathbf{d}$, then $\mathbf{a}_k$ is a criterion. If attribute $\mathbf{a}_k$ is not a criterion, it is a regular attribute, and belongs to $A - B$ [43].

The direction of a monotonic relationship is important in evaluating classification performance. In practice, there are two kinds of monotonicity: increasing and decreasing. In a multicriteria decision system, monotonic decrease can be transformed into a monotonic increase, which requires distinguishing monotonic directions from the evaluation of the rank discrimination measurement.

A monotonic function f: $U \to D$ assigns a class label to each object in U. For $\forall \mathbf{x}_i$, $\mathbf{x}_j \in U$, we say that $\mathbf{a}$ increases monotonically with $\mathbf{d}$ if $\mathbf{x}_i \preccurlyeq_{\mathbf{a}} \mathbf{x}_j \Rightarrow f(\mathbf{x}_i) \leq f(\mathbf{x}_j)$. Analogously, we say that $\mathbf{b}$ decreases monotonically with respect to $\mathbf{d}$ if $\mathbf{x}_i \preccurlyeq_{\mathbf{b}} \mathbf{x}_j \Rightarrow f(\mathbf{x}_i) \geq f(\mathbf{x}_j)$. A multicriteria decision task involves an ordinal decision attribute and at least one criterion. According to the monotonic function, we say that **a, b** are increasing and decreasing criteria, respectively. For partially monotonic classification tasks, if the relationship between the attribute and the decision does not satisfy monotonicity constraints, then the attribute is a regular attribute.

Given a set of objects U, $B \subseteq A$, $\widetilde{y}_i$ is a class label of Y. In monotonic classification, the upper and lower approximations of $\widetilde{y}_i^{\leqslant}$ are defined as

$$\underline{R_B^{\preccurlyeq}}\widetilde{y}_i^{\leqslant} = \{\mathbf{x}_j \in U \mid \left[\mathbf{x}_j\right]_B^{\preccurlyeq} \subseteq \widetilde{y}_i^{\leqslant}\},$$
$$\overline{R_B^{\preccurlyeq}}\widetilde{y}_i^{\leqslant} = \{\mathbf{x}_j \in U \mid [\mathbf{x}_i]_B^{\succcurlyeq} \cap \widetilde{y}_i^{\leqslant} \neq \phi\}. \tag{4}$$

In dominance-based rough sets, we cannot construct an effective decision model based on the formal framework because it is sensitive to noise. Therefore, we introduce some more robust measurements.

2.2. Evaluating feature quality

This section presents some measures of feature quality in partially monotonic classification tasks, such as entropy, mutual information (MI), rank entropy, RMI, and the maximal information coefficient (MIC).

Shannon's entropy is a measure of classification consistency. In 2010, Hu et al. generalized Shannon's entropy [26]. Let $[\mathbf{x}_i]_B$ be the equivalence class of samples $\mathbf{x}_i$ and attribute set B. Then, the entropy of U is defined as

$$H_B(U) = -\frac{1}{|U|}\sum_{i=1}^{n} \log \frac{|[\mathbf{x}_i]_B|}{|U|}. \tag{5}$$

MI illustrates the degree of consistency between attribute and decision values. It is widely used in feature selection and classification learning [29]. The MI of attribute sets B and C is defined as

$$MI(B,C) = -\frac{1}{|U|}\sum_{i=1}^{n} \log_2 \frac{|[\mathbf{x}_i]_B| \times |[\mathbf{x}_i]_C|}{|U| \times |[\mathbf{x}_i]_B \cap [\mathbf{x}_i]_C|}. \tag{6}$$

Rank entropy is a rank version of Shannon's entropy and can characterize monotonicity constraints on ordinal classification tasks. RMI is driven by both MI and rank entropy, and measures monotonic consistency.

Definition 3. Let U be a set of objects described by attribute set A, $B \subseteq A$. The descending rank entropies with respect to B are defined as

$$RH_B^{\preccurlyeq}(U) = -\frac{1}{|U|}\sum_{i=1}^{n} \log_2 \frac{\left|[\mathbf{x}_i]_B^{\preccurlyeq}\right|}{|U|}, \tag{7}$$

and the ascending rank entropies with respect to B are defined as

$$RH_B^{\succcurlyeq}(U) = -\frac{1}{|U|}\sum_{i=1}^{n} \log_2 \frac{\left|[\mathbf{x}_i]_B^{\succcurlyeq}\right|}{|U|}. \tag{8}$$

Definition 4. Let U be a set of objects described by attribute sets A, $B \subseteq A$ and $C \subseteq A$. The downward rank mutual information (DRMI) of set U regarding B and C is defined as

$$RMI^{\leqslant}(B,C) = -\frac{1}{|U|}\sum_{i=1}^{n}\log_2\frac{\left|[\mathbf{x}_i]_B^{\leqslant}\right| \times \left|[\mathbf{x}_i]_C^{\leqslant}\right|}{|U| \times \left|[\mathbf{x}_i]_B^{\leqslant} \cap [\mathbf{x}_i]_C^{\leqslant}\right|}, \tag{9}$$

and the ascending rank mutual information (ARMI) of set U regarding B and C is defined as

$$RMI^{\geqslant}(B,C) = -\frac{1}{|U|}\sum_{i=1}^{n}\log_2\frac{\left|[\mathbf{x}_i]_B^{\geqslant}\right| \times \left|[\mathbf{x}_i]_C^{\geqslant}\right|}{|U| \times \left|[\mathbf{x}_i]_B^{\geqslant} \cap [\mathbf{x}_i]_C^{\geqslant}\right|}. \tag{10}$$

Rank entropy and RMI are widely discussed in and applied to ordinal classification. In this study, we use MI and RMI to estimate the feature importance of the splitting point.

MIC is generally regarded as a normalized estimate of MI; however, it is defined as a statistic rather than a measurement [31]. For partially monotonic problems, MIC is used to remove irrelevant features.

For a dataset of size N, and two random variables X and Y, $|X|$ and $|Y|$ denote the number of bins imposed along the $X-$ and Y-axes, respectively. The total number of bins $|X||Y|$ does not exceed specified value T. MIC is computed by

$$MIC(X,Y) = \max_{|X||Y|<T}\frac{MI(X,Y)}{\log_2(\min(|X|,|Y|))}, \tag{11}$$

where $T = N^{0.6}$ or $T = N^{0.55}$ [31].

3. Feature monotonicity

Attributes and criteria generally coexist in decision support systems. To help managers make correct decisions, we propose PMDTs. Decision trees are generally considered effective and efficient for building classification models. These algorithms are greedy by nature, and build trees in a top-down manner. Feature evaluation is therefore an important strategy for constructing decision trees.

In this study, we use the upward and downward RIR to determine whether features and decisions have a monotonic relationship. Furthermore, we can use RIR to support the monotonic directions of the criteria and decisions. Let $\mathbf{x_1}, \mathbf{x_2} \in U$ be two objects and $B \subseteq A$ be a subset of condition attributes. In this case, object $\mathbf{x_1}$ is better than $\mathbf{x}_2$ in terms of features B, but $\mathbf{x_1}$ obtains a worse decision than $\mathbf{x_2}$. We say that $\mathbf{x_1}$ is an inconsistency object of $\mathbf{x}_2$.

