On Integrating Logical Analysis of Data into Random Forests

David Ing , **Said Jabbour** , **Lakhdar Saïs**
CRIL, CNRS - Université d'Artois, France
{ing, jabbour, sais}@cril.fr

Abstract

Random Forests (RFs) are one of the most popular classifiers in machine learning. RF is an ensemble learning method that combines multiple Decision Trees (DTs), providing a more robust and accurate model than a single DT. However, one of the main step of RFs is the random selection of many different features during the construction phase of DTs, resulting in a forest with various features, which makes it difficult to extract short and concise explanations. In this paper, we propose integrating Logical Analysis of Data (LAD) into RFs. LAD is a pattern learning framework that combines optimization, Boolean functions, and combinatorial theory. One of its main goals is to generate minimal support sets (MSSes) that discriminate between different groups of data. More precisely, we show how to enhance the classical RF algorithm by randomly choosing MSSes rather than randomly choosing feature subsets that potentially contain irrelevant features for constructing DTs. Experiments on benchmark datasets reveal that integrating LAD into classical RFs using MSSes can maintain similar performance in terms of accuracy, produce forests of similar size, reduce the set of used features, and enable the extraction of significantly shorter explanations compared to classical RFs.

1 Introduction

Random Forests [Breiman, 2001] are among the widely used classifiers in the field of Machine Learning (ML). RF is an ensemble learning method that combines multiple Decision Trees (DTs) into a unified model through a specific process of aggregation and is widely used across various fields. By combining the predictions of many different DTs, RFs offer more robust and accurate models than a single DT, making them a well-known tool in many ML tasks. Despite its high performance, one of the shortcomings of RF is the use of many features during the construction of different DTs, which makes it difficult to extract short and concise explanations for end users. Indeed, the recent advancements in ML and the expected rise in ML-driven applications, particularly those sensitive applications that affect people or involve safety-critical systems, have highlighted the growing need to understand and explain the predictions made by these models. Therefore, in recent years, the field of eXplainable Artificial Intelligence (XAI) has experienced significant growth (see e.g. [Ribeiro *et al.*, 2016; Guidotti *et al.*, 2018; Ignatiev *et al.*, 2019; Audemard *et al.*, 2020; Ignatiev *et al.*, 2020; Ignatiev and Marques-Silva, 2021]). Explaining RFs has recently been investigated, and different approaches have been proposed to extract the smallest explanations for a given example or instance (see e.g. [Izza and Marques-Silva, 2021; Audemard *et al.*, 2022a; Audemard *et al.*, 2022b; Izza *et al.*, 2023; Audemard *et al.*, 2023]) by considering minimal subsets of features that explain the classification, known as Abductive eXplanations (AXps), or those that allow to change the class, known as Contrastive eXplanations (CXps). Despite recent advancements, the sizes of the explanations for RFs trained on datasets containing numerous features are still too large, since the limited size of an explanation that can be understood as a whole by humans or end users is usually set to 7 ± 2 [Miller, 1956]. Therefore, it is important to emphasize that minimizing the number of features required to build RFs can significantly impact the succinctness and comprehensibility of the derived explanations. This is the main goal addressed in this work.

To achieve the aforementioned objective, in this paper, we follow the recent advancement made for building Optimal Decision Trees thanks to Logical Analysis of Data (LAD) [Ing *et al.*, 2024]. More precisely, we propose an original integration of LAD [Crama *et al.*, 1988; Hammer, 1986] into the classical RF algorithm. LAD is a logic-based data analysis methodology that combines optimization, Boolean functions, and combinatorial theory. The primary aspects of LAD involve identifying minimal support sets (MSSes) necessary for explaining observations and uncovering hidden data patterns that differentiate observations among various groups of data. Explicitly, we demonstrate how to enhance the classical RF algorithm by randomly choosing MSSes rather than randomly choosing feature subsets that potentially contain irrelevant features for building different DTs. By doing this, the DTs of the RFs are built on randomly chosen MSSes to avoid DTs with irrelevant features. Experiments on standard benchmark datasets show that integrating MSSes of LAD into classical RFs can maintain similar performance in terms of accuracy, produce forests of similar size, decrease the set of

used features, and enable us to derive considerably shorter explanations compared to classical RFs.

