A Cost-Sensitive Imprecise Credal Decision Tree based on Nonparametric Predictive Inference

Serafín. Moral-García[a,*], Joaquín Abellán[a], Tahani Coolen-Maturi[b], Frank P.A. Coolen[b]

[a]*Department of Computer Science and Artificial Intelligence*
University of Granada, Granada, Spain
[b] *Department of Mathematical Sciences*
Durham University, Durham, United Kingdom

Abstract

Classifiers sometimes return a set of values of the class variable since there is not enough information to point to a single class value. These classifiers are known as imprecise classifiers. Decision Trees for Imprecise Classification were proposed and adapted to consider the error costs when classifying new instances. In this work, we present a new cost-sensitive Decision Tree for Imprecise Classification that considers the error costs by weighting instances, also considering such costs in the tree building process. Our proposed method uses the Nonparametric Predictive Inference Model, a nonparametric model that does not assume previous knowledge about the data, unlike previous imprecise probabilities models. We show that our proposal might give more informative predictions than the existing cost-sensitive Decision Tree for Imprecise Classification. Experimental results reveal that, in Imprecise Classification, our proposed cost-sensitive Decision Tree significantly outperforms the one proposed so far; even though the cost of erroneous classifications is higher with our proposal, it tends to provide more informative predictions.

Keywords:
cost-sensitive classifiers, Imprecise Classification, error costs, cost-sensitive

*Corresponding Author
Email addresses: `seramoral@decsai.ugr.es` (Serafín. Moral-García), `jabellan@decsai.ugr.es` (Joaquín Abellán), `tahani.maturi@durham.ac.uk` (Tahani Coolen-Maturi), `frank.coolen@durham.ac.uk` (Frank P.A. Coolen)

1. Introduction

Classification algorithms often aim to minimize the number of instances incorrectly classified. This approach would be optimal if all classification errors had the same importance. Nevertheless, in practical applications, classification errors usually have different costs. For example, in *medical diagnosis*, the cost of incorrectly predicting that a patient does not have a serious disease may be much higher than the cost of erroneously predicting that the patient is ill [1, 2, 3]; in *credit fraud detection*, predicting that a credit card is legal when it is fraudulent is likely to cause far higher economical losses for banks and financial institutions than predicting a normal credit card as fraudulent [4, 5, 6]; in *software defect prediction*, the cost of non-defective modules predicted as defective is probably far lower than the cost of defective modules predicted as non-defective [7, 8, 9]. For this reason, classifiers that take the costs of errors into consideration, also called *cost-sensitive classifiers*, have been developed. Examples are cost-sensitive Decision Trees [10, 11, 12], cost-sensitive Naive Bayes [13, 14], cost-sensitive K-Nearest Neighbors [15], and cost-sensitive Neural Networks [16, 17, 18].

In order to classify new instances, cost-sensitive and cost-insensitive classifiers tend to predict a single class value. However, in many cases, there is not enough information available for a classifier to clearly point to a single value of the class variable. In such situations, it is more logical that classifiers predict a set of class values rather than a single one. Predictions of this type are called *imprecise predictions*, and classifiers that make imprecise predictions are known as *imprecise classifiers* [19].

When an imprecise classifier is employed, a set of values of the class variable may be obtained. It is composed of those class values that are not "defeated" by another one according to an established criterion, usually called *dominance criterion.* The obtained set of class values is known as the *non-dominated states set* [19]. An evaluation metric for an imprecise prediction has to consider whether the prediction is correct (the real class value belongs to the non-dominated states set) and how informative the prediction is, which is measured by the cardinality of the non-dominated states set. Likewise, an evaluation metric for a cost-sensitive imprecise classifier must take into account the costs of incorrect predictions and, when the predictions are correct, the number of non-dominated states, as a higher number of non-dominated states implies a less informative prediction.

For developing Imprecise Classification algorithms, theories based on *imprecise probabilities* are more suitable than classical Probability Theory [20]. Many imprecise probability theories have been developed in the literature. Examples are *credal sets*, *probability intervals*, *Choquet capacities*, and *lower and upper probabilities*. These theories are described in detail in [21].

So far, few methods for Imprecise Classification have been proposed. The first one was the *Naive Credal Classifier* (NCC) [19, 22]. It utilizes the Imprecise Dirichlet Model (IDM) [23] combined with the Naive assumption (given the class variable, all features are independent) to provide imprecise predictions. Afterwards, the first Imprecise Classification algorithm based on Decision Trees was introduced by Abellán and Masegosa [20]. It was called Imprecise Credal Decision Tree (ICDT). That method employs uncertainty measures on credal sets (closed and convex sets of probability distributions) for building the tree. At leaf nodes, it determines a probability interval for each class value and applies a dominance criterion on these intervals to obtain the non-dominated states set.

The ICDT algorithm was also adapted for cost-sensitive classification by Abellán and Masegosa [20]. Such an adaptation uses the same building process as the original ICDT method. At leaf nodes, it determines a risk interval for each value of the class variable considering the costs of errors. Then, it obtains the non-dominated states set via a strong dominance criterion on these risk intervals. Abellán and Masegosa [20] also adapted NCC for cost-sensitive classification. They experimentally show that the adaptation of ICDT outperforms the adaptation of NCC since the former is more informative. The mentioned adaptations are the only algorithms proposed so far for cost-sensitive Imprecise Classification.

ICDT uses the IDM in the building process and for the probability intervals at leaf nodes. The IDM assumes previous knowledge about the data via a parameter. Although the *Representation Invariance Principle* (RIP) [23] establishes that inferences should not depend on the arrangement of the data, each dataset has associated with it an optimal value of the IDM parameter in classification [24]. This shortcoming is solved with the Nonparametric Predictive Inference Model (NPI-M) [25, 26]. This model does not make prior assumptions about the data and is non-parametric. The NPI-M has obtained equivalent results to the IDM with the best choice of the parameter in both precise and imprecise classification [27, 28].

It is important to remark that the adaptation of ICDT for cost-sensitive classification proposed so far does not take the error costs into account in the procedure to build the tree; it considers, in each step of the procedure, that all instances have the same importance, regardless of their class values. This is obviously not optimal in scenarios where different classification errors lead to different costs. Hence, we consider, as in the Weighted Decision Tree algorithm for precise classification (Weighted-DT) [10], that the instances with a higher cost of error of the corresponding class value should have more weight than the instances for which the error cost of their class value is lower.

In this research, we propose a new cost-sensitive Imprecise Credal Decision Tree that employs the NPI-M and considers weights for the training instances depending on the error costs of their class values, similar to Weighted-DT. In this way, for calculating the uncertainty measures in the split criterion and determining the probability intervals for the class values at leaf nodes, the instances with a higher cost of error of their class value have more importance. We also show that the criterion used by our new cost-sensitive Imprecise Credal Decision Tree for classifying instances may be more informative than the one of the existing cost-sensitive ICDT.

An experimental analysis is carried out in this work to compare the original cost-sensitive ICDT using the IDM and the NPI-M and our proposed cost-sensitive Imprecise Credal Decision Tree. Such an experimental study highlights that the NPI-M obtains equivalent results to the IDM with the recommended value of the parameter when both models are used in the existing cost-sensitive ICDT and that the new cost-sensitive Imprecise Credal Decision Tree significantly outperforms the existing one; even though our proposed method obtains higher misclassification costs than the existing cost-sensitive ICDT, it is usually more informative and achieves a better trade-off between low cost of incorrect classifications and informative predictions.

The remainder of this paper is structured as follows: Section 2 describes the Weighted Decision Tree algorithm, the Imprecise Dirichlet Model, the Nonparametric Predictive Inference Model, and the existing cost-sensitive Imprecise Credal Decision Tree. In Section 3, our proposed cost-sensitive Imprecise Credal Decision Tree is presented. Section 4 details the experimental study carried out in this work. Conclusions and ideas for future research are given in Section 5.

2. Background

2.1. Weighted Decision Tree

In Decision Trees (DTs), each node corresponds to an attribute and has a branch for each possible value. When inserting an attribute in a node does not provide more uncertainty-based information about the class variable, or there are no more attributes to insert, a *leaf* or *terminal* node is reached. A class value is assigned to each terminal node.

Let C be the class variable and $\{c_1, c_2, \dots, c_K\}$ its set of possible values. Let M be a non-null matrix of dimension $K \times K$ where the m_{ij} value indicates the cost of predicting c_i for a new instance when the real class value is $c_j, \quad \forall i, j \in \{1, 2, \dots, K\}$. It always holds that $m_{jj} = 0, \quad \forall j = 1, 2, \dots, K$.

Weighted Decision Tree (Weight-DT) [10] estimates the cost of incorrectly classifying an instance whose true class value is $c_j, \quad \forall j = 1, 2, \dots, K$. The following cost estimate tends to be used [29]:

$$Cost(j) = \sum_{i=1}^{K} m_{ij}, \quad \forall j = 1, 2, \dots, K. \tag{1}$$

Using these costs, Weight-DT computes weights for the training instances depending on their class values. Specifically, the weight of a training instance whose real class value is c_j is computed as follows:

$$w_j = Cost(j) \times \frac{N_{train}}{\sum_{i=1}^{K} n_{train}(c_i) \times Cost(i)}, \quad \forall j = 1, 2, \dots, K, \tag{2}$$

where N_{train} is the total number of instances in the training set and $n_{train}(c_i)$ is the number of training instances for which $C = c_i, \quad \forall i = 1, 2, \dots, K$. We may note that the sum of all instance weights is equal to $\sum_{j=1}^{K} w_j \times n_{train}(c_j) = N_{train}$.

When building a DT, the most important point might be the *split criterion*, i.e. the criterion employed for selecting the attribute to split in each node. Such a criterion is normally based on an uncertainty measure of the class variable in the corresponding node. In classical probability theory, the Shannon entropy [30] is a well-established uncertainty measure. It is defined in the following way:

$$H(C) = -\sum_{j=1}^{K} p(c_j) \log_2(p(c_j)), \tag{3}$$

where $p(c_j)$ is the probability that $C = c_j$ in the corresponding node, $\forall j = 1, 2, \ldots, K$.

Classical DTs often estimate such probabilities by using relative frequencies. In contrast, Weighted-DT estimates these probabilities through proportions of instance weights.

Formally, let $n(c_j)$ denote the number of instances in a certain node for which $C = c_j$ and W_j the sum of weights of such instances:

$$W_j = n(c_j) \times w_j, \quad \forall j = 1, 2, \ldots, K. \tag{4}$$

Let W be the total sum of weights in the node:

$$W = \sum_{i=1}^{K} n(c_i) \times w_i. \tag{5}$$

The probability that $C = c_j$ in the node is estimated as follows:

$$p(c_j) = \frac{W_j}{W}, \quad \forall j = 1, 2, \ldots, K. \tag{6}$$

For classifying a new instance with Weighted-DT, a path from the root node to a leaf node is made using its attribute values. The predicted class value is the one with the highest probability at that leaf node according to Equation (6). Consequently, at that terminal node, the instances whose class value has a higher misclassification cost have more importance.