VC-DRSA can discover global and local monotonicity relationships using consistency measures [4]. This approach can also distinguish the monotonic direction of attributes from that of the decision. In this work, inspired by dominance inconsistency rates, we propose RIR to calculate unknown monotonicity.

3.1. Rank inconsistency rate

In multicriteria decision tasks, let $\mathbf{a} \in A$ be an ordinal condition attribute, and $\mathbf{d}$ a decision. Assume that we do not know whether the monotonic relationship between the values of attribute $\mathbf{a}$ and those of decision $\mathbf{d}$ is increasing or decreasing. To resolve the monotonic direction of $\mathbf{a}$, we define RIR to determine whether $\mathbf{a}$ is an increasing or a decreasing criterion.

Definition 5. Let U be a set of objects described using attribute set A, $a \in A$ be an ordinal condition attribute set, and $\mathbf{d}$ be a decision. The upward rank inconsistency rate (*URIR*) with respect to a is defined as

$$URIR^{\geqslant}(a,d) = -\frac{1}{|U|}\sum_{i=1}^{n}\log_2\frac{\left|[\mathbf{x}_i]_a^{\geqslant}\right| \times \left|[\mathbf{x}_i]_d^{\leqslant}\right|}{|U| \times \left|[\mathbf{x}_i]_a^{\geqslant} \cap [\mathbf{x}_i]_d^{\leqslant}\right|}, \tag{12}$$

and the downward rank inconsistency rate (*DRIR*) with respect to a is defined as

$$DRIR^{\leqslant}(a,d) = -\frac{1}{|U|}\sum_{i=1}^{n}\log_2\frac{\left|[\mathbf{x}_i]_a^{\leqslant}\right| \times \left|[\mathbf{x}_i]_d^{\geqslant}\right|}{|U| \times \left|[\mathbf{x}_i]_a^{\leqslant} \cap [\mathbf{x}_i]_d^{\geqslant}\right|}. \tag{13}$$

Moreover, we represent the difference between $URIR^{\geq}(a, d)$ and $DRIR^{\leq}(a, d)$ as

$$diff_a = URIR^{\geqslant}(a,d) - DRIR^{\leqslant}(a,d). \tag{14}$$

Note that the definitions of *URIR* and *DRIR* present a single ordinal attribute. Intuitively, $\mathbf{a}$ is strictly monotonically nondecreasing with regard to $\mathbf{d}$ for $URIR^{\geqslant}(a,d) = 0$. Analogously, when $DRIR^{\leqslant}(a,d) = 0$, $\mathbf{a}$ is strictly monotonically nonincreasing with regard to $\mathbf{d}$. This means that there are some inconsistency objects for higher absolute values of $URIR^{\geq}(a, d)$ or $DRIR^{\leq}(a, d)$. $diff_a$ may be very small. Thus, we set a threshold $\delta \in [0, 1]$ to judge whether a globally monotonic relationship exists between $\mathbf{a}$ and $\mathbf{d}$. Specifically, we consider that $\mathbf{a}$ has no globally monotonic relationship with $\mathbf{d}$ for $diff_a \in [-\delta, \delta]$. We must consider the direction of the monotonic relationship when $diff_a \notin [-\delta, \delta]$. Generally, if $diff_a \in (\delta, \infty)$, $\mathbf{a}$ decreases monotonically with regard to $\mathbf{d}$; if $diff_a \in (-\infty, -\delta)$, $\mathbf{a}$ increases monotonically with regard to $\mathbf{d}$.

Table 2
An example of the rank inconsistency rate.

Sample	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	$\mathbf{x}_6$	$\mathbf{x}_7$	$\mathbf{x}_8$	$\mathbf{x}_9$	$\mathbf{x}_{10}$	$\mathbf{x}_{11}$	$\mathbf{x}_{12}$
$\mathbf{a}_1$	0.7	0.6	0.8	0.6	0.5	0.4	0.5	0.4	0.2	0.2	0.2	0.1
$\mathbf{a}_2$	0.2	0.1	0.2	0.3	0.4	0.5	0.6	0.5	0.9	1	0.8	1
$\mathbf{a}_3$	0.5	0.1	0.3	0.5	0.2	0.3	0.3	0.1	0.4	0.6	0.5	0.1
$\mathbf{d}$	0	0	0	0	1	1	1	1	2	2	2	2

3.2. Discriminating feature monotonicity

To illustrate the effectiveness of $diff_a$ in determining attribute preference, we obtain a set of objects, as shown in Table 2. In this example, there were 12 samples with two condition attributes and one decision.

$\mathbf{a_1}$ clearly appears to decrease monotonically with regard to $\mathbf{d}$. Conversely, $\mathbf{a_2}$ is likely to be monotonically increasing with regard to $\mathbf{d}$. Moreover, relationship between $\mathbf{a_3}$ and $\mathbf{d}$ is not globally monotonic. Furthermore, we calculated the indicators relevant for verifying the effectiveness of RIR and this intuitive conclusion. We calculated the difference between $\mathbf{a_1}$, $\mathbf{a_2}$, $\mathbf{a_3}$, and the decision, which were $dif f_{a_1} = URIR^{\geqslant}(a_1, d) - DRIR^{\leqslant}(a_1, d) = 0.7233 - (-0.3828) = 1.1061$, $dif f_{a_2} = URIR^{\geqslant}(a_2, d) - DRIR^{\leqslant}(a_2, d) = (-0.4659) - 0.7233 = -1.1892$, and $dif f_{a_3} = URIR^{\geqslant}(a_3, d) - DRIR^{\leqslant}(a_3, d) = 0.0961 - 0.0770 = 0.0192$ in this example. We set the threshold $\delta = 0.10$. We confirmed $\mathbf{a_1}$ as a decreasing criterion, $\mathbf{a_2}$ as an increasing criterion, and $\mathbf{a_3}$ as a regular attribute. These were consistent with our intuitive conclusion.

As per the above analysis, we intend to judge some problems with partial monotonicity constraints using RIR. For example, we can determine the globally monotonic relationship and direction of each ordinal condition attribute. The RIR algorithm is shown in Algorithm 1. Herein, threshold δ was set from 0.00 to a certain value, in increments of 0.05.

Algorithm 1 Feature monotonicity discrimination with *RIR*

Input:
$S = (\mathbf{x}_i, y_i)$: a set of objects.
δ: a threshold.
Output:
$dif f_a$: Difference between upward and downward RIR.

```
Read the original datasets (x_i, y_i).
for Each attribute of the objects do
    Compute URIR^≥(a, d) using Eq.(12).
    Compute DRIR^≥(a, d) using Eq.(13).
    Compute dif f_a using Eq.(14).
end for
if dif f_a ⩾ −δ and dif f_a ⩽ δ. then
    Return the regular attributes.
    if dif f_a > δ then
        Return the criteria of monotonic decrease
    end if
else
    Return the criteria of monotonic increase.
end if
```

4. Partially monotonic decision trees

PMDTs are constructed in four stages: First, we use MIC to remove irrelevant features. In the second stage, we distinguish attributes from criteria by utilizing the difference between the upward and downward RIRs. Moreover, the difference can also identify monotonic directions. In the third stage, we use the best split to generate partitions using MI. We set the threshold value to stop the growth of the decision trees. In the last stage, we generate partitions using RMI and make the best split on the partition obtained in the third stage. Moreover, the threshold value is set to stop further splitting.