The remaining sections of this paper are structured as follows. Section 2 provides some preliminaries, including ML classification, explainable AI, LAD and MSSes enumeration problem. In Section 3, we describe how to integrate LAD into the classical RF. We conduct the comparative experimental evaluation for RF methods in Section 4. Finally, Section 5 concludes the paper and addresses potential future works.

2 Preliminaries

In this section, we briefly present a formal background to introduce necessary notations used throughout the paper.

2.1 Machine Learning Classification

This paper focuses on classification for data with binary features. Using the fairly standard one-hot encoding and other standard techniques (e.g. [Fayyad and Irani, 1993; Perner and Trautzsch, 1998; Kotsiantis and Kanellopoulos, 2006; García *et al.*, 2013]), non-binary categorical and numerical features of a dataset can be transformed into binary values.

Let $\mathcal{F} = \{x_1, \ldots, x_t\}$ be a set of t binary features, all of them take value in $\{0, 1\}$. Let $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ be a binary dataset partitioned into a set of positive instances $\mathcal{E}^+$ and a set of negative instances $\mathcal{E}^-$. To refer to a binary observation vector of t features, we use the notation $\mathbf{v} = (v_1, \ldots, v_t)$, where $v_i \in \{0, 1\}$, $i = 1, \ldots, t$. An instance denotes a pair $\epsilon_q = (\mathbf{v}_q, c_q) \in \mathcal{E}$, where $\mathbf{v}_q \in \{0, 1\}^t$ and $c_q \in \{0, 1\}$ is the class value. We have $c_q = 1$ if $\epsilon_q \in \mathcal{E}^+$ and $c_q = 0$ if $\epsilon_q \in \mathcal{E}^-$. We denote a binary dataset as a set of n training instances $\mathcal{E} = \{(\mathbf{v}_1, c_1), \ldots, (\mathbf{v}_n, c_n)\}$.

The classification problem aims to learn some function f (the classifier) mapping binary observation vectors to class values, matching as accurately as possible the actual function τ on the training data (i.e. $\tau(\mathbf{v}_q) = c_q$, $q = 1, \ldots, n$) and tries to generalize well to new and unseen data. In this paper, we focus on Decision Tree and Random Forest classifiers.

For Decision Tree (DT), the function f is represented by a binary and full DT, where every internal node has exactly two children. Every internal node is associated with a feature or its negation and every leaf node is associated with a class.

Random Forest (RF) is an ensemble method that constructs multiple decision trees (DTs) on various subsamples of the training set and uses averaging to improve the accuracy. RF mitigates overfitting by combining diverse DTs, where each DT might have high variance, into a more robust and generalized model. Formally, let $\mathcal{RF} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$ be a collection of K DTs. Each tree $\mathcal{T}_i \in \mathcal{RF}$ is trained on a different bootstrap sample from the training set and is constructed recursively based on a random subset of features. Given an observation vector $\mathbf{v}$, each tree $\mathcal{T}_i$ predicts a class label $\mathcal{T}_i(\mathbf{v}) \in \{0, 1\}$. The final prediction of the $\mathcal{RF}$, $f(\mathbf{v})$ depends on a voting mechanism among the K DTs:

$$f(\mathbf{v}) = \text{vote}\left(\mathcal{T}_1(\mathbf{v}), \mathcal{T}_2(\mathbf{v}), \ldots, \mathcal{T}_K(\mathbf{v})\right) \in \{0, 1\}.$$

In this study, we consider two different voting mechanisms: majority-vote and soft-vote. These two mechanisms can produce different results as described in Example 1.