2.2. Imprecise probabilities models

In this subsection, we describe two mathematical models based on imprecise probabilities that are useful to make inferences about the probability distribution of a discrete variable. Let X be an attribute that takes values in $\{x_1, x_2, \ldots, x_t\}$. Suppose that we have a sample of N independent and identically distributed outcomes of X.

2.2.1. The Imprecise Dirichlet Model

According to the Imprecise Dirichlet Model (IDM) [23], the probability that X takes its possible value x_i belongs to the following interval:

$$\mathcal{I}_i = \left[\frac{n(x_i)}{N+s}, \frac{n(x_i)+s}{N+s}\right], \tag{7}$$

where $n(x_i)$ is the number of ocurrences of x_i in the sample, $\forall i = 1, 2, \ldots, t$, and $s > 0$ a given parameter of the model.

The set of probability intervals given in Equation (7) is coherent and has the following credal set (closed and convex set of probability distributions) on C associated with it [31]:

$$\mathcal{P}^{IDM}(X) = \left\{ p \in \mathcal{P}(X) \mid \frac{n(x_i)}{N+s} \leq p(x_i) \leq \frac{n(x_i)+s}{N+s}, \quad \forall i = 1, 2, \ldots, t \right\}, \tag{8}$$

$\mathcal{P}(X)$ being the set of all probability distributions on X.

The selection of the s parameter is a crucial issue. We may note that IDM intervals are wider as the s value is higher. The s parameter indicates the estimated imprecision degree in the data. In [23], two values are suggested: $s = 1$ and $s = 2$, and the value $s = 1$ is recommended, but it is not explained.

2.2.2. The Nonparametric Predictive Inference Model

The Nonparametric Predictive Inference Model (NPI-M) [25, 26] represents the data via a *probability wheel.* On it, a line from the center of the wheel to its boundary is used for representing each outcome. The wheel is partitioned into N equally sized slices. Each value of X can only be represented by a unique sector of the wheel. Thus, lines associated with the same value must be positioned next to each other on the wheel. The basis of the NPI-M is the circular-$\mathcal{A}(N)$ assumption [26], which establishes that the next observation falls into any slice with equal probability $\frac{1}{N}$. It must be decided which value of X represents each slice. If a slice is bordered by two lines representing the same value, then such a value must be assigned to that slice. When two lines representing distinct values border to a slice, any of these two values, or anyone not observed yet, can be assigned to that slice.

Given a subset $A \subseteq \{x_1, x_2, \ldots, x_t\}$, the NPI-M obtains a probability interval for A, where the lower (upper) bound is determined by the minimum (maximum) proportion of slices that can be assigned to a value belonging to A, among all possible configurations of the wheel.

The following example [26] illustrates the idea of the NPI-M:

Example 1. *Let us assume that we have a discrete attribute, called Color, whose set of possible values is* {*Yellow (Y), Blue (B), White (W), Green (G), Red (R), Other (O)*}.

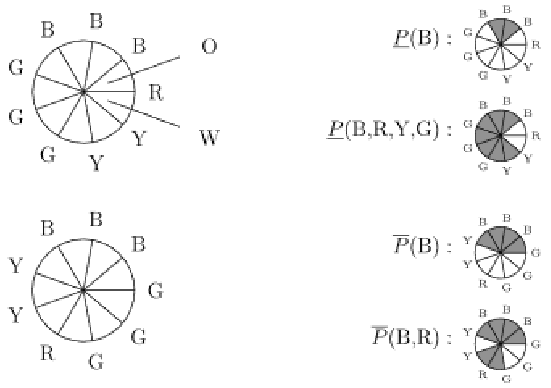

Figure 1: Two possible configurations of the probability wheel in Example 1.

Suppose that there are $N = 9$ observations about Color, arranged in the following way:

$$Y - 2, \quad B - 3, \quad W - 0, \quad G - 3, \quad R - 1, \quad O - 0.$$

Figure 1 allows us to see two possible configurations of the probability wheel. In both of them, lines associated with the same color are positioned next to each other.

In the first configuration, the slice bordered by a line that represents B and another one corresponding to R is assigned to O (non-observed), the one bordered by lines representing G and B is assigned to G, and the one bordered by lines corresponding to Y and R is assigned to W (non-observed).

In the second configuration, the slices bordered by a line representing R are assigned to R. The same occurs with B.

We may deduce that the first configuration is appropriate to obtain the lower probability of $\{B\}$ and $\{B, R, Y, G\}$, and the second configuration is suitable for the upper probability of $\{B\}$ and $\{B, R\}$.

Let $n(x_i)$ denote the number of ocurrences of x_i in the sample, $\forall i = 1, 2, \ldots, t$. For singletons $\{x_i\}$, the NPI-M lower and upper probabilities are determined as follows:

$$\underline{P}\left(\{x_i\}\right) = \max\left(\frac{n(x_i) - 1}{N}, 0\right), \quad \forall i = 1, 2, \ldots, t,$$

$$\overline{P}\left(\{x_i\}\right) = \min\left(\frac{n(x_i) + 1}{N}, 1\right), \quad \forall i = 1, 2, \ldots, t.$$

In consequence, we have the following set of NPI-M probability intervals for singletons:

$$\left\{ \left[\max\left(\frac{n(x_i)-1}{N}, 0 \right), \min\left(\frac{n(x_i)+1}{N}, 1 \right) \right], \quad i = 1, 2, \ldots, t \right\}. \tag{9}$$

This set of probability intervals is coherent and gives rise to a credal set on X [32]. Furthermore, the NPI-M lower and upper probabilities for each $A \subseteq \{x_1, x_2, \ldots, x_t\}$ can be obtained from these intervals [32]. Nonetheless, some probability distributions belonging to the credal set corresponding to these intervals may not be consistent with the NPI-M. In fact, the set of probability distributions compatible with the NPI-M is not convex [32].

By considering all the probability distributions compatible with the intervals given in Equation (9), an approximate model, called Approximate Nonparametric Predictive Inference Model (A-NPI-M), is obtained [32]. It corresponds to the credal set associated with such intervals, i.e. the convex hull of the set of probability distributions consistent with the NPI-M. Hence, the A-NPI-M avoids considerable constraints of the exact model, notably simplifying it. The NPI-M and the A-NPI-M have a similar behavior when they are utilized in classification [27]. For these reasons, we use the A-NPI-M in this work.

2.3. Cost-sensitive Imprecise Credal Decision Tree

The Imprecise Credal Decision Tree algorithm (ICDT), proposed by Abellán and Masegosa [20], consists of an adaptation of the Credal Decision Tree method (CDT) [33] for Imprecise Classification. Both algorithms use the same split criterion. In addition, the adaptation of ICDT for cost-sensitive classification (CS-ICDT), also introduced by Abellán and Masegosa [20], employs the same split criterion as the original ICDT algorithm.

Let $\mathcal{D}$ denote a subset of the training set in a certain node. Let C be the class variable and $\{c_1, c_2, \ldots, c_K\}$ its set of possible values. Let $\mathcal{P}^{\mathcal{D}}(C)$ denote the credal set on C corresponding to $\mathcal{D}$, obtained via an imprecise probabilities model $\mathcal{M}$. Let X be an attribute whose set of possible values is $\{x_1, x_2, \ldots, x_t\}$.

The basis of the split criterion of the ICDT method is the maximum entropy on $\mathcal{P}^{\mathcal{D}}(C)$ [34]:

$$H^*\left(\mathcal{P}^{\mathcal{D}}(C)\right) = \max_{p \in \mathcal{P}^{\mathcal{D}}(C)} H(p), \tag{10}$$

where $H(p)$ is the Shannon entropy of the probability distribution p.

The split criterion used in ICDT is called the Imprecise Information Gain (IIG) [33]. It is defined in the following way:

$$IIG(C,X) = H^*\left(\mathcal{P}^{\mathcal{D}}\left(C\right)\right) - \sum_{i=1}^{t} P^{\mathcal{D}}(X = x_i) \times H^*\left(\mathcal{P}^{\mathcal{D}}\left(C \mid X = x_i\right)\right), \quad (11)$$

where $P^{\mathcal{D}}(X = x_i)$ is the probability that $X = x_i$ on $\mathcal{D}$, estimated via relative frequencies, and $H^*\left(\mathcal{P}^{\mathcal{D}}\left(C \mid X = x_i\right)\right)$ is the maximum entropy on the credal set on C on the partition of $\mathcal{D}$ composed of those instances for which $X = x_i, \quad \forall i = 1, 2, \ldots, t$. Such a credal set is also obtained with $\mathcal{M}$.

The original ICDT algorithm and its adaptation for cost-sensitive classification (CS-ICDT) differ on the criterion utilized to classify new instances. Let M be the matrix of error costs defined in Section 2.1.

For classifying a new instance at a leaf node, CS-ICDT uses a criterion based on the Bayes decision rule proposed by Duda and Hart [35]. According to that rule, the class value with the lowest expected posterior risk is selected, i.e, the class value c_t such that

$$c_t = \arg \min_{i=1,2,\ldots,K} R(c_i), \quad (12)$$

where

$$R(c_i) = \sum_{j=1}^{K} m_{ij} \times p(c_j), \quad \forall i = 1, 2, \ldots, K, \quad (13)$$

$p(c_j)$ being the probability that $C = c_j, \quad \forall j = 1, 2, \ldots, K$.

Let $\underline{P}(c_j)$ and $\overline{P}(c_j)$ be, respectively, the lower and upper probabilities of the c_j value at that leaf node, which are computed via $\mathcal{M}$. From these lower and upper probabilities, CS-ICDT calculates, for each class value, the lower and upper risk as follows:

$$\underline{R}(c_i) = \sum_{j=1}^{K} m_{ij}\underline{P}(c_j), \quad \overline{R}(c_i) = \sum_{j=1}^{K} m_{ij}\overline{P}(c_j), \quad \forall i = 1, 2, \ldots, K. \quad (14)$$

For determining the non-dominated states set, CS-ICDT uses a *strong dominance criterion* on these risk intervals, according to which a class value c_j dominates another one c_i if, and only if, $\overline{R}(c_j) \leq \underline{R}(c_i)$,
$\forall i, j \in \{1, 2, \ldots, K\}$. Such a criterion is the most utilized in the literature

[36]. Therefore, the non-dominated states set predicted by CS-ICDT at that leaf node is determined in the following way:

$$\left\{c_i \mid \underline{R}(c_i) < \overline{R}(c_j) \quad \forall j \in \{1,2,\ldots,K\} \setminus \{i\}\right\}. \tag{15}$$

3. The new cost-sensitive Imprecise Credal Decision Tree

Our proposed cost-sensitive Imprecise Credal Decision Tree combines the idea of weighing instances of the existing Weighted-DT for precise classification, exposed in Section 2.1, with the A-NPI-M. We call our proposed method Weighted Imprecise Credal Decision Tree (Weighted-ICDT).