4.1. Measuring the splitting point

When constructing decision trees, it is important to find the best splitting point by utilizing effective measurement [9,41,45]. Splitting criteria include Gini impurity, condition entropy, MI, and RMI. The feature evaluation of PMDTs can be divided into two parts. First, the inducer searches for the best attribute with which to divide the objects into two partitions using MI. Second, the algorithm searches for the best splitting criterion by employing RMI. The difference in processing between the two steps is an object-oriented distinction, whereby the former handles attributes and the latter criteria.

Here, we set a stopping condition for the first stage. Depending on the case, we use different constraints to decide whether to split in the next stage. In the context of decision trees, MI is also called information gain, and is an important criterion that uses impurity as a measure of entropy. RMI can measure ordinal structures and monotonic consistency.

Because regular attributes and criteria coexist in multicriteria decision making, we employ different discrimination measurements to determine the importance of the splitting point. Depending on the task, we use MI and RMI to construct binary decision trees. At each splitting point, the samples are divided into two partitions. We assume that U_i is the subset of objects in the given node and v is the best splitting point. The objects are divided into U_{i1} and U_{i2}, where $U_{i1} = \{\mathbf{x} \in U_i, \mathbf{a}_i(\mathbf{x}) \leqslant v\}$ and $U_{i2} = \{\mathbf{x} \in U_i, \mathbf{a}_i(\mathbf{x}) > v\}$. Each division generates the best split based on the splitting criterion. The procedure continues until the stopping criterion is satisfied.

For binary trees, each attribute $\mathbf{a_i}$ must be binarized. In detail, the values of the attributes $\mathbf{a}_i$ are denoted by $\mathbf{a}_i^{v_k}$, where v_k is the *kth* value. Then, the binary attribute is defined as

$$\mathbf{a}_i^{v_k} = \begin{cases} 0 & \mathbf{a}_i(\mathbf{x}) \leq v_k \\ 1 & \mathbf{a}_i(\mathbf{x}) > v_k \end{cases} . \tag{15}$$

First, we employ MI to evaluate the quality of regular attributes. Given a set of objects U_i and attribute set $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_k\}$, we select the best splitting point that satisfies the following condition:

$$\begin{aligned} \mathbf{x}^*_{\mathbf{a}_k} &= \arg\max MI(\mathbf{a}_k, \mathbf{d}) \\ &= \arg\max -\frac{1}{|U_i|} \sum_{\mathbf{x} \in U_i} \log \frac{\left|[\mathbf{x}_i]_{\mathbf{a}_k}\right| \times |[\mathbf{x}_i]_d|}{|U_i| \times \left|[\mathbf{x}_i]_{\mathbf{a}_k} \cap [\mathbf{x}_i]_d\right|}. \end{aligned} \tag{16}$$

Second, we evaluate the quality of the criterion using RMI. For a set of criteria $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_t\}$ and subset of objects U_i, we select the best splitting point that satisfies the following condition:

$$\begin{aligned} \mathbf{x}^*_{\mathbf{a}_t} &= \arg\max RMI(\mathbf{a}_t, \mathbf{d}) \\ &= \arg\max -\frac{1}{|U_i|} \sum_{\mathbf{x} \in U_i} \log_2 \frac{\left|[\mathbf{x}_i]^{\leqslant}_{\mathbf{a}_t}\right| \times \left|[\mathbf{x}_i]^{\leqslant}_{d}\right|}{|U_i| \times \left|[\mathbf{x}_i]^{\leqslant}_{\mathbf{a}_t} \cap [\mathbf{x}_i]^{\leqslant}_{d}\right|}. \end{aligned} \tag{17}$$

4.2. The PMDT algorithm

Before presenting the algorithm, we illustrate the rules of building the decision tree. First, we introduce the splitting rules in detail. In two stages, the proposed algorithm searches for the best splitting point with which to partition objects based on the splitting rules. The algorithm stops when the stopping criterion is satisfied. The change in impurity between the two steps is referred to as the impurity decrement. Labeling rules are utilized to assign class labels to leaf nodes.

Since a binary tree is used, one value of the attributes is used to split the objects in each node. We compute the largest MI between regular attributes and the decision, and the largest RMI between the criteria and the decision, in the given node. Next, we discuss the stopping criterion. In this study, we set a few specified conditions to guarantee an impurity decrement. When the impurity decrement of two successive nodes is decreasing, the inducer uses the criteria to generate the best splitting point until all objects belonging to the same class, or the RMI of the best criterion, are smaller in number than threshold θ. With regard to the labeling rule, all objects in a leaf node are from the same class. The leaf node obtains the class label at any given time. Nevertheless, the objects are from different classes, and the median class is assigned to the leaf node. Furthermore, if two classes have the same number of objects and the given node is a left partition, we assign the worse class to this node; otherwise, we assign it the better class.

The PMDT algorithm is formulated in Algorithm 2. The characteristics of the proposed algorithm are apparent, and the decision tree generated is not strictly monotonic.

The proposed algorithm can form decision trees with partial monotonicity constraints. In Algorithm 3, we illustrate the impurity decrement procedure, which is restricted to hold a certain criterion.

5. Experiment and discussion

In this section, we evaluate the performance of the proposed algorithm in several experiments. We compared our algorithm with baseline algorithms on a few monotonic classification tasks. Before comparison, we validated the effects of RIR on artificial datasets. Moreover, we also used these data to construct a PMDT model. To demonstrate the effectiveness of the proposed algorithm, we introduce some monotone classifiers, such as monotonic univariate trees with RMI (REMT) [25], monotonic univariate trees with rank Gini impurity (RGMT) [35], the ordinal learning model (OLM) [1], and the ordinal stochastic dominance learner (OSDL) [34]. In 2011, Błaszczyński proposed the VC-DomLEM algorithm with monotonicity constraints [7] . Wang et al. refined a preprocessing transformation method to determine preference directions [43]. It was applied to the VC-DomLEM algorithm for monotonic classification tasks. We call it DIR-DomLEM. We generated PMDTs to handle nonmonotonic attributes.

Algorithm 2 Partially monotonic decision trees

Input:

$S = \{(\mathbf{x}_i, y_i)\}$: a set of samples.

ϵ: stopping criterion.

Output:

T: partially monotonic decision trees.

Read the original datasets and normalize samples $\mathbf{x}_i$.
Remove irrelevant features using *MIC*.
Use new feature sets $S^{'}(\mathbf{x}_i, y_i)$, which exclude irrelevant features.
Determine monotonic relationships based on *RIR* (see Algorithm 1).
Create node N using $S^{'}$.
if all samples belong to class C in N **then**
    Return N as a leaf node and assign it class C.
    if N contains fewer than Min_{obj} samples **then**
        Return N as a leaf node and assign it to majority class C.
    end if
else
    The impurity decrement condition is shown in Algorithm 3.
    if Condition is TRUE **then**
        Generate the best split $\mathbf{x}^*_{\mathbf{a}_k}$ using Eq. (16).
    else
        Generate the best split $\mathbf{x}^*_{\mathbf{a}_k}$ using Eq.(17).
    end if
    if *partition* is empty or $\mathbf{x}^*_{\mathbf{a}_k}$ less than ϵ **then**
        Stop growing tree and return T.
    else
        Recursively produce new splits according to the above.
    end if
end if

Algorithm 3 Impurity decrement condition

Input:

ds: a subset of objects.

ra: sizes of regular attributes.

cr: Boolean flag of impurity decrement.