Example 1. *Consider a case where a soft-vote produces a different outcome compared to a majority-vote. Suppose we have a random forest made of three DTs, $\mathcal{RF} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$. The predictions of those three DTs are given as follows:*

1. *$\mathcal{T}_1$ predicts class 0 with probability 0.7 for class 0 and 0.3 for class 1.*
2. *$\mathcal{T}_2$ predicts class 1 with probability 0.4 for class 0 and 0.6 for class 1.*
3. *$\mathcal{T}_3$ predicts class 1 with probability 0.45 for class 0 and 0.55 for class 1.*

***Majority-vote**: class 0 is predicted by $\mathcal{T}_1$, class 1 is predicted by $\mathcal{T}_2$ and $\mathcal{T}_3$. Thus, the final prediction of $\mathcal{RF}$ is class 1.*
***Soft-vote**: The mean probability for class 0 = $\frac{0.7+0.4+0.45}{3} \simeq 0.5167$ and the mean probability for class 1 = $\frac{0.3+0.6+0.55}{3} \simeq 0.4833$. In this case, class 0 has a higher mean probability of 0.5167. Thus, the final prediction of $\mathcal{RF}$ is class 0.*

2.2 Explainable Artificial Intelligence

Explainable Artificial Intelligence (XAI) has recently received significant attention and has emerged as a fundamental field, aiming to allow human users to understand and trust the results produced by AI/ML systems [Barredo Arrieta *et al.*, 2020]. Explainability for AI systems has grown along with the success and adoption of deep learning systems, which are applied in many fields where trustworthiness is critically needed, such as in legal systems [Lipton, 2018; Rudin, 2019], autonomous driving [Tian *et al.*, 2018], cybersecurity [Shone *et al.*, 2018], health care and criminal justice [Rudin and Ustun, 2018], financial risk assessment [Chen *et al.*, 2018], detecting heart attacks [Weng *et al.*, 2017], surveillance systems [Ding *et al.*, 2018], among others.

In this paper, we focus on two well-known explanations, abductive explanation (AXp) as an answer to a "why?" question and contrastive explanation (CXp) as an answer to "why not?" question, subject of recent works for explaining RFs [Izza and Marques-Silva, 2021; Audemard *et al.*, 2022b].

Given an ML model, computing a classification function τ on feature space $\mathbb{F} = \{0, 1\}^t$, a specific point $\mathbf{v} \in \mathbb{F}$, with prediction $c = \tau(\mathbf{v})$, where $\mathbf{v} = (v_1, \ldots, v_t)$. To refer to an arbitrary point in $\mathbb{F}$, we use the notation $\mathbf{u} = (u_1, \ldots, u_t)$.

1. A PI-explanation or Abductive eXplanation (AXp) is any minimal subset $\mathcal{X} \subseteq \mathcal{F}$ such that:

$$\forall(\mathbf{u} \in \mathbb{F}).[\bigwedge_{i \in \mathcal{X}} (u_i = v_i)] \rightarrow (\tau(\mathbf{u}) = c)$$

2. A Contrastive eXplanation (CXp) is any minimal subset $\mathcal{Y} \subseteq \mathcal{F}$ such that:

$$\exists(\mathbf{u} \in \mathbb{F}). \bigwedge_{j \in \mathcal{F} \setminus \mathcal{Y}} (u_j = v_j) \wedge (\tau(\mathbf{u}) \neq c)$$

2.3 Logical Analysis of Data

Logical Analysis of Data (LAD) is a logic-based data analysis methodology combining ideas and concepts from discrete optimization, combinatorics and the theory of Boolean functions. The idea of LAD was first described by Peter L. Hammer in a lecture given in 1986 [Hammer, 1986] and was later

expanded and developed in [Crama *et al.*, 1988]. The central concepts behind LAD are the computation of minimal support sets of features for explaining all observations, and the discovery of hidden data patterns capable of distinguishing positive and negative sets of observations. A collection of such patterns is used for building a classification procedure, clustering, feature selection and other related problems.

LAD has undergone continuous development in various domains, including medical applications [Hammer and Bonates, 2006], financial applications [Hammer *et al.*, 2007], the airline industry [Mortada *et al.*, 2012], condition-based maintenance applications [Yacout *et al.*, 2017], industrial chemical processes [Ragab *et al.*, 2018], among others. For more details on LAD, we refer the reader to [Boros *et al.*, 2000; Hammer and Bonates, 2006]. Our goal in this paper is to utilize the main first step of LAD, which involves discovering support sets that distinguish observations among various groups of data, and to integrate these support sets into RFs.

x_1	x_2	x_3	x_4	x_5	$class$
0	0	0	1	1	1
0	0	0	0	1	1
1	1	1	1	0	1
1	0	1	0	0	0
1	0	1	1	1	0

Figure 1: Binary dataset.