Let C be the class variable and $\{c_1, c_2, \ldots, c_K\}$ its set of possible values. Let M be the matrix of error costs defined in Section 2.1. Let N_{train} denote the number of training instances and $n_{train}(c_j)$ the number of training instances that satisfy $C = c_j, \quad \forall j = 1,2,\ldots,K$. Let us consider the A-NPI-M lower and upper probabilities for each value of the class variable in the training set:

$$\begin{aligned} l_j &= \max\left(\frac{n_{train}(c_j)-1}{N_{train}}, 0\right), \quad \forall j = 1,2,\ldots,K, \\ u_j &= \min\left(\frac{n_{train}(c_j)+1}{N_{train}}, 1\right), \quad \forall j = 1,2,\ldots,K. \end{aligned}$$

We have the following set of A-NPI-M probability intervals:

$$\mathcal{I}_C = \{[l_j, u_j], \quad j = 1,2,\ldots,K\}. \tag{16}$$

The following credal set corresponds to the intervals given by Equation (16):

$$\mathcal{P}(\mathcal{I}_C) = \{p \in \mathcal{P}(C) \mid l_j \leq p(c_j) \leq u_j, \quad \forall j = 1,2,\ldots,K\}, \tag{17}$$

$\mathcal{P}(C)$ being the set of all probability distributions on C. On this credal set, uncertainty measures can be applied. The maximum entropy [34] is a well-established uncertainty measure on credal sets that verifies good properties [21].

Thus, we consider the arrangement of the training instances $(\hat{n}_{train}(c_1), \hat{n}_{train}(c_2), \ldots, \hat{n}_{train}(c_K))$ for which the probability distribution that reaches the maximum entropy on $\mathcal{P}(\mathcal{I}_C)$ is attained. If

$(\hat{p}_{train}(c_1), \hat{p}_{train}(c_2), \ldots, \hat{p}_{train}(c_K))$ is the probability distribution that obtains the maximum entropy on $\mathcal{P}(\mathcal{I}_C)$, then $\hat{n}_{train}(c_j) = N_{train} \times \hat{p}_{train}(c_j)$ $\forall j = 1, 2, \ldots, K$. In order to compute such an arrangement, we use the algorithm proposed in [37] for obtaining the probability distribution that reaches the maximum entropy value on an A-NPI-M credal set.

Let $K_{train}(r)$ be the number of class values for which there are r training instances associated with them:

$$K_{train}(r) = |\{c_j \mid n_{train}(c_j) = r, \quad 1 \leq j \leq K\}| . \tag{18}$$

Algorithm 1 lets us obtain the arrangement that gives rise to the maximum entropy value on $\mathcal{P}(\mathcal{I}_C)$. In such an algorithm, firstly, the arrangement that attains the A-NPI-M lower probability is computed (Lines 2-9); the value mass, computed in Line 10, indicates the remaining mass to distribute among the class values. The class values that have only one observation and the ones non-observed have the lowest A-NPI-M lower probability. If the number of values of the class variable with one or zero observations is lower or equal than mass, then the mass value is equally distributed between such values (Lines 21-29). Otherwise, we sum 1 to each value with one or zero observations, making the corresponding updates in the mass value (Lines 13-20). Then, the non-observed values have the arrangement that yields the A-NPI-M upper probability and, between the rest of the class values, the ones with one or two observations have the lowest probability. This process is iteratively repeated until the mass is entirely distributed among the class values (Lines 12-30).

The proposed Weighted-ICDT method considers weights for the instances using the error costs, as Weighted-DT. However, while Weighted-DT uses the relative frequencies in the training set, Weighted-ICDT employs the arrangement that gives rise to the maximum entropy on $\mathcal{P}(\mathcal{I}_C)$.

Formally, Weighted-ICDT computes the weight of a training instance with true class value c_j via the following formula:

$$w_j = Cost(j) \times \frac{N_{train}}{\sum_{i=1}^{K} \hat{n}_{train}(c_i) \times Cost(i)}, \tag{19}$$

where $Cost(j)$ is the cost of misclassifying an instance whose real class value is c_j, calculated by Equation (1), $\forall j = 1, 2, \ldots, K$.

Let $\mathcal{D}$ denote the subset of the training set associated with a certain node, N the number of instances in $\mathcal{D}$, and $n(c_j)$ the number of instances in $\mathcal{D}$ for

```
Procedure **Determine arrangement of maximum entropy on $\mathcal{P}(\mathcal{I}_C)$**(frequencies of the class values in the training set $(n_{train}(c_1), n_{train}(c_2), \ldots, n_{train}(c_K))$)
for $j = 1$ to $K$ do
    if $n_{train}(c_j) \leq 1$ then
        $\hat{n}_{train}(c_j) = 0$
    end
    else
        $\hat{n}_{train}(c_j) = n_{train}(c_j) - 1$
    end
end
$mass = |\{c_j \mid n_{train}(c_j) > 0, \quad 1 \leq j \leq K\}|$
$r = 0$
while $mass > 0$ do
    if $(K_{train}(r) + K_{train}(r+1)) < mass$ then
        for $j = 1$ to $K$ do
            if $n_{train}(c_j) = r$ or $n_{train}(c_j) = r + 1$ then
                $\hat{n}_{train}(c_j) = \hat{n}_{train}(c_j) + 1$
                $mass = mass - 1$
            end
        end
    end
    else
        for $j = 1$ to $K$ do
            if $n_{train}(c_j) = r$ or $n_{train}(c_j) = r + 1$ then
                $\hat{n}_{train}(c_j) = \hat{n}_{train}(c_j) + \frac{mass}{K_{train}(r)+K_{train}(r+1)}$
            end
        end
        $mass = 0$
    end
    $r = r + 1$
end
return $(\hat{n}_{train}(c_1), \hat{n}_{train}(c_2), \ldots, \hat{n}_{train}(c_K))$
```

Algorithm 1: Procedure to obtain the arrangement that attains the maximum entropy on $\mathcal{P}(\mathcal{I}_C)$.

which $C = c_j, \quad \forall j = 1, 2, \ldots, K$. For the split criterion, Weighted-ICDT considers the A-NPI-M credal set on C corresponding to $\mathcal{D}$:

$$\mathcal{P}^{\mathcal{D}}\left(C\right) = \left\{p \in \mathcal{P}\left(C\right) \mid l_j^{\mathcal{D}} \leq p(c_j) \leq u_j^{\mathcal{D}}, \quad \forall j = 1, 2, \ldots, K\right\}, \tag{20}$$

where $l_j^{\mathcal{D}} = \max\left(\frac{n(c_j)-1}{N}, 0\right)$ and $u_j^{\mathcal{D}} = \min\left(\frac{n(c_j)+1}{N}, 1\right), \quad \forall j = 1, 2, \ldots, K$.

Let $(\hat{n}(c_1), \hat{n}(c_2), \ldots, \hat{n}(c_K))$ be the arrangement of the class values in the node that gives rise to the maximum entropy on $\mathcal{P}^{\mathcal{D}}\left(C\right)$. That arrangement is obtained via a similar procedure to the one given in Algorithm 1.

Then, Weighted-ICDT estimates the probability of each class value in that node through a weighted proportion of instances, as Weighted-DT. Nevertheless, Weighted-ICDT uses the arrangement that reaches the maximum entropy with the A-NPI-M, unlike Weighted-DT, which employs the relative frequencies in the node. So, the probability of the c_j value estimated by Weighted-ICDT in that node is given by:

$$\hat{p}(c_j) = \frac{w_j \times \hat{n}(c_j)}{\sum_{i=1}^{K} w_i \times \hat{n}(c_i)}, \quad \forall j = 1, 2, \ldots, K. \tag{21}$$

In this way, with Weighted-ICDT, the uncertainty about the class variable in that node is calculated via the Shannon entropy of the probability distribution $\hat{p}$, determined according to Equation (21):

$$\hat{H}^{\mathcal{D}}\left(C\right) = -\sum_{j=1}^{K} \hat{p}(c_j) \log_2 \hat{p}(c_j). \tag{22}$$

Let X be an attribute that takes values in $\{x_1, x_2, \ldots, x_t\}$. The split criterion of Weighted-ICDT is called the *Weighted Information Gain* (*WIG*). It is based on the entropy defined in Equation (22) and is given by:

$$WIG^{\mathcal{D}}(C, X) = \hat{H}^{\mathcal{D}}\left(C\right) - \sum_{i=1}^{t} \hat{P}^{\mathcal{D}}(X = x_i) \times \hat{H}^{\mathcal{D}}\left(C \mid X = x_i\right), \tag{23}$$

where $\hat{H}^{\mathcal{D}}\left(C \mid X = x_i\right)$ is the entropy of C on the subset of $\mathcal{D}$ composed of those instances for which $X = x_i$ and $\hat{P}^{\mathcal{D}}(X = x_i)$ the probability that $X = x_i$ on $\mathcal{D}$, estimated via proportion of weights:

$$\hat{P}^{\mathcal{D}}(X = x_i) = \frac{\sum_{j=1}^{K} n(x_i, c_j) \times w_j}{\sum_{j=1}^{K} n(c_j) \times w_j}, \quad \forall i = 1, 2, \ldots, t, \tag{24}$$

$n(x_i, c_j)$ being the number of instances in $\mathcal{D}$ that satisfy $X = x_i$ and $C = c_j, \quad \forall i = 1, 2, \ldots, t, \; j = 1, 2, \ldots, K$.

Once the instance weights are computed by means of Equation (19), the building process of the proposed Weighted-ICDT method can be summarized in Algorithm 2.

```
1  Procedure **Build Weighted-ICDT**(Node $N_0$, set of attributes $\mathcal{L}$)
2  if $\mathcal{L} = \emptyset$ then
3      Exit
4  end
5  Let $\mathcal{D}$ be the dataset corresponding to $N_0$
6  $\alpha = \max_{X \in \mathcal{L}} \left\{ WIG^{\mathcal{D}}(C, X) \right\}$
7  if $\alpha \leq 0$ then
8      Exit
9  end
10 Let $X_l$ the attribute for which $\alpha$ is obtained
11 Assign $X_l$ to $N_0$
12 for $x_l$ possible value of $X_l$ do
13     Add node $N_{l0}$ child of $N_0$
14     Call Build Weighted-ICDT($N_{l0}$, $\mathcal{L} \setminus \{X_l\}$)
15 end
```

Algorithm 2: Pseudo-code of building process of Weighted-ICDT.

For classifying a new instance at a leaf node, Weighted-ICDT computes, for each value of the class variable, a probability interval based on the A-NPI-M lower and upper probabilities that also takes the weight of the class value into account.

Formally, let $n(c_j)$ be the frequency of the c_j value at the leaf node, $\forall j = 1, 2, \ldots, K$. We know that the A-NPI-M lower and upper probabilities of c_j at that leaf node are given by:

$$\underline{P}(c_j) = \max\left(\frac{n(c_j) - 1}{N}, 0\right), \quad \overline{P}(c_j) = \min\left(\frac{n(c_j) + 1}{N}, 1\right), \quad \forall j = 1, \ldots, K, \tag{25}$$

where N is the number of instances in the terminal node.