Output:

Condition: the of impurity decrement condition.

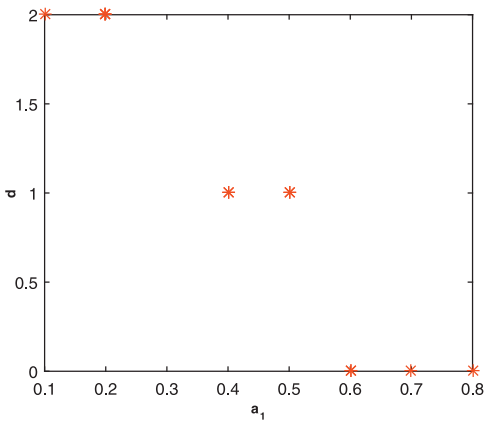
Initialize *cr*.
Compute *MI* of *ds*:*DMI*.
Compute the average of *DMI*:*AvgMI*
if $ra == 1$ **then**
    cr is *TRUE*.
end if
if $ra == 2$ and the decrement of *AvgMI* is less than a threshold **then**
    cr is *TRUE*.
end if
if $ra > 2$ and the decrement of *AvgMI* is decreasing **then**
    cr is *TRUE*.
end if

5.1. Evaluation on artificial datasets

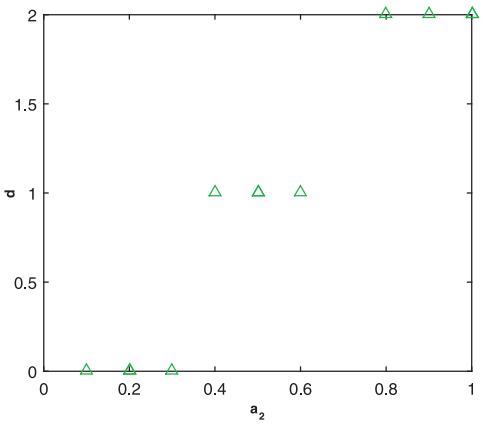
We generated a set of artificial samples with three classes, as shown in Table 2. The first feature, $\mathbf{a}_1$, decreased monotonically with the decision and the second, $\mathbf{a}_2$, increased monotonically. The third feature, $\mathbf{a}_3$, was nonmonotonic with the decision.

We first present the relationship between the features and decision. Figs. 1a and b show features with monotonicity constraints and different monotonic directions. In Fig. 1c, feature $\mathbf{a}_3$ is nonmonotonic with the decision.

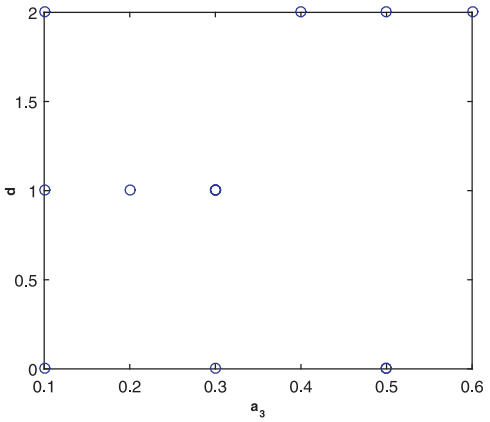
The results in Table 2 are consistent with Fig. 1. We also drew a 3*D* scatter plot of the data, as shown in Fig. 2a. The monotonicity of the data distribution was obvious. To illustrate the characteristics of different data distributions, we intro-



(a) The relationship between $\mathbf{a}_1$ and $\mathbf{d}$



(b) The relationship between $\mathbf{a}_2$ and $\mathbf{d}$



(c) The relationship between $\mathbf{a}_3$ and $\mathbf{d}$

Fig. 1. The relationship between features and the decision.

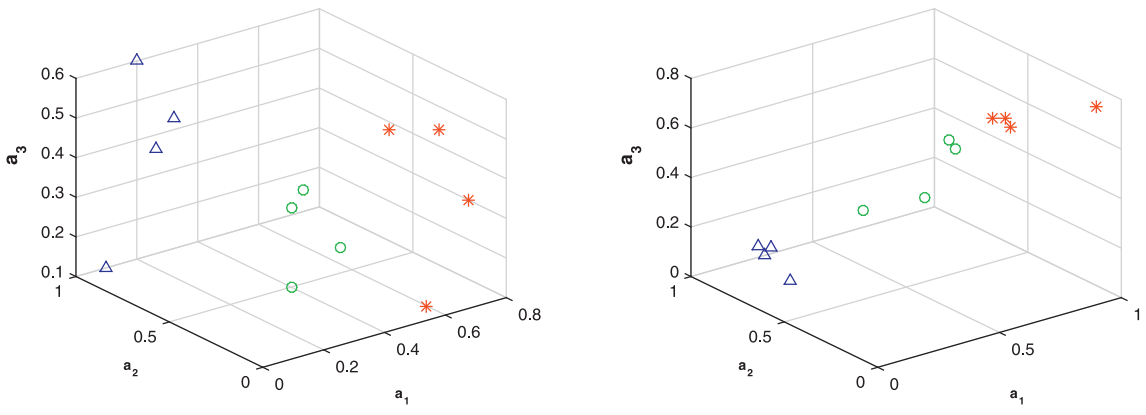
Table 3
A toy partially monotonic classification task.

Sample	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	$\mathbf{x}_6$	$\mathbf{x}_7$	$\mathbf{x}_8$	$\mathbf{x}_9$	$\mathbf{x}_{10}$	$\mathbf{x}_{11}$	$\mathbf{x}_{12}$
$\mathbf{a}_1$	0.6	0.7	0.9	0.7	0.5	0.7	0.6	0.4	0.1	0.2	0.1	0.3
$\mathbf{a}_2$	0.1	0.2	0	0.3	0.4	0.5	0.4	0.6	0.6	0.9	0.7	1
$\mathbf{a}_3$	0.8	0.7	0.8	0.7	0.4	0.5	0.6	0.3	0.1	0.1	0.2	0
$\mathbf{d}$	0	0	0	0	1	1	1	1	2	2	2	2

duced a second classification task with monotonicity constraints, shown in Table 3. The 3*D* plot is shown in Fig. 2b. One can see the monotonic relationship between attributes and the decision. The regular attribute led to the difference between the two distributions.