Example 2. *Consider dataset in Figure 1. The projection of $\mathcal{E}^+$ and $\mathcal{E}^-$ on $\mathcal{S} = \{x_1, x_2\}$ is $\mathcal{E}^+_{\mathcal{S}} = \{(0,0),(1,1)\}$ and $\mathcal{E}^-_{\mathcal{S}} = \{(1,0)\}$, respectively. Moreover, $\{x_1, x_2\}$, $\{x_2, x_3\}$, $\{x_3, x_4, x_5\}$, and $\{x_1, x_4, x_5\}$ are minimal (w.r.t. inclusion) support sets. Concretely, $\{x_1, x_2\}$ and $\{x_2, x_3\}$ are the minimum size support sets (w.r.t. cardinality).*

In the sequel, we use similar notations and definitions to those of [Hammer and Bonates, 2006]. Given a set of binary features $\mathcal{S} \subseteq \mathcal{F}$, let $\mathcal{E}^+_{\mathcal{S}}$ (resp. $\mathcal{E}^-_{\mathcal{S}}$) be the projection of $\mathcal{E}^+$ (resp. $\mathcal{E}^-$) on $\mathcal{S}$. $\mathcal{S}$ is called a *support set* if $\mathcal{E}^+_{\mathcal{S}} \cap \mathcal{E}^-_{\mathcal{S}} = \emptyset$. Moreover, $\mathcal{S}$ is called *irredundant* or *minimal* if no proper subset of it is a support set. To compute minimum support sets in $\mathcal{E}$, we associate with every feature $x_k \in \mathcal{F}$ a new binary variable y_k, where $k = 1, \ldots, t$, $y_k = 1$ if x_k is part of the support set, and $y_k = 0$ otherwise. Let $\mathbf{v} = (v_1, \ldots, v_t)$ and $\mathbf{v}' = (v'_1, \ldots, v'_t)$ be the binary observation vectors of t features associated with $\mathcal{E}^+$ and $\mathcal{E}^-$, respectively. We further associate the vectors $\mathbf{v}$ and $\mathbf{v}'$ with a vector $w(\mathbf{v}, \mathbf{v}') = (w_1(\mathbf{v}, \mathbf{v}'), \ldots, w_t(\mathbf{v}, \mathbf{v}'))$, where $w_k(\mathbf{v}, \mathbf{v}') = v_k \oplus v'_k (mod\ 2)$, i.e. $w_k(\mathbf{v}, \mathbf{v}') = 1$ if $v_k \neq v'_k$, and $w_k(\mathbf{v}, \mathbf{v}') = 0$ otherwise. We can obtain the minimum support sets by solving the following set covering problem:

$$\begin{aligned} min \quad & \sum_{k=1,\ldots,t} y_k \\ s.t. \quad & \\ \sum_{k=1,\ldots,t} & w_k(\mathbf{v}, \mathbf{v}') y_k \geq 1,\ \forall (\mathbf{v}, c) \in \mathcal{E}^+,\ \forall (\mathbf{v}', c') \in \mathcal{E}^- \\ & y_k \in \{0, 1\} \end{aligned} \tag{1}$$

Example 3. *For the dataset in Figure 1, the formulation of LAD Minimum Support Sets (LAD-MSS) is given by:*

$$\begin{aligned} min \quad & y_1 + y_2 + y_3 + y_4 + y_5 \\ & s.t. \\ & y_1 + y_3 + y_4 + y_5 \geq 1 \\ & y_1 + y_3 \geq 1 \\ & y_1 + y_3 + y_5 \geq 1 \\ & y_1 + y_3 + y_4 \geq 1 \\ & y_2 + y_4 \geq 1 \\ & y_2 + y_5 \geq 1 \end{aligned} \tag{2}$$

The above problem is a known NP-hard and involves a quadratic number of linear inequalities. For large datasets, solving this optimization problem is difficult in practice. In Section 2.4, we circumvent this problem by proposing a more efficient alternative, formulating it as the problem of generating minimal hitting sets.