Weighted-ICDT considers, for the lower (upper) probability, the proportion of weights in an arrangement of the class values for which the A-NPI-M lower (upper) probability is attained. Hence, at that leaf node, we have the following probability interval for each class value:

$$\left[\max\left(\frac{w_j \times (n(c_j)-1)}{W}, 0\right), \min\left(\frac{w_j \times (n(c_j)+1)}{W}, 1\right)\right], \forall j = 1, 2, \ldots, K, \tag{26}$$

W being the sum of all weights at that leaf node, i.e, $W = \sum_{i=1}^{K} w_i \times n(c_i)$.

Then, a dominance criterion is applied to these probability intervals to obtain the set of non-dominated states. Our proposed Weighted-ICDT algorithm utilizes a strong dominance criterion, known as *stochastic dominance* [19], on these intervals. According to that criterion, a class value c_j dominates another one c_i if, and only if,

$$\begin{aligned}
&\max\left(\frac{w_j \times (n(c_j)-1)}{W}, 0\right) \geq \min\left(\frac{w_i \times (n(c_i)+1)}{W}, 1\right) \Leftrightarrow \\
&\frac{w_j \times (n(c_j)-1)}{W} \geq \frac{w_i \times (n(c_i)+1)}{W} \Leftrightarrow \\
&w_j \times (n(c_j)-1) \geq w_i \times (n(c_i)+1), \quad \forall i, j \in \{1, 2, \ldots, K\}.
\end{aligned}$$

Consequently, the non-dominated states set predicted by Weighted-ICDT at that leaf node is determined as follows:

$$\{c_i,\ 1 \leq i \leq K \mid w_i \times (n(c_i)+1) > w_j \times (n(c_j)-1), \quad \forall j = 1, 2, \ldots, K\}. \tag{27}$$

Algorithm 3 summarizes the procedure employed by Weighted-ICDT to determine the non-dominated states set at a leaf node.

```
1 Procedure **Classify Weighted-ICDT**(Leaf node $N_0$)
2 for $j = 1$ to $K$ do
3     Compute a probability interval $[\underline{P}(c_j), \overline{P}(c_j)]$ via Equation (25)
4 end
5 Determine the non-dominated states set at $N_0$ by means of Equation (27).
```

Algorithm 3: Procedure to determine the non-dominated states set at a leaf node with Weighted-ICDT.

3.1. Justification of our proposal

The most relevant issues of our proposed Weighted-ICDT method can be summarized in the following way:

- Similar to Weighted-DT, Weighted-ICDT computes a weight for each instance depending on the cost of misclassifying the corresponding class value. Both methods estimate the costs in the same way. Nevertheless, whereas Weighted-DT estimates the instance weights based on such costs by using the class frequencies in the training set, Weighted-ICDT utilizes the arrangement that reaches the maximum entropy on the A-NPI-M credal set. Therefore, unlike Weighted-DT, Weighted-ICDT considers that the training set is not totally reliable and employs the well-established uncertainty measure on credal sets.

- In the split criterion, the existing CS-ICDT method considers that all instances have the same importance, regardless of their class values. It is oriented to minimize the number of classification errors but it neglects varying costs of errors. In contrast, for computing the uncertainty of the class variable in a certain node, Weighted-ICDT considers the proportion of instance weights for each class value. In this way, in Weighted-ICDT, the instances whose class value has a higher cost of misclassification have more importance. For example, suppose that we have two class values, c_1 and c_2, where the cost of erroneously classifying an instance with real class value c_1 is ten times the cost of misclassifying an instance whose real class value is c_2. Suppose that, in a certain node, there are four instances whose true class value is c_1 and another four instances with real class value c_2. In this case, for the split criterion, CS-ICDT considers that there is total uncertainty about the class variable, while our proposed Weighted-ICDT algorithm estimates that, in that node, the uncertainty of the class variable is considerably low. So, the uncertainty value estimated by our proposal is intuitively far more reasonable than the one estimated by CS-ICDT because, if that node were terminal, it would be quite logical to predict c_1.

- Indeed, Weight-DT also considers that the importance of an instance to calculate the uncertainty value depends on the error cost of the associated class value. However, for estimating the probability of each class value, Weighted-ICDT employs the arrangement that obtains the maximum entropy with the A-NPI-M, whereas Weighted-DT uses the arrangement associated with relative frequencies. In consequence, unlike Weighted-DT, Weighted-ICDT considers that the dataset in a certain node is not totally reliable and employs the well-established uncertainty measure on the corresponding A-NPI-M credal set.

- To classify an instance at a leaf node, our proposed Weighted-ICDT method considers, for each class value, a probability interval that depends on the frequency of that class value in that terminal node and the cost of incorrectly classifying an instance that has such a class value. Consequently, as in the split criterion, the instances whose class value has a higher cost of erroneous classification have more importance. In contrast, CS-ICDT calculates the lower and upper probabilities for each class value by considering that all instances have the same weight. Afterwards, it computes a risk interval for each class value in which the lower (upper) risk is calculated taking into account the lower (upper) probabilities of the remaining class values and the costs of predicting that class value when the real class value is another one. Thereby, the lower and upper probabilities of the corresponding class value do not directly influence the computation of the risk interval, and the cost of misclassifying an instance with that class value is also not taken into account. Concerning the dominance criterion on the probability intervals at leaf nodes, as CS-ICDT, Weighted-ICDT uses the stochastic dominance criterion, the strongest dominance criterion on a given set of probability intervals, and the one well-established in these cases [36].

  For these reasons, the risk intervals computed by CS-ICDT might be generally less informative than the probability intervals computed by Weighted-ICDT. This issue is illustrated in Example 2.

- The original CS-ICDT method, proposed in [20], uses the IDM for the uncertainty measures in the split criterion and for the lower and upper probabilities at leaf nodes. In contrast, our proposed Weighted-ICDT algorithm utilizes the A-NPI-M for estimating the instance weights, in the split criterion, and for the probability intervals at leaf nodes. The A-NPI-M is a more appropriate model than the IDM as it does not assumes previous knowledge about the data via a parameter.

Example 2. *Suppose that we have a training set of $N_{train} = 150$ instances. Let C the class variable and $\{c_1, c_2, c_3\}$ its possible values. Let M be the matrix of error costs where $m_{ii} = 0 \quad \forall i = 1, 2, 3, \quad m_{i1} = 1$ for $i = 2, 3$, $m_{i2} = 2$ for $i = 1, 3$, and $m_{i3} = 3$ for $i = 1, 2$.*

The costs of misclassifying each class value are given by:

$$Cost(1) = m_{21} + m_{31} = 2,$$

$$Cost(2) = m_{12} + m_{32} = 4,$$

$$Cost(3) = m_{13} + m_{23} = 6.$$

Let us assume the following class frequencies in the training set: $n_{train}(c_1) = n_{train}(c_2) = n_{train}(c_3) = 50$.

In this case, the arrangement that attains the maximum entropy with the A-NPI-M coincides with the one corresponding to relative frequencies. Therefore, the instance weights computed by Weighted-ICDT for the class values are the following ones:

$$w_1 = Cost(1) \times \frac{N_{train}}{\sum_{i=1}^{3} Cost(i) \times n_{train}(c_i)} = \frac{2 \times 150}{100 + 200 + 300} = 0.5,$$

$$w_2 = Cost(2) \times \frac{N_{train}}{\sum_{i=1}^{3} Cost(i) \times n_{train}(c_i)} = \frac{4 \times 150}{100 + 200 + 300} = 1,$$

$$w_3 = Cost(3) \times \frac{N_{train}}{\sum_{i=1}^{3} Cost(i) \times n_{train}(c_i)} = \frac{6 \times 150}{100 + 200 + 300} = 1.5.$$

Suppose that, at a certain leaf node, $n(c_1) = n(c_2) = n(c_3) = 3$. In such a case, according to the A-NPI-M, $\underline{P}(c_i) = \frac{2}{9}$ and $\overline{P}(c_i) = \frac{4}{9}$, for $i = 1, 2, 3$. The risk intervals determined by CS-ICDT for the class values are given by:

$$\underline{R}(c_1) = \underline{P}(c_2)m_{12} + \underline{P}(c_3)m_{13} = \frac{10}{9}, \quad \overline{R}(c_1) = \overline{P}(c_2)m_{12} + \overline{P}(c_3)m_{13} = \frac{20}{9},$$

$$\underline{R}(c_2) = \underline{P}(c_1)m_{21} + \underline{P}(c_3)m_{23} = \frac{8}{9}, \quad \overline{R}(c_2) = \overline{P}(c_1)m_{21} + \overline{P}(c_3)m_{23} = \frac{16}{9},$$

$$\underline{R}(c_3) = \underline{P}(c_1)m_{31} + \underline{P}(c_2)m_{32} = \frac{6}{9}, \quad \overline{R}(c_3) = \overline{P}(c_1)m_{31} + \overline{P}(c_2)m_{32} = \frac{12}{9}.$$

In consequence, $\overline{R}(c_i) > \underline{R}(c_j) \quad \forall i, j \in \{1, 2, 3\}$ and, thus, any of the class values is dominated under the stochastic dominance criterion on these risk intervals.

Regarding Weighted-ICDT, it holds that:

$$w_1 \times (n(c_1) - 1) = 0.5 \times 2 = 1, \quad w_1 \times (n(c_1) + 1) = 0.5 \times 4 = 2,$$

$$w_2 \times (n(c_2) - 1) = 1 \times 2 = 2, \quad w_2 \times (n(c_2) + 1) = 1 \times 4 = 4,$$

$$w_3 \times (n(c_3) - 1) = 1.5 \times 2 = 3, \quad w_3 \times (n(c_3) + 1) = 1.5 \times 4 = 6,$$

Thereby, according to the stochastic dominance criterion utilized in Weighted-ICDT, c_1 is dominated by both c_2 and c_3. The non-dominated states set predicted by Weighted-ICDT is $\{c_2, c_3\}$.

Hence, in this situation, the prediction made by Weighted-ICDT is more informative and intuitive than the one made by CS-ICDT.

Table 1 summarizes the differences between Weighted-DT, the existing CS-ICDT, and our proposed Weighted-ICDT. It should be noted that, in the proposed Weighted-ICDT method, the weight of an instance for the split criterion depends on the error cost of its class value, unlike CS-ICDT; the criterion used by Weighted-ICDT to classify instances at leaf nodes may be more effective than the one employed by CS-ICDT because the predicted intervals are probably more informative. For these reasons, it is expected that Weighted-ICDT performs better than CS-ICDT. This point is corroborated in Section 4 with exhaustive experimentation.

Table 1: Summary of the differences between Weighted-DT, CS-ICDT, and Weighted-ICDT.