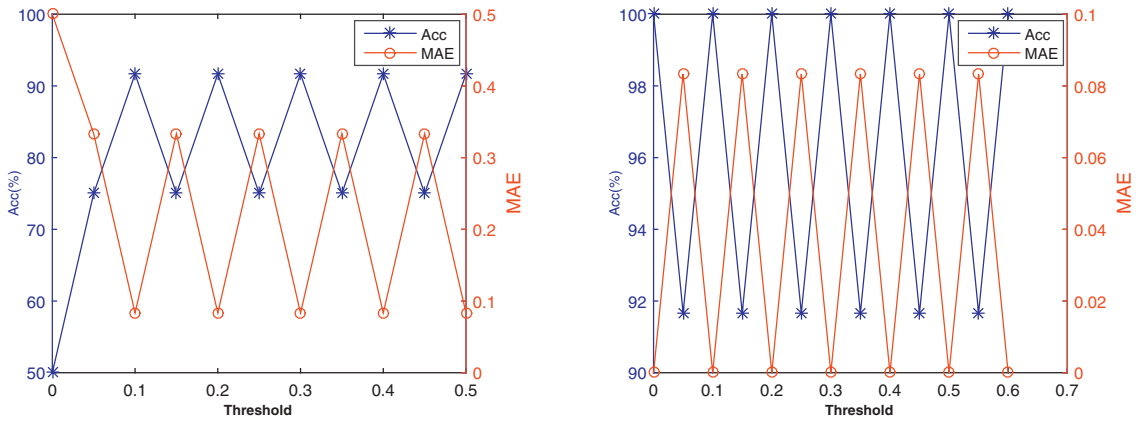
We also considered the effect that changes in threshold δ might have on performance. We computed the *URIR*, *DRIR*, and $diff_a$ between the features and decision on the artificial datasets. In this task, we set the threshold in terms of $diff_a$:

$$diff_{norm} = \frac{diff_a - \min(diff_a)}{\max(diff_a) - \min(diff_a)}.$$



(a) Scatter plot of the first task. (b) Scatter plot of the second task.

Fig. 2. Scatter plots of the artificial data in 3D space.



(a) The performance curves for the first task. (b) The performance curves for the second task.

Fig. 3. Performance curves with threshold changes.

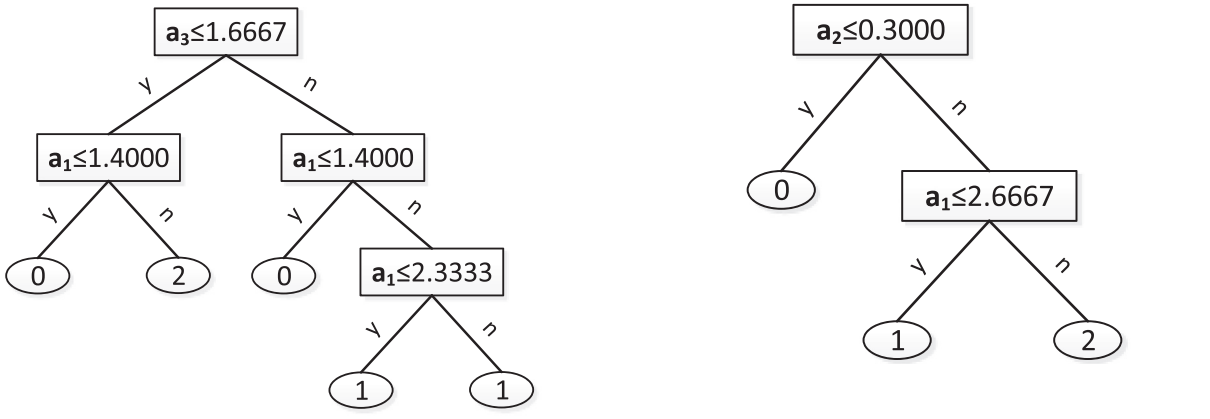
We trained the algorithm on the artificial datasets with $diff_a$ ranging from 0.00 to V_{diff}, in increments of 0.05, where V_{diff} is the absolute value of the mean of $diff_{norm}$. The performance curves of the first task are shown in Fig. 3a. Performance changed with the threshold value. Fig. 3b shows the performance curves for the second task. The classifier selected the best performance in the experiment and obtained a reasonable threshold value.

Finally, we generated the PMDT model. In Fig. 4a, we see that decision trees model contained four splitting points on the first task. Of course, we generated decision rules that were not all monotonically consistent in the partially monotonic classification tasks.

A new decision tree model is shown in Fig. 4b for the second task. It is obvious that all decision rules were monotonic, which was significantly different from the first task. Thus, the proposed algorithm can process monotonic and partially monotonic tasks, and perfectly fit the training data.

5.2. Comparison with other algorithms on real-world tasks

We discuss the performance of PMDT on real-world tasks. A total of 12 real datasets were collected from the UCI [33] and Weka [24] repositories. Their detailed descriptions are shown in Table 4. In these tasks, monotonic directions were unknown for the ordinal condition attributes. All datasets were preprocessed by normalizing the attributes and removing missing data. Monotonic directions were determined by RIR. Monotonic decreases were transformed to monotonic increases by using negative feature values.



(a) PMDT model with partial monotonicity (b) PMDT model with complete monotonicity

Fig. 4. The decision tree model.

Table 4
Information concerning datasets.

No.	Data set	Instance	Attribute	Class
1	Car	1728	6	4
2	Wine	1599	11	6
3	Tae	151	5	3
4	Credit	690	15	2
5	Breast	699	9	2
6	CPU	209	6	3
7	Housing	506	13	5
8	Diabetes	768	8	2
9	Eucalyptus	641	15	5
10	Grub-damage	155	8	4
11	Ionosphere	351	34	2
12	Heart	270	13	2

In this experiment, we compared PMDT with other monotonic classification algorithms, including REMT, RGMT, and the two algorithms in [43]. We evaluated their performance using 5×2 cross-validation. Our algorithm handled the criteria and attributes, and could distinguish monotonic directions. Therefore, the MI of each regular attribute and the RMI of each criterion were calculated on the training set. We performed 5×2 cross-validation on each dataset and computed the average performance. The mean absolute error (MAE) and classification accuracy (Acc) were used to evaluate the performance of the proposed algorithm.

For each task, we used the criteria and attributes to obtain the best splitting point. Furthermore, we built PMDTs using different strategies. In Fig. 5, our algorithm handled more criteria on the Car, Wine, Breast, CPU, Housing, Diabetes, Eucalyptus, and Heart tasks, and more attributes were captured on other tasks. We briefly analyze two of these to illustrate some phenomena. In the Car task, we found that the feature *person* (number of persons to carry) was ordinal with scales 1 (two persons), 2 (four people), 3 (more than four people), and that feature *door* (number of doors) was also ordinal. Moreover, our algorithm used monotonic preference information between these features and the decision. In the Tae task, our method treated all features as attributes. Although the *Class size* feature was ordinal, our approach did not use the monotonic preference between it and the decision, which may have been because these monotonic preferences did not provide more knowledge for decision making. Therefore, we can conclude that the proposed algorithm works well with criteria and attributes, and obtained the best features with which to complete classification tasks.

For the PMDT, threshold δ was tuned to achieve optimal performance, thus generating better Acc based on 5×2 cross-validation. The results of the experiment are shown in Table 5. The fifth and sixth columns show the differences and ranks of our algorithm and DIR-DomLEM, respectively. The last two columns show the difference between and rank of our algorithm and VC-DomLEM, respectively. PMDT obtained better Acc on 11 tasks.