2.4 Minimal Support Sets Enumeration

We first provide a relation between minimal support sets (MSSes) and minimal hitting sets (MHSes), and then describe how to enumerate them. We begin by recalling the minimal hitting set problem.

Definition 1. *Given a collection $H = \{H_1, \ldots, H_m\}$ of subsets over a universe U. A hitting set (traversal) T of H is a subset of U that intersects (hits) every set $H_i \in H$ i.e. $T \cap H_i \neq \emptyset$. Moreove, T is minimal if there is no $T' \subset T$ s.t. T' is a hitting set.*

The minimal hitting set problem is relevant to multiple fields, including Boolean algebra [Fredman and Khachiyan, 1996], computational biology [Ideker *et al.*, 1999], data mining [Bailey *et al.*, 2003], combinatorics [Eiter *et al.*, 2008], and more.

In the literature, there exists several algorithms and efficient implementations to compute the set of MHSes [Gainer-Dewar and Vera-Licona, 2016]. Murakami and Uno [Murakami and Uno, 2014] propose "pMMCS", a parallel implementation version of their first algorithm MMCS (Minimal-to-Maximal Conversion Search), for solving the MHSes enumeration, whose complexity is in $O(\|H\|)$ time per MHS and $O(\|H\|)$ memory, $\|H\|$ denotes the sum of the sizes of elements of H. The advantage of this algorithm is its ability to tackle large-scale problems involving millions of hyperedges, quickly producing numerous solutions, which influenced our decision. Additionally, it supports multithreaded implementations on systems with multiple CPUs.

Let $\mathcal{F} = \{x_1, \ldots, x_t\}$ be a set of t binary features, $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ a binary dataset, $\mathbf{v}$ and $\mathbf{v}'$ the binary observation vectors associated with $\mathcal{E}^+$ and $\mathcal{E}^-$, respectively. Using the result of Section 2.3, we can derive a set of binary vectors from $\mathcal{E}$ denoted as $\mathcal{E}^+ \oplus \mathcal{E}^- = \mathcal{E}_\oplus = \{w(\mathbf{v}, \mathbf{v}'),\ \forall (\mathbf{v}, c) \in \mathcal{E}^+ \wedge \forall (\mathbf{v}', c') \in \mathcal{E}^-\}$. We denote by $H(\mathcal{E}_\oplus)$ the following set $H(\mathcal{E}_\oplus) = \{\mathcal{E}_\oplus^{(\mathbf{v},\mathbf{v}')},\ \forall (\mathbf{v}, c) \in \mathcal{E}^+ \wedge\ \forall (\mathbf{v}', c') \in \mathcal{E}^-\}$, where $\mathcal{E}_\oplus^{(\mathbf{v},\mathbf{v}')} = \{x_k \in \mathcal{F} \mid k \in \{1, \ldots, t\}$ and $w_k(\mathbf{v}, \mathbf{v}') = 1\}$.

As proven in [Ing *et al.*, 2024], the minimal support sets (MSSes) of $\mathcal{E}$ correspond to $MHSes(H(\mathcal{E}_\oplus))$. Consequently, the MSSes generation can be performed as follows:

1. *XOR Operations* ($\mathcal{E} \rightarrow \mathcal{E}^+ \oplus \mathcal{E}^-$): execute XOR operations between every observation in $\mathcal{E}^+$ and every observation in $\mathcal{E}^-$ of $\mathcal{E}$ denoted as $\mathcal{E}_\oplus = \mathcal{E}^+ \oplus \mathcal{E}^-$, a set of binary vectors. For multi-class classification, we perform XOR operations between each pair of classes.
2. Generate MSSes for $H(\mathcal{E}_\oplus)$, noted as $MSSes(H(\mathcal{E}_\oplus))