Property	Weighted-DT	CS-ICDT	Weighted-ICDT
Mathematical model	precise probabilities	IDM	A-NPI-M
Error costs in the split criterion	yes	no	yes
Criterion to classify instances	precise prediction	little informative	very informative

4. Experimental analysis

In our experimentation, we aim to compare the performance of the CS-ICDT algorithm with the IDM (CS-ICDT-IDM) and with the A-NPI-M (CS-ICDT-NPI) and our proposed Weighted-ICDT method.

4.1. Experimental setup

4.1.1. Datasets

For testing the performance of the algorithms considered in this experimentation, the 34 datasets used in the experimental analysis carried out by Abellán and Masegosa in [20] have been employed. All of them can be found in *UCI Machine Learning Repository* [38]. They are diverse in terms of size, number of values of the class variable, number of continuous and discrete features, ranges of values of discrete attributes, etc. The datasets have been selected so that they have at least three class values as, with only two possible values of the class variable, an imprecise classifier always predicts all class values or only one. Table 2 shows the most important characteristics of each dataset.

4.1.2. Procedure

Consistently with the experimental studies about Imprecise Classification algorithms carried out in [20, 39], the datasets have been preprocessed as follows: missing values have been replaced with mean values for continuous attributes and with modal values for discrete features. Afterward, the datasets have been discretized via Fayyad and Irani's discretization method [40]. The preprocessing has been applied to the training set and, then, it has been translated to the test set. The filters given in the Weka software [41] have been employed for the preprocessing.

Three algorithms have been used in this experimental analysis: CS-ICDT-IDM, CS-ICDT-NPI, and Weighted-ICDT. Remark that the computational complexity of these three methods is similar since they are Decision Trees whose split criterion is based on the maximum entropy on a mathematical model based on coherent probability intervals. We do not use more algorithms because, as explained in the introduction, CS-ICDT-IDM and the adaptation of NCC for cost-sensitive classification are the only methods for cost-sensitive Imprecise Classification proposed so far, and, since the former algorithm significantly outperforms the latter, considering the adaptation of NCC could introduce noise in the statistical comparisons.

The implementation of the original ICDT algorithm provided in Weka has been employed, and the required structures and methods for using CS-ICDT-IDM, CS-ICDT-NPI, and Weighted-ICDT have been added to this software. For CS-ICDT-IDM, the value $s = 1$, one of the values recommended

Table 2: Description of the datasets employed in our experiments. Column "N" is the number of instances, column "Attr" is the number of attributes, column "Cont" is the number of continuous features, column "Disc" is the number of discrete features, column "K" is the number of class values, and column "Range" is the range of values of the discrete attributes.

Dataset	N	Attr	Cont	Disc	K	Range
anneal	898	38	6	32	6	2-10
arrhythmia	452	279	206	73	16	2
audiology	226	69	0	69	24	2-6
autos	205	25	15	10	7	2-22
balance-scale	625	4	4	0	3	-
car	1728	6	0	6	4	3-4
cmc	1473	9	2	7	3	2-4
dermatology	366	34	1	33	6	2-4
ecoli	366	7	7	0	7	-
flags	194	30	2	28	8	2-13
hypothyroid	3772	30	7	23	4	2-4
iris	150	4	4	0	3	-
letter	20000	16	16	0	26	-
lymphography	146	18	3	15	4	2-8
mfeat-pixel	2000	240	0	240	10	4-6
nursery	12960	8	0	8	4	2-4
optdigits	5620	64	64	0	10	-
page-blocks	5473	10	10	0	5	-
pendigits	10992	16	16	0	10	-
postop-patient-data	90	9	0	9	3	2-4
primary-tumor	339	17	0	17	21	2-3
segment	2310	19	16	0	7	-
soybean	683	35	0	35	19	2-7
spectrometer	531	101	100	1	48	4
splice	3190	60	0	60	3	4-6
sponge	76	44	0	44	3	2-9
tae	151	5	3	2	3	2
vehicle	946	18	18	0	4	-
vowel	990	11	10	1	11	2
waveform	5000	40	40	0	3	-
wine	178	13	13	0	3	-
zoo	101	16	1	16	7	2

in [23], has been used, as in the experimental analysis carried out in [20][1]. Consistently with such an experimental analysis, five cost matrices have been employed, which we describe below.

Let $\sigma : \{1, 2, \ldots, K\} \to \{1, 2, \ldots, K\}$ be a permutation that yields a decreasing order of the frequencies of the class values in the training set, i.e $n_{train}\left(c_{\sigma(i)}\right) \geq n_{train}\left(c_{\sigma(j)}\right) \quad \forall 1 \leq i \leq j \leq K$. The cost matrices used in this experimentation are the following ones:

- **Cost Matrix 0/1**: The costs of all erroneous predictions are equal to 1, i.e:

$$\begin{aligned} &m_{ij} = 1 \quad \forall i, j \in \{1, 2, \ldots, K\}, \, j \neq i, \\ &m_{jj} = 0 \quad \forall j = 1, 2, \ldots, K. \end{aligned}$$

- **Cost Matrix (I)**: The cost of an incorrect prediction only depends on the real class value. The class values with lower frequencies have more cost than the ones with higher frequencies. Specifically, the cost of misclassifying an instance whose real class value is the one with the highest frequency is equal to one, the cost of misclassifying an instance whose true class value is the one with the second-highest frequency is equal to 2, and so on. Formally:

$$\begin{aligned} &m_{i\sigma(j)} = j \quad \forall i, j \in \{1, 2, \ldots, K\}, \, \sigma(j) \neq i, \\ &m_{jj} = 0 \quad \forall j = 1, 2, \ldots, K. \end{aligned}$$

- **Cost Matrix (II)**: Only the predicted class value influences the cost of an erroneous prediction. Again, the class values with lower frequencies have more cost than the ones with higher frequencies. The cost of erroneously predicting the class value with the highest frequency is equal to 1, the cost of incorrectly predicting the class value with the second-highest frequency is equal to 2, and so on:

$$\begin{aligned} &m_{\sigma(j)i} = j \quad \forall i, j \in \{1, 2, \ldots, K\}, \, \sigma(j) \neq i, \\ &m_{jj} = 0 \quad \forall j = 1, 2, \ldots, K. \end{aligned}$$

[1] Experiments have been carried out with $s = 2$, but the obtained results are always worse than with $s = 1$. So, they are not reported.

- **Cost Matrix (III)**: This cost matrix is equivalent to Cost Matrix (I), but now the class values with lower frequencies have lower costs than the class values with higher frequencies:

$$\begin{aligned}&m_{i\sigma(j)} = K - j + 1 \quad \forall i, j \in \{1, 2, \ldots, K\}, \sigma(j) \neq i,\\&m_{jj} = 0 \quad \forall j = 1, 2, \ldots, K.\end{aligned}$$

- **Cost Matrix (IV)**: It is similar to Cost Matrix (II). However, now the class values with lower frequencies have lower costs than the class values with higher frequencies:

$$\begin{aligned}&m_{\sigma(j)i} = K - j + 1 \quad \forall i, j \in \{1, 2, \ldots, K\}, \sigma(j) \neq i,\\&m_{jj} = 0 \quad \forall j = 1, 2, \ldots, K.\end{aligned}$$

For each preprocessed dataset and cost matrix, a cross-validation procedure of 10 folds has been repeated 10 times.

4.1.3. Evaluation metrics

Since all the algorithms employed in our experimental analysis consider costs of errors, an accuracy measure is not useful enough here. Instead, we use the evaluation metric proposed in [20] for imprecise classifiers that take the error costs into consideration. Such an evaluation metric, called MIC, for misclassifications, takes the maximum cost of predicting a class value belonging to the non-dominated states set into account. Also, for correct predictions, MIC considers the number of non-dominated states.

Formally, let c_{t_i} denote the real class value of the ith test instance, with $t_i \in \{1, 2, \ldots, K\}$, and U_i the predicted non-dominated states set for such an instance. For instances incorrectly classified, it is considered:

$$\alpha_{t_i} = \max_{c_j \in U_i} m_{j t_i}. \tag{28}$$

Then, MIC is defined as:

$$MIC = \frac{1}{N_{test}} \times \left(- \sum_{i:Correct} \log_2 \frac{|U_i|}{K} - \frac{1}{K-1} \times \sum_{i:Error} -\alpha_{t_i} \times \log_2 K \right), \tag{29}$$

where N_{test} is the number of test instances.

This measure penalizes the errors in a strict sense. It reaches its optimal value ($\log_2(K)$) when all predictions are correct and precise. When an imprecise classifier always predicts all class values, the MIC value is equal to 0. It makes sense since, in such a case, the classifier is not informative.

The following metrics allow separately evaluating how informative the predictions are and the costs of misclassifications:

- **Determinacy**: It indicates the proportion of test instances for which a single class value is predicted.
- **Indeterminacy size**: It measures, among the test instances for which there are two or more non-dominated states, the average number of predicted values of the class variable.
- **Single Cost**: It indicates, among the test instances precisely classified, the average cost of misclassification.
- **Set Cost**: It measures the average error cost between the test instances for which more than a class value is predicted:

$$\frac{1}{|\{1 \leq i \leq N_{test} \mid |U_i| \geq 2\}|} \times \sum_{i=1, |U_i| \geq 2 \wedge i:Error}^{N_{test}} \alpha_{t_i}, \tag{30}$$

  where α_{t_i} is given by Equation (28).

Determinacy and Indeterminacy size focus on how informative the predictions are, while Single Cost and Set Cost focus on the costs of incorrect classifications.[2]

4.1.4. Statistical evaluation

Following the recommendations given in [42] for statistical comparisons between the results obtained by three or more algorithms on many datasets, the following statistical tests have been used with a level of significance of $\alpha = 0.05$ to compare the performance of the algorithms considered here:

[2]Single Cost and Set Cost are, respectively, the adaptations of the Single Accuracy and Set Accuracy measures, proposed in [19], for cost-sensitive scenarios.

- **Friedman test** [43]: This test is non-parametric and separately ranks the algorithms for each dataset: the algorithm that achieves the best result is assigned to position 1, the one that obtains the second-best result to position 2, and so on. The null hypothesis of this test is that all algorithms obtain equivalent results.
- **Nemenyi test** [44]: It compares all algorithms pairwise when the null hypothesis of the Friedman test is rejected.

We use critical diagrams [42] to present the results of the Friedman and Nemenyi tests. A critical diagram utilizes an enumerated axis for drawing the average Friedman ranks of the methods. The algorithms are arranged so that the ones with the highest rank are placed at the right-most side. Segments are used to connect the algorithms for which there are no statistically significant differences according to the Nemenyi test.

4.2. Results and discussion

Table 3 lets us observe the average Friedman rank obtained by each algorithm for each cost matrix in MIC. The best result for each cost matrix is marked in bold. Figures 2, 3, 4, 5, and 6 show the critical diagrams corresponding to Cost Matrices 0/1, (I), (II), (III), and (IV), respectively. In Appendix A, the complete MIC results can be found.

Table 3: Average Friedman rank obtained by each algorithm in MIC for each cost matrix.