We used the Wilcoxon signed-rank method to compare the difference between our algorithm and the others. According to [44], the total rank of difference1 was $TR_1 = min(TR_1^+, TR_1^-)$, where $TR_1^+ = 74$ is the sum of positive ranks, and $TR_1^- = 4$ is the sum of negative ranks. Similarly, the total rank of different2 was $TR_2 = min(TR_2^+, TR_2^-) = min(75, 3) = 3$. Because $TR_1 = 4$ and $TR_2 = 3$ were smaller than the critical value 10, we rejected the null hypothesis with confidence 0.95. That is, our algorithm is significantly better than the other two.

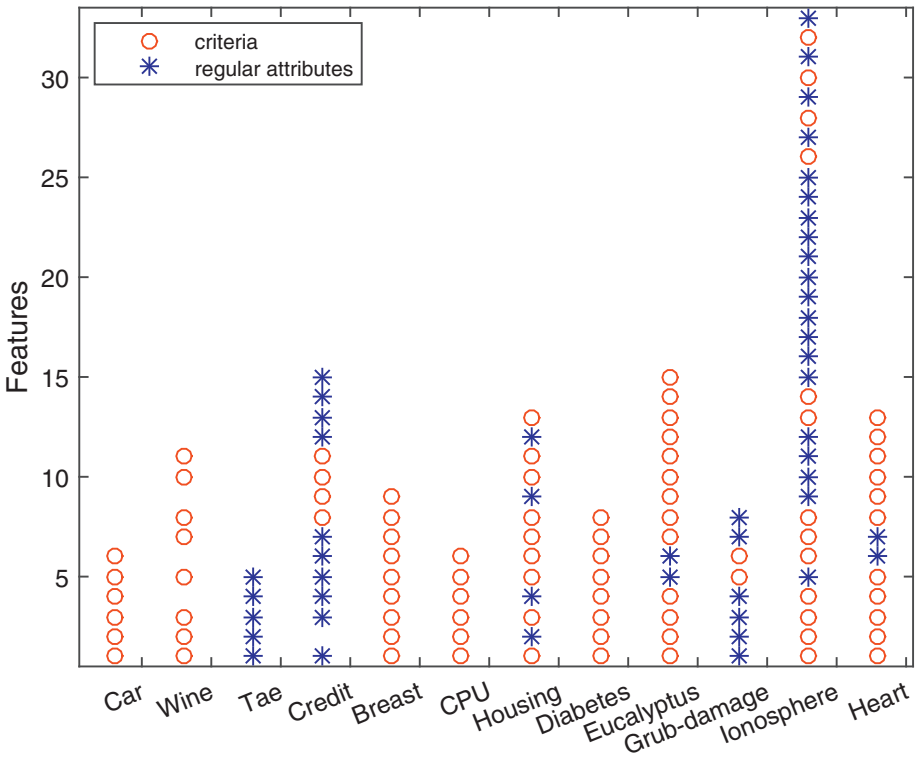


Fig. 5. Criteria and regular attributes, where irrelevant features have been removed using MIC. Of these, the criteria refer to those with a monotonic relationship between features and the decision; attributes are those with a nonmonotonic relationship with the decision.

Table 5
Accuracy of algorithms on real-world tasks.

Datasets	PMDT	DIR-DomLEM	VC-DomLEM	Difference1	Rank1	Difference2	Rank2
Car	96.67	76.49	76.53	+20.18	+12	+20.14	+12
Wine	63.75	56.17	50.81	+7.58	+7	+12.94	+10
Tae	59.09	51.01	50.46	+8.08	+8	+8.63	+7
Credit	89.49	81.38	80.73	+8.11	+9	+8.76	+8
Breast	95.36	94.67	94.85	+0.69	+2	+0.51	+1
CPU	89.14	91.11	91.11	−1.97	−3	−1.97	−3
Housing	72.28	62.09	60.99	+10.19	+10	+11.29	+9
Diabetes	72.64	73.26	72.09	−0.62	−1	+0.55	+2
Eucalyptus	59.77	53.95	53.14	+5.82	+5	+6.63	+4
Grub-damage	51.61	44.14	44.65	+7.47	+6	+6.96	+6
Ionosphere	92.86	77.84	76.81	+15.02	+11	+16.05	+11
Heart	83.33	79.48	76.52	+3.85	+4	+6.81	+5

Table 6
MAE of algorithms on real-world tasks.

Datasets	PMDT	DIR-DomLEM	VC-DomLEM	Difference1	Rank1	Difference2	Rank2
Car	0.0420	0.2539	0.2530	−0.2120	−12	−0.2110	−12
Wine	0.3813	0.4873	0.5641	−0.1061	−7	−0.1829	−10
Tae	0.4914	0.6209	0.6211	−0.1295	−8	−0.1297	−7
Credit	0.1051	0.1868	0.1927	−0.0818	−5	−0.0876	−5
Breast	0.0464	0.0510	0.0515	−0.0045	−1	−0.0051	−1
CPU	0.1086	0.0947	0.0947	+0.0139	+2	+0.0139	+3
Housing	0.2871	0.4427	0.4514	−0.1556	−10	−0.1643	−9
Diabetes	0.2736	0.2586	0.2792	−0.0150	−3	−0.0055	−2
Eucalyptus	0.4453	0.5426	0.5517	−0.0973	−6	−0.1063	−6
Grub-damage	0.5645	0.7665	0.7587	−0.2020	−11	−0.1942	−11
Ionosphere	0.0714	0.2125	0.2319	−0.1411	−9	−0.1605	−8
Heart	0.1667	0.2067	0.2348	−0.0400	−4	−0.0681	−4

Table 7
Comparative analysis of accuracy.

Datasets	PMDT	REMT	RGMT	OLM	OSDL
Car	**96.301.70**	87.211.92	87.611.92	72.292.91	71.643.27
Wine	**60.911.90**	58.233.25	57.793.18	35.962.88	54.973.81
Tae	**60.9610.45**	44.3311.55	46.3011.34	39.6714.09	33.089.78
Credit	**88.123.04**	87.684.39	87.684.39	77.684.69	73.045.76
Breast	94.712.43	92.425.39	93.423.45	88.564.09	**96.002.00**
CPU	**88.955.22**	88.056.03	88.056.03	83.716.86	74.1412.58
Housing	**73.905.62**	63.075.92	64.826.07	10.273.91	35.996.28
Diabetes	73.825.05	**74.214.64**	73.165.17	29.817.10	57.294.45
Eucalyptus	**59.929.41**	57.256.30	58.035.90	30.447.75	39.325.52
Grub-damage	48.9613.10	**50.9211.47**	45.9210.26	31.6311.84	31.549.97
Ionosphere	**90.874.44**	88.323.15	89.734.32	51.836.36	65.217.57
Heart	82.599.41	82.963.98	**83.333.60**	67.044.77	60.005.18
Average	**76.67**	72.89	72.99	51.57	57.69

Table 8
Comparative analysis of MAE.