	Cost Matrix				
Algorithm	0/1	(I)	(II)	(III)	(IV)
CS-ICDT-IDM	2.2794	2.0588	2.2353	2.0882	2.5588
CS-ICDT-NPI	2.5882	2.6324	2.5441	2.4559	2.6176
Weighted-ICDT	**1.1324**	**1.3088**	**1.2206**	**1.4559**	**1.1765**

We express the following comments about these results:

- As can be seen in Table 3 and Figures 2-6, CS-ICDT-IDM obtains a lower average Friedman rank than CS-ICDT-NPI for all the cost matrices considered in this experimentation. Nevertheless, as CS-ICDT-IDM and CS-ICDT-NPI are connected via a segment in all critical diagrams (Figures 2-6), there are no statistically significant differences according to the Nemenyi test between these two algorithms for any of the five cost matrices considered. In consequence, we could say that CS-ICDT-NPI obtains statistically equivalent results to CS-ICDT-IDM.

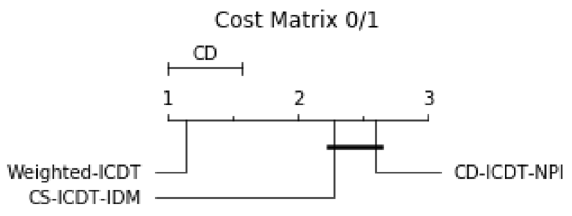

Figure 2: Critical diagram for the MIC measure with Cost Matrix 0/1. CD = Critical Distance.

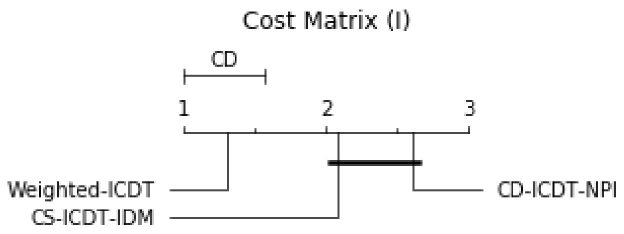

Figure 3: Critical diagram for the MIC measure with Cost Matrix (I). CD = Critical Distance.

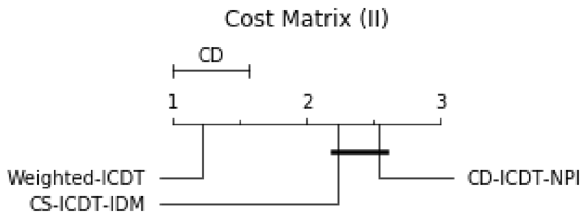

Figure 4: Critical diagram for the MIC measure with Cost Matrix (II). CD = Critical Distance.

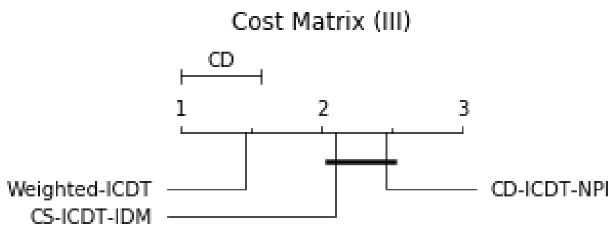

Figure 5: Critical diagram for the MIC measure with Cost Matrix (III). CD = Critical Distance.

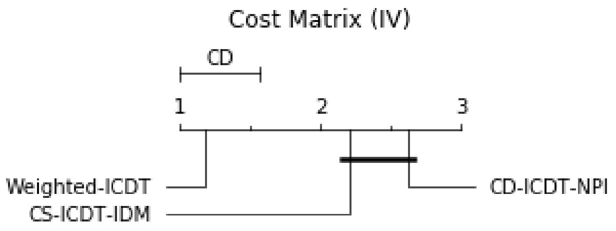

Figure 6: Critical diagram for the MIC measure with Cost Matrix (IV). CD = Critical Distance.

- Table 3 shows that, for all cost matrices, the lowest average Friedman rank is achieved by our proposed Weighted-ICDT method. In addition, as can be seen in Figures 2-6, in the critical diagrams, Weighted-ICDT is not connected with the other two algorithms via segments. Thus, according to the Nemenyi test, Weighted-ICDT performs significantly better than CS-ICDT-NPI and CS-ICDT-IDM for the five cost matrices. So, we can state that Weighted-ICDT achieves, by far, the best results.

For a deeper analysis, Table 4 presents, for each cost matrix considered in our experimental analysis, the average results obtained by each algorithm in Determinacy, Indeterminacy Size, Single Cost, and Set Cost. The best result for each measure and cost matrix is marked in bold.

These results indicate the following points for each metric:

- **Determinacy**:
  - CS-ICDT-IDM makes more precise predictions than CS-ICDT-NPI since the average Determinacy value obtained by CS-ICDT-

Table 4: Average values obtained by each algorithm in the individual evaluation metrics for each cost matrix.

		Cost Matrix				
Measure	Algorithm	0/1	(I)	(II)	(III)	(IV)
Determinacy	CS-ICDT-IDM	0.7094	0.6248	0.6457	0.7159	0.5432
	CS-ICDT-NPI	0.6835	0.6070	0.6404	0.7025	0.5068
	Weighted-ICDT	**0.9002**	**0.8083**	**0.8756**	**0.8485**	**0.8685**
Single Cost	CS-ICDT-IDM	0.0890	0.1739	0.6457	0.3625	0.1613
	CS-ICDT-NPI	**0.0806**	**0.1598**	**0.1183**	**0.3584**	**0.1345**
	Weighted-ICDT	0.1521	0.4294	0.4156	0.9636	1.0995
Indeterminacy Size	CS-ICDT-IDM	8.6956	8.7030	**3.6924**	8.4671	**3.4241**
	CS-ICDT-NPI	8.8938	9.0162	4.4448	8.6728	3.9586
	Weighted-ICDT	**7.9228**	**7.1027**	7.2984	**7.0778**	7.4300
Set Cost	CS-ICDT-IDM	**0.0041**	0.0145	0.3001	**0.0220**	0.8438
	CS-ICDT-NPI	0.0061	**0.0087**	0.2100	0.0355	0.6742
	Weighted-ICDT	0.0075	0.0993	**0.0636**	0.1074	**0.1528**

NPI is lower than the one obtained by CS-ICDT-IDM for all cost matrices.

 – Our proposed Weighted-ICDT algorithm achieves, by far, the highest average Determinacy value for all cost matrices. Hence, this method is, by far, the one that makes more precise predictions among the ones considered in this experimentation.

- **Indeterminacy Size:**

  – For imprecise predictions, CS-ICDT-NPI predicts more class values than CS-ICDT-IDM due to the average results obtained in Indeterminacy Size. Thus, it can be stated that the predictions made by CS-ICDT-IDM are more informative than the ones made by CS-ICDT-NPI.

  – The cost matrix influences the average Indeterminacy Size value of Weighted-ICDT. For Cost Matrices 0/1, (I), and (III), Weighted-ICDT obtains the lowest average Indeterminacy Size value. Therefore, for such cost matrices, the imprecise predictions made by Weighted-ICDT are the most informative ones. The opposite happens with cost matrices (II) and (IV). Nonetheless, we must remark that, with these cost matrices, the average Determinacy value obtained by Weighted-ICDT is pretty high.

- **Single Cost:**

  - CS-ICDT-NPI obtains a better result than CS-ICDT-IDM concerning the misclassification cost of precise predictions for all the cost matrices considered here.
  - Weighted-ICDT gets the highest average value of Single Cost for the five cost matrices. Thereby, it obtains the highest misclassification cost when predicting a single class value.

- **Set Cost:**
  - For Cost Matrices 0/1, (I), and (III), CS-ICDT-NPI obtains a higher average Set Cost value than CS-ICDT-IDM, while, for Cost Matrices (II) and (IV), CS-ICDT-NPI achieves the lowest average Set Cost value.
  - Weighted-ICDT obtains the highest average Set Cost value for Cost Matrices 0/1, (I), and (III). Thereby, for these cost matrices, the cost of incorrect imprecise predictions with Weighted-ICDT is higher than with the other algorithms. The contrary happens with Cost Matrices (II) and (IV).

Summary of the results: The results obtained in this experimental study can be summarized as follows:

- From a general point of view, our proposed Weighted-ICDT method performs far better than the remaining ones considering the MIC measure, the most important metric used here. We can see in Table 3 and Figures 2-6 that, for all cost matrices, the Friedman rank is notably lower for Weighted-ICDT. About the rest of the metrics that we use in the comparison (Table 4), the new proposal achieves the best results in Determinacy and Indeterminacy size. Only the CS-ICDT-NPI outperforms the rest on Single cost. However, this method obtains the worst results in Determinacy. The Size Cost value is more distributed among all methods.

Considering comparisons method versus method, it can be remarked that:

- The predictions made by CS-ICDT-NPI are less informative than the ones made by CS-ICDT-IDM. It is because, as shown in [37], given a sample of outcomes of a discrete attribute, IDM probability intervals with $s = 1$ are contained in A-NPI-M probability intervals. Since the predictions made by CS-ICDT-IDM are more precise than the ones made by CS-ICDT-NPI, the risk of misclassification is higher with the former algorithm and, therefore, the cost of incorrect classifications is generally higher with CS-ICDT-IDM.

- The results obtained in MIC allow deducing that CS-ICDT-IDM and CS-ICDT-NPI achieve an equivalent trade-off between informative predictions and low misclassification cost and, thus, they perform equivalently. It is consistent with the experimental studies carried out in [27, 28], where it was shown that the NPI-M obtains equivalent results to the IDM with the recommended value of the parameter when both models are utilized in precise and imprecise classification.

- Our proposed Weighted-ICDT method makes much more informative predictions than the other algorithms, even though it leads to a higher cost of incorrect predictions. Weighted-ICDT achieves the best trade-off between informative predictions and low misclassification costs. It is because, as argued in Section 3.1, the criterion employed by Weighted-ICDT to classify instances at leaf nodes is probably more effective than the one used by CS-ICDT as the predicted intervals may be more informative and, unlike CS-ICDT, Weighted-ICDT considers the costs of errors for the uncertainty measures in the split criterion.

- Hence, Weighted-ICDT is, by far, the algorithm that obtains the best performance among the ones considered in this experimentation.

5. Conclusions and future work

Classification errors usually have different costs in practical applications. Also, classifiers normally predict a single value of the class variable. However, in some situations, there is not enough information available to point to a unique class value. In these cases, it is more logical that classifiers predict a set of values of the class variable. This is known as Imprecise Classification. A Decision Tree for Imprecise Classification that takes the costs of errors into account, called cost-sensitive Imprecise Credal Decision Tree, was introduced a few years ago.

Based on the idea of an existing Decision Tree for cost-sensitive precise classification, in this work, we have proposed a new cost-sensitive Imprecise

Credal Decision Tree that weights the instances by taking the misclassification cost of the corresponding class value into account. This new method considers the error costs in the building process, unlike the existing cost-sensitive Imprecise Credal Decision Tree, which only considers the error costs for classifying instances at leaf nodes. Thereby, for the split criterion, an instance has more importance as the misclassification cost of the corresponding class value is higher. In this sense, our proposal presents an advantage over the existing cost-sensitive Imprecise Credal Decision Tree because the aim is to minimize the cost of incorrect classifications and not the number of erroneous predictions. Furthermore, our proposed cost-sensitive Imprecise Credal Decision Tree uses the Nonparametric Predictive Inference Model, whereas the existing one employs the Imprecise Dirichlet Model. As explained before, the former model is more suitable than the latter as it does not assume previous knowledge about the data via a parameter. We have also argued that the criterion employed by our proposed algorithm to classify instances at leaf nodes is probably more effective than the one used by the existing cost-sensitive Imprecise Credal Decision Tree since the predictions made may be more informative.

An experimental study has been carried out to check the performance of the existing cost-sensitive Imprecise Credal Decision Tree using the Imprecise Dirichlet Model and the Nonparametric Predictive Inference Model and our proposal. Such an experimental study has revealed that the Nonparametric Predictive Inference Model obtains statistically equivalent results to the Imprecise Dirichlet Model with the recommended value of the parameter when both models are utilized in the existing cost-sensitive Imprecise Credal Decision Tree and, as expected, our proposed cost-sensitive Imprecise Credal Decision Tree performs significantly better than the existing one; even though the cost of erroneous predictions of our proposed method is higher, it is far more informative and achieves a better trade-off between low misclassification cost and informative predictions.

Therefore, it can be concluded that our proposed cost-sensitive Imprecise Credal Decision Tree is more suitable than the existing one for practical applications where the misclassification costs are different and the available information is not enough for classifiers to point to a unique class value.

As future research, other Decision Trees for cost-sensitive Imprecise Classification could be developed by considering weights for the instances in a different way for the split criterion or through other criteria for classifying instances at leaf nodes. Moreover, it would be interesting to develop ensemble methods for cost-sensitive Imprecise Classification that use our proposed algorithm as the base classifier.

Acknowledgments

This work has been supported by UGR-FEDER funds under Project A-TIC-344-UGR20, by the "FEDER/Junta de Andalucía-Consejería de Transformación Económica, Industria, Conocimiento y Universidades" under Project P20_00159, and by research scholarship FPU17/02685.

Appendix A. Complete results

In this appendix, we show the complete experimental results corresponding to the MIC measure. Best results are marked in bold font.

References

[1] C. Ling, V. Sheng, Q. Yang, Test strategies for cost-sensitive decision trees, IEEE Transactions on Knowledge and Data Engineering 18 (8) (2006) 1055–1067. doi:10.1109/TKDE.2006.131.

[2] R. Santos-Rodríguez, D. García-García, J. Cid-Sueiro, Cost-sensitive classification based on bregman divergences for medical diagnosis, in: 2009 International Conference on Machine Learning and Applications, 2009, pp. 551–556. doi:10.1109/ICMLA.2009.82.

[3] Y.-J. Park, S.-H. Chun, B.-C. Kim, Cost-sensitive case-based reasoning using a genetic algorithm: Application to medical diagnosis, Artificial Intelligence in Medicine 51 (2) (2011) 133–145. doi:10.1016/j.artmed.2010.12.001.

[4] Y. Sahin, S. Bulkan, E. Duman, A cost-sensitive decision tree approach for fraud detection, Expert Systems with Applications 40 (15) (2013) 5916–5923. doi:10.1016/j.eswa.2013.05.021.

[5] S. Akila, U. Srinivasulu Reddy, Cost-sensitive risk induced bayesian inference bagging (ribib) for credit card fraud detection, Journal of Computational Science 27 (2018) 247–254. doi:10.1016/j.jocs.2018.06.009.

Table 5: MIC results with Cost Matrix 0/1.

Dataset	CS-ICDT-IDM	CS-ICDT-NPI	Weighted-ICDT
anneal	1.7305	1.7198	**1.7563**
arrhythmia	1.3775	1.4515	**1.8104**
audiology	0.7751	0.7727	**2.5217**
autos	0.9149	0.8138	**1.3259**
balance-scale	0.6540	**0.7091**	0.6433
bridges-version1	0.7745	0.7571	**1.0134**
bridges-version2	0.7615	0.6833	**0.9706**
car	1.1822	1.0974	**1.2126**
cmc	0.2661	0.2660	**0.2684**
dermatology	1.4903	1.4535	**1.6186**
ecoli	1.5292	1.5482	**1.6030**
flags	0.7185	0.7154	**1.0206**
hypothyroid	1.3679	1.3629	**1.3705**
iris	0.9976	**1.0082**	0.9992
letter	1.1264	1.1284	**2.3996**
lymphography	0.8710	0.8785	**0.9063**
mfeat-pixel	1.4248	1.4232	**1.6837**
nursery	**1.5222**	1.4813	1.5072
optdigits	1.3074	1.2797	**1.6636**
page-blocks	1.5208	1.5169	**1.5262**
pendigits	1.6942	1.6953	**1.9613**
postoperative-patient-data	0.6201	**0.6225**	**0.6225**
primary-tumor	0.0288	0.0290	**1.1697**
segment	1.6840	1.6657	**1.7878**
soybean	1.7505	1.2819	**2.6865**
spectrometer	0.0000	0.0000	**1.5663**
splice	0.9701	0.9656	**0.9739**
sponge	0.9610	0.9532	**0.9701**
tae	0.2122	0.1883	**0.2196**
vehicle	0.7871	0.7873	**0.8068**
vowel	0.6793	0.6566	**1.6183**
waveform	0.6547	0.6549	**0.6608**
wine	0.9538	0.9501	**0.9615**
zoo	1.4224	1.2370	**1.7778**
Average	1.0215	0.9928	**1.3413**

[6] S. Nami, M. Shajari, Cost-sensitive payment card fraud detection based on dynamic random forest and k-nearest neighbors, Expert Systems with Applications 110 (2018) 381–392. `doi:10.1016/j.eswa.2018.06.011`.

[7] O. F. Arar, K. Ayan, Software defect prediction using cost-sensitive neural network, Applied Soft Computing 33 (2015) 263–277. `doi:10.1016/j.asoc.2015.04.045`.

[8] M. Liu, L. Miao, D. Zhang, Two-stage cost-sensitive learning for software defect prediction, IEEE Transactions on Reliability 63 (2) (2014) 676–686. `doi:10.1109/TR.2014.2316951`.

Table 6: MIC results with Cost Matrix (I).

Dataset	CS-ICDT-IDM	CS-ICDT-NPI	Weighted-ICDT
anneal	1.6296	1.6319	**1.7335**
arrhythmia	0.6306	0.2152	**1.4780**
audiology	0.0000	0.0000	**1.9018**
autos	0.4651	0.4112	**0.9456**
balance-scale	**0.6084**	0.5887	0.6066
bridges-version1	0.4842	0.2471	**0.4899**
bridges-version2	0.4074	0.3408	**0.6166**
car	1.1128	1.0729	**1.1336**
cmc	0.1019	**0.1221**	0.0968
dermatology	1.4730	1.4277	**1.6533**
ecoli	1.3058	1.2943	**1.4408**
flags	0.2942	0.3398	**0.6423**
hypothyroid	1.3605	1.3579	**1.3621**
iris	**0.9607**	0.9581	0.9592
letter	0.9718	0.9444	**2.1057**
lymphography	0.7524	0.7513	**0.7771**
mfeat-pixel	1.2639	1.3062	**1.4642**
nursery	**1.4884**	1.4476	1.4608
optdigits	1.1364	1.1099	**1.4572**
page-blocks	1.4797	1.4784	**1.4909**
pendigits	1.5302	1.5391	**1.7828**
postoperative-patient-data	0.4492	**0.4517**	**0.4517**
primary-tumor	0.0000	0.0000	**0.3789**
segment	1.6258	1.5961	**1.7254**
soybean	0.6831	0.8058	**2.4081**
spectrometer	0.0000	0.0000	**0.7302**
splice	0.9383	0.9372	**0.9444**
sponge	**0.9112**	0.8967	0.8659
tae	-0.0813	**-0.0454**	-0.0778
vehicle	0.6105	0.5956	**0.6155**
vowel	0.5356	0.5103	**1.1891**
waveform	**0.5600**	**0.5600**	0.5432
wine	0.9174	0.9073	**0.9308**
zoo	1.3434	1.1279	**1.5987**
Average	0.8221	0.7920	**1.1148**

[9] M. J. Siers, M. Z. Islam, Software defect prediction using a cost sensitive decision forest and voting, and a potential solution to the class imbalance problem, Information Systems 51 (2015) 62–71. `doi:10.1016/j.is.2015.02.006`.

[10] K. M. Ting, An instance-weighting method to induce cost-sensitive trees, IEEE Transactions on Knowledge and Data Engineering 14 (3) (2002) 659–665. `doi:10.1109/TKDE.2002.1000348`.

[11] B. Krawczyk, M. Woniak, G. Schaefer, Cost-sensitive decision tree ensembles for effective imbalanced classification, Applied Soft Computing 14 (2014) 554–562. `doi:10.1016/j.asoc.2013.08.014`.

Table 7: MIC results with Cost Matrix (II).

Dataset	CS-ICDT-IDM	CS-ICDT-NPI	Weighted-ICDT
anneal	1.7292	1.7193	**1.7559**
arrhythmia	1.7001	1.6763	**1.7330**
audiology	1.7800	1.8532	**2.4119**
autos	1.0635	0.9493	**1.2087**
balance-scale	**0.6677**	0.6671	0.6435
bridges-version1	0.7209	0.7421	**0.8366**
bridges-version2	0.7776	0.8310	**0.8721**
car	1.1450	1.0995	**1.1717**
cmc	0.1376	**0.1482**	0.1202
dermatology	1.3722	1.3513	**1.5759**
ecoli	1.4250	1.4183	**1.5389**
flags	0.9166	**0.9373**	0.9304
hypothyroid	1.3648	1.3604	**1.3671**
iris	0.9326	0.9633	**0.9806**
letter	1.3914	1.3869	**2.1807**
lymphography	0.8753	**0.9033**	0.8835
mfeat-pixel	1.4010	1.4088	**1.5643**
nursery	1.4496	1.4058	**1.4805**
optdigits	1.2234	1.1924	**1.5468**
page-blocks	1.5112	1.5053	**1.5187**
pendigits	1.4801	1.4721	**1.8624**
postoperative-patient-data	0.6213	**0.6225**	**0.6225**
primary-tumor	0.9598	0.8751	**0.9966**
segment	1.5738	1.5517	**1.7382**
soybean	1.9445	1.9249	**2.6380**
spectrometer	1.0975	0.9911	**1.3549**
splice	0.9447	0.9424	**0.9491**
sponge	0.9620	0.9701	**0.9805**
tae	**0.0179**	0.0161	-0.0916
vehicle	0.5771	0.6009	**0.6547**
vowel	0.8123	0.6975	**1.3307**
waveform	0.5149	0.5027	**0.5372**
wine	**0.9413**	0.9242	0.9266
zoo	1.4206	1.3326	**1.6382**
Average	1.1015	1.0866	**1.2488**

[12] X. Li, H. Zhao, W. Zhu, A cost sensitive decision tree algorithm with two adaptive mechanisms, Knowledge-Based Systems 88 (2015) 24–33. `doi:10.1016/j.knosys.2015.08.012`.