Datasets	PMDT	REMT	RGMT	OLM	OSDL
Car	**0.04340.0230**	0.14470.0214	0.14180.0211	0.38140.0404	0.30900.0384
Wine	**0.41590.0238**	0.45590.0402	0.45710.0394	0.89620.0560	0.49970.0462
Tae	**0.47710.1539**	0.68920.1281	0.66330.1337	0.87500.2351	0.76880.1434
Credit	**0.11880.0304**	0.12320.0439	0.12320.0439	0.22320.0469	0.26960.0576
Breast	0.05290.0243	0.07580.0539	0.06580.0345	0.11440.0409	**0.04000.0200**
CPU	**0.11050.0522**	0.11950.0603	0.11950.0603	0.16290.0686	0.25860.1258
Housing	**0.27890.0663**	0.37920.0630	0.37350.0622	1.63590.0897	0.88930.0888
Diabetes	0.26180.0505	**0.25790.0464**	0.26840.0517	0.70190.0710	0.42710.0445
Eucalyptus	**0.45070.07687**	0.46810.0776	0.45720.0854	1.31470.1748	0.74710.0674
Grub-damage	0.65210.2079	**0.63250.1629**	0.72630.2358	1.08130.2726	0.93130.1601
Ionosphere	**0.09130.0444**	0.11680.0315	0.10270.0432	0.48170.0636	0.34790.0757
Heart	0.17410.0941	0.17040.0398	**0.16670.0360**	0.32960.0477	0.40000.0518
Average	**0.2606**	0.3028	0.3055	0.6832	0.4907

Similarly, we evaluated the performance of the PMDT using MAE, as shown in Table 6. The fifth column shows the difference between our algorithm and DIR-DomLEM, and the sixth column shows the ranks of the differences. The seventh and eighth columns are the differences and ranks of PMDT and VC-DomLEM, respectively. PMDT showed better performance on 11 tasks. As shown in Table 6, PMDT's accuracy was superior to that of the other algorithms. We performed a significance test on the MAE. The ranks of the differences were $TR_1^+ = 2$, $TR_1^- = 76$, $TR_2^+ = 3$, and $TR_2^- = 75$. Moreover, $TR_1 = min(TR_1^+, TR_1^-) = 2$ and $TR_2 = min(TR_2^+, TR_2^-) = 3$ were less than the critical value of 10. In other words, the DIR-DomLEM and VC-DomLEM algorithms handled the criteria and attributes, but did not yield better performance. RIR was applied to our algorithm with varying feature importance measurements. MI and RMI were better feature evaluation measurements for the attributes and criteria, respectively. Thus, the PMDT algorithm can significantly improve classification performance.

We now compare the performance of PMDT with that of other monotonic algorithms based on 10-fold cross-validation. These algorithms were all tested in the same experimental environment. The main purpose of the comparisons was to explain the difference in performance on partially monotone classification tasks. In Tables 7 and 8, we list the Acc and MAE, respectively, and the corresponding standard deviations, for each data item and classification algorithm. The highest Acc and lowest MAE of each dataset are given in bold. As shown in Table 7, PMDT obtained the best classification accuracy on 8 tasks, whereas the other algorithms yielded better performance in 5 tasks. PMDT also outperformed the others for 8 tasks in terms of MAE, as shown in Table 8.

These results show that PMDT outperforms other classification algorithms for most tasks in terms of both Acc and MAE. Threshold selection affected the algorithm's performance. On 12 tasks, we obtained a reasonable threshold δ in the range [0, 1]. Moreover, some tasks may have included attributes and criteria, whereas others were completely monotonic. The PMDT algorithm achieved better performance by processing both.

To compare these algorithms, we needed to verify their average performance. We used an effective statistic test to determine whether the PMDT algorithm significantly improved classification. We used one-tailed paired *t*-tests for pairwise comparison of the average performance of all algorithms [13]. We conducted the *t*-test on Acc and MAE, as shown in Tables 9 and 10. The bold numbers show significant differences between the two algorithms, for p-values less than 0.05. As shown in Tables 9 and 10, only PMDT is significantly different from most other algorithms, including REMT, RGMT, OLM and OSDL. According to the results, our algorithm did not achieve the best performance on all classification tasks. However, its average performance was better than that of all other algorithms with 95% confidence.

Table 9
Pairwise significance level of average Acc for the different algorithms.

	PMDT	REMT	RGMT	OLM	OSDL
PMDT	–	–	–	–	–
REMT	**0.02**	–	–	–	–
RGMT	**0.01**	0.43	–	–	–
OLM	**0.00**	**0.00**	**0.00**	–	–
OSDL	**0.00**	**0.00**	**0.00**	0.07	–

Table 10
Pairwise significance level of average MAE for the different algorithms.

	PMDT	REMT	RGMT	OLM	OSDL
PMDT	–	–	–	–	–
REMT	**0.02**	–	–	–	–
RGMT	**0.01**	0.38	–	–	–
OLM	**0.00**	**0.00**	**0.00**	–	–
OSDL	**0.00**	**0.00**	**0.00**	**0.01**	–

6. Conclusion and future work

Classification methods with monotonicity constraints are widely applied to multicriteria decision making. Most methods assume that all features are monotonic with the decision. However, this assumption may not be reasonable in real-world applications. Monotonic and nonmonotonic feature/decision relationships generally coexist in real-world classification tasks. Moreover, nonmonotonic features affect the performance of conventional algorithms with monotonicity constraints. PMDT handles these features to improve classification performance.

In general, monotonic features are called criteria and other features are called regular attributes. The preference directions are unknown for these criteria. Thus, we proposed RIR to determine whether features are monotonic with the decision and estimate the direction of preference.

We conducted some numerical experiments on an artificial dataset and real-world tasks. The results showed that PMDT outperformed REMT, RGMT, and other methods. PMDT was effective for partially monotonic classification tasks. It could handle both regular attributes and criteria to improve its classifications. In this work, MI measured the quality of regular attributes, and RMI determined the quality of criteria and reflects a monotonic relationship.

There remains interesting work in multicriteria decision making. The effectiveness of RIR for feature selection should be validated, and a determination of the effects of threshold δ can yield more practical value in different tasks. We will develop feature selection techniques to discuss these problems in the future.

Acknowledgments

This work was partly supported by the National Program on Key Basic Research Project under Grant 2013CB329304, the National Natural Science Foundation of China under Grant (61222210 and 61432011), and the Qinghai Natural Science Foundation of China (2016-ZJ-922Q).