[13] X. Chai, L. Deng, Q. Yang, C. Ling, Test-cost sensitive naive bayes classification, in: Fourth IEEE International Conference on Data Mining (ICDM'04), 2004, pp. 51–58. `doi:10.1109/ICDM.2004.10092`.

[14] G. M. Di Nunzio, A new decision to take for cost-sensitive naive bayes classifiers, Information Processing & Management 50 (5) (2014) 653–674. `doi:10.1016/j.ipm.2014.04.008`.

Table 8: MIC results with Cost Matrix (III).

Dataset	CS-ICDT-IDM	CS-ICDT-NPI	Weighted-ICDT
anneal	1.7519	1.7352	**1.7643**
arrhythmia	1.1872	1.2259	**1.3567**
audiology	1.5435	1.5002	**2.1397**
autos	0.7931	0.7234	**1.0508**
balance-scale	0.5773	0.5714	**0.5780**
bridges-version1	0.6646	0.6599	**0.7204**
bridges-version2	0.7026	0.7089	**0.7217**
car	1.1071	1.0459	**1.1286**
cmc	0.0506	**0.0884**	0.0733
dermatology	1.3937	1.3581	**1.5933**
ecoli	1.4030	**1.4090**	1.3952
flags	0.6304	0.6116	**0.6337**
hypothyroid	**1.3648**	1.3582	1.3637
iris	**0.9959**	0.9956	0.9885
letter	1.0018	0.9751	**1.9560**
lymphography	0.7043	0.7363	**0.7742**
mfeat-pixel	1.3190	1.3277	**1.4948**
nursery	**1.4973**	1.4505	1.4793
optdigits	1.1707	1.1426	**1.4335**
page-blocks	1.4906	1.4826	**1.4987**
pendigits	1.4873	1.4778	**1.8051**
postoperative-patient-data	0.4748	**0.4761**	0.4721
primary-tumor	0.2596	0.2076	**0.3160**
segment	1.5722	1.5542	**1.7013**
soybean	1.6754	1.7108	**2.3520**
spectrometer	0.1440	0.1535	**0.5864**
splice	0.9390	0.9390	**0.9461**
sponge	0.9610	0.9701	**0.9888**
tae	-0.0278	**0.0413**	-0.0006
vehicle	**0.6574**	0.6522	0.6498
vowel	0.5882	0.5615	**1.2123**
waveform	**0.5128**	0.5106	**0.5128**
wine	**0.9123**	0.9076	0.8547
zoo	1.3667	1.3643	**1.6409**
Average	0.9668	0.9598	**1.1230**

[15] S. Zhang, Cost-sensitive knn classification, Neurocomputing 391 (2020) 234–242. `doi:10.1016/j.neucom.2018.11.101`.

[16] D. Fuqua, T. Razzaghi, A cost-sensitive convolution neural network learning for control chart pattern recognition, Expert Systems with Applications 150 (2020) 113275. `doi:10.1016/j.eswa.2020.113275`.

[17] S. K. Biswas, M. Chakraborty, H. R. Singh, D. Devi, B. Purkayastha, A. K. Das, Hybrid case-based reasoning system by cost-sensitive neural network for classification, Soft Computing 21 (24) (2017) 7579–7596.

[18] H. Li, L. Zhang, X. Zhou, B. Huang, Cost-sensitive sequential three-way

Table 9: MIC results with Cost Matrix (IV).

Dataset	CS-ICDT-IDM	CS-ICDT-NPI	Weighted-ICDT
anneal	1.6667	1.6614	**1.7491**
arrhythmia	0.3823	0.6981	**1.2272**
audiology	1.0410	1.1103	**2.1437**
autos	0.4807	0.4357	**0.9747**
balance-scale	0.5199	0.5185	**0.5458**
bridges-version1	0.1321	0.3153	**0.6036**
bridges-version2	0.3184	0.3453	**0.6633**
car	0.9919	0.9025	**1.1183**
cmc	0.0490	0.0536	**0.0822**
dermatology	1.4329	1.3830	**1.5742**
ecoli	0.9690	0.9762	**1.3647**
flags	-0.0703	-0.0345	**0.5277**
hypothyroid	1.3562	1.3528	**1.3636**
iris	**0.9943**	0.9908	0.9889
letter	1.3800	1.3745	**2.1825**
lymphography	0.4843	0.5913	**0.6843**
mfeat-pixel	1.3489	1.3573	**1.5595**
nursery	1.4116	1.3509	**1.4517**
optdigits	1.1612	1.1445	**1.5310**
page-blocks	1.4273	1.4251	**1.4807**
pendigits	1.4874	1.4702	**1.8694**
postoperative-patient-data	0.3027	0.0895	**0.3052**
primary-tumor	-0.4137	-0.1325	**0.0014**
segment	1.6467	1.6276	**1.7562**
soybean	1.6113	1.6229	**2.5474**
spectrometer	-0.3791	-0.1550	**0.3876**
splice	0.9376	0.9339	**0.9441**
sponge	**0.9147**	0.9025	0.8877
tae	-0.0169	0.0155	**0.0770**
vehicle	0.4720	0.4314	**0.6176**
vowel	0.8199	0.7249	**1.3274**
waveform	0.5293	0.5231	**0.5514**
wine	0.8856	0.8940	**0.9099**
zoo	1.1130	0.9975	**1.7079**
Average	0.8055	0.8205	**1.1090**

decision modeling using a deep neural network, International Journal of Approximate Reasoning 85 (2017) 68–78. `doi:10.1016/j.ijar.2017.03.008`.

[19] M. Zaffalon, The naive credal classifier, Journal of Statistical Planning and Inference 105 (1) (2002) 5 – 21, imprecise Probability Models and their Applications. `doi:10.1016/S0378-3758(01)00201-4`.

[20] J. Abellán, A. R. Masegosa, Imprecise classification with credal decision trees, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 20 (05) (2012) 763–787. `doi:10.1142/S0218488512500353`.

[21] G. J. Klir, Uncertainty and Information: Foundations of Generalized Information Theory, John Wiley And Sons, Inc., 2006. doi:10.1002/0471755575.

[22] G. Corani, M. Zaffalon, Learning reliable classifiers from small or incomplete data sets: the naive credal classifier 2, Journal of Machine Learing Research 9 (2008) 581–621.

[23] P. Walley, Inferences from multinomial data; learning about a bag of marbles (with discussion), Journal of the Royal Statistical Society. Series B (Methodological) 58 (1) (1996) 3–57. doi:10.2307/2346164.

[24] J. Abellán, C. J. Mantas, J. G. Castellano, AdaptativeCC4.5: Credal C4.5 with a rough class noise estimator, Expert Systems with Applications 92 (Supplement C) (2018) 363 – 379. doi:10.1016/j.eswa.2017.09.057.

[25] F. P. A. Coolen, Learning from multinomial data: a nonparametric predictive alternative to the imprecise dirichlet model, in: ISIPTA05: Proceedings of the Fourth International Symposium on Imprecise Probabilities and their Applications, Fabio G. Cozman, Robert Nau and Teddy Seidenfeld (Editors). (Published by SIPTA, 2005, pp. 125–134.

[26] F. Coolen, T. Augustin, A nonparametric predictive alternative to the imprecise dirichlet model: The case of a known number of categories, International Journal of Approximate Reasoning 50 (2) (2009) 217 – 230. doi:10.1016/j.ijar.2008.03.011.

[27] J. Abellán, R. M. Baker, F. P. Coolen, R. J. Crossman, A. R. Masegosa, Classification with decision trees from a nonparametric predictive inference perspective, Computational Statistics & Data Analysis 71 (2014) 789 – 802. doi:10.1016/j.csda.2013.02.009.

[28] S. Moral, C. J. Mantas, J. G. Castellano, J. Abellán, Imprecise classification with non-parametric predictive inference, in: M.-J. Lesot, S. Vieira, M. Z. Reformat, J. P. Carvalho, A. Wilbik, B. Bouchon-Meunier, R. R. Yager (Eds.), Information Processing and Management of Uncertainty in Knowledge-Based Systems, Springer International Publishing, 2020, pp. 53–66.

[29] W.-Y. Loh, Classification and regression trees, WIREs Data Mining and Knowledge Discovery 1 (1) (2011) 14–23. doi:doi.org/10.1002/widm.8.

[30] C. E. Shannon, A mathematical theory of communication, Bell System Technical Journal 27 (3) (1948) 379–423. doi:10.1002/j.1538-7305.1948.tb01338.x.

[31] J. Abellán, Uncertainty measures on probability intervals from the imprecise dirichlet model, International Journal of General Systems 35 (5) (2006) 509–528. doi:10.1080/03081070600687643.

[32] J. Abellán, R. M. Baker, F. P. Coolen, Maximising entropy on the nonparametric predictive inference model for multinomial data, European Journal of Operational Research 212 (1) (2011) 112 – 122. doi:10.1016/j.ejor.2011.01.020.

[33] J. Abellán, S. Moral, Building classification trees using the total uncertainty criterion, International Journal of Intelligent Systems 18 (12) (2003) 1215–1225. doi:10.1002/int.10143.

[34] J. Abellan, S. Moral, Maximum of entropy for credal sets, International Journal of Uncertainty, Fuzziness and Knowledge-Based 11 (5) (2003) 587–597. doi:10.1142/S021848850300234X.

[35] R. O. Duda, P. E. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, New York, 1973.

[36] J. Abellán, Equivalence relations among dominance concepts on probability intervals and general credal sets, International Journal of General Systems 41 (2) (2012) 109–122. doi:10.1080/03081079.2011.607449.

[37] S. Moral-García, J. Abellán, Uncertainty-based information measures on the approximate non-parametric predictive inference model, International Journal of General Systems 50 (2) (2021) 159–181. doi:10.1080/03081079.2020.1866567.

[38] M. Lichman, UCI machine learning repository (2013).
URL http://archive.ics.uci.edu/ml

[39] S. Moral-García, C. J. Mantas, J. G. Castellano, M. D. Benítez, J. Abellán, Bagging of credal decision trees for imprecise classification, Expert Systems with Applications 141 (2020) 112944. `doi:10.1016/j.eswa.2019.112944`.

[40] U. Fayyad, K. Irani, Multi-valued interval discretization of continuous-valued attributes for classification learning, in: Proceeding of the 13th International joint Conference on Artificial Inteligence, Morgan Kaufmann, 1993, pp. 1022–1027.

[41] I. H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition, Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[42] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[43] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32 (1937) 675–701.

[44] P. Nemenyi, Distribution-free multiple comparisons, Doctoral dissertation, Princeton University, New Jersey, USA (1963).