References

[1] A. Ben-David, Automatic generation of symbolic multiattribute ordinal knowledge-based DSSS: methodology and applications, Decis Sci 23 (6) (1992) 1357–1372.
[2] A. Ben-David, Monotonicity maintenance in information-theoretic machine learning algorithms, Mach Learn 19 (1995) 29–43.
[3] J. Błaszczyński, S. Greco, R. Słowiński, Multi-criteria classification – a new scheme for application of dominance-based decision rules, Eur J Oper Res 181 (3) (2007) 1030–1044.
[4] J. Błaszczyński, S. Greco, R. Słowiński, Inductive discovery of laws using monotonic rules, Eng Appl Artif Intell 25 (2012) 284–294.
[5] J. Błaszczyński, S. Greco, R. Słowiński, M. Szelag, On variable consistency dominance-based rough set approaches, Springer-Verlag, 2006.
[6] J. Błaszczyński, S. Greco, R. Słowiński, M. Szelag, Monotonic variable consistency rough set approaches, Int. J. Approx Reason 50 (2009) 979–999.
[7] J. Błaszczyński, R. Słowiński, M. Szelag, Sequential covering rule induction algorithm for variable consistency rough set approaches, Inf Sci (Ny) 181 (5) (2011) 987–1002.
[8] J. Błaszczyński, R. Słowiński, M. Szelag, Induction of ordinal classification rules from incomplete data, Lect. Notes Comput. Sci. 7413 (2012) 56–65.
[9] L. Breiman, J. Friedman, C.J. Stone, R.A. Olshen, Classification and regression trees, CRC press, 1984.
[10] S. Chakhar, A. Ishizaka, A. Labib, I. Saad, Dominance-based rough set approach for group decisions, Eur J Oper Res 251 (2016) 206–224.
[11] C.-C. Chen, S.-T. Li, Credit rating with a monotonicity-constrained support vector machine model, Expert Syst Appl 41 (16) (2014) 7235–7247.
[12] H. Daniels, M. Velikova, Monotone and partially monotone neural networks, IEEE Trans. Neural Netw 21 (6) (2010) 906–917.
[13] J. Demšar, Statistical comparisons of classifiers over multiple data sets, JMach Learn Res 7 (2006) 1–30.
[14] M. Doumpos, C. Zopounidis, A multicriteria decision support system for bank rating, Decis Support Syst 50 (2010) 55–63.
[15] A. Feelders, Monotone relabeling in ordinal classification, in: Proceedings of IEEE 10th international conference on data mining (ICDM), IEEE Computer Society, 2010.

[16] A. Feelders, M. Pardoel, Pruning for Monotone Classification Trees, in: Proceedings of advances in intelligent data analysis V, Springer, Berlin Heidelberg, 2003.
[17] D. Genest, M. Chein, A content-search information retrieval process based on conceptual graphs, Knowl Inf Syst 8 (2005) 292–309.
[18] S. González, F. Herrera, S. García, Monotonic random forest with an ensemble pruning mechanism based on the degree of monotonicity, New Gener Comput 33 (4) (2015) 367–388.
[19] S. Greco, B. Matarazzo, R. Slowinski, Rough approximation of a preference relation by dominance relations, Eur J Oper Res 117 (1) (1999) 63–83.
[20] S. Greco, B. Matarazzo, R. Slowinski, Rough approximation by dominance relations, Int. J. Intell. Syst. 17 (2) (2002) 153–171.
[21] S. Greco, B. Matarazzo, R. Slowinski, Customer satisfaction analysis based on rough set approach, Z Betr 77 (3) (2007) 325–339.
[22] S. Greco, B. Matarazzo, R. Slowinski, J. Stefanowski, Variable consistency model of dominance-based rough sets approach, in: Revised Papers from the Proceedings of Second International Conference on Rough Sets and Current Trends in Computing, Springer-Verlag, 2001.
[23] P.A. Gutierrez, M. Perez-Ortiz, J. Sanchez-Monedero, F. Fernandez-Navarro, C. Hervas-Martinez, Ordinal regression methods: survey and experimental study, IEEE Tran Knowl Data Eng 28 (1) (2016) 127–146.
[24] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The weka data mining software: an update, ACM SIGKDD Explor Newsl 11 (1) (2009) 10–18.
[25] Q. Hu, X. Che, L. Zhang, D. Zhang, M. Guo, D. Yu, Rank entropy-based decision trees for monotonic classification, IEEE Trans Knowl Data Eng 24 (11) (2012) 2052–2064.
[26] Q. Hu, M. Guo, D. Yu, J. Liu, Information entropy for ordinal classification, Sci China Inf Sci 53 (6) (2010) 1188–1200.
[27] Q. Hu, W. Pan, Y. Song, D. Yu, Large-margin feature selection for monotonic classification, Knowl Based Syst 31 (2012) 8–18.
[28] Q. Hu, W. Pan, L. Zhang, D. Zhang, Y. Song, M. Guo, D. Yu, Feature selection for monotonic classification, IEEE Trans. Fuzzy Syst. 20 (1) (2012) 69–81.
[29] N. Jain, C.A. Murthy, A new estimate of mutual information based measure of dependence between two variables: properties and fast implementation, Int. J. Mach. Learn. Cybern. 7 (5) (2016) 857–875.
[30] M. Kadziński, S. Greco, R. Słowiński, Robust ordinal regression for dominance-based rough set approach to multiple criteria sorting, Inf Sci (Ny) 283 (2014) 211–228.
[31] J.B. Kinney, G.S. Atwal, Equitability, mutual information, and the maximal information coefficient, Proc. Natl. Acad. Sci. USA. 111 (9) (2013) 3354–3359.
[32] S.-T. Li, C.-C. Chen, A regularized monotonic fuzzy support vector machine model for data mining with prior knowledge, IEEE Trans. Fuzzy Syst. 23 (2015) 1713–1727.
[33] M. Lichman, School of Information and Computer Sciences (2013).
[34] S. Lievens, B. De Baets, K. Cao-Van, A probabilistic framework for the design of instance-based supervised ranking algorithms in an ordinal setting, Ann Oper Res 163 (1) (2008) 115–142.
[35] C. Marsala, D. Petturiti, Rank discrimination measures for enforcing monotonicity in decision tree induction, Inf Sci (Ny) 291 (2015) 143–171.
[36] K. Pelckmans, M. Espinoza, J. Brabanter, J.A.K. Suykens, B. Moor, Primal-dual monotone kernel regression, Neural Process Lett 22 (2) (2005) 171–182.
[37] M. Piltan, T. Sowlati, A multi-criteria decision support model for evaluating the performance of partnerships, Expert Syst Appl 45 (2016) 373–384.
[38] Y. Qian, H. Xu, J. Liang, B. Liu, J. Wang, Fusing monotonic decision trees, IEEE Trans Knowl Data Eng 27 (2015) 2717–2728.
[39] D. Schall, A multi-criteria ranking framework for partner selection in scientific collaboration environments, Decis Support Syst 59 (2014) 1–14.
[40] R. Sousa, I. Yevseyeva, J.F.P.d. Costa, J.S. Cardoso, Multicriteria models for learning ordinal data: a literature review, Artificial intelligence, evolutionary computing and metaheuristics: in the footsteps of Alan Turing, Springer, Berlin Heidelberg, 2013.
[41] J. Tanha, M. van Someren, H. Afsarmanesh, Semi-supervised self-training for decision tree classifiers, Int. J. Mach. Learn. Cybern. 8 (1) (2017) 355–370.
[42] M. Velikova, H. Daniels, Decision trees for monotone price models, Comput Manag Sci 1 (3) (2004) 231–244.
[43] H. Wang, M. Zhou, K. She, Induction of ordinal classification rules from decision tables with unknown monotonicity, Eur J Oper Res 242 (1) (2015) 172–181.
[44] F. Wilcoxon, Individual comparisons by ranking methods, Springer, New York, 1992.
[45] H. Zhu, J. Zhai, S. Wang, X. Wang, Monotonic decision tree for interval valued data, in: Proceedings of Machine Learning and Cybernetics: 13th International Conference, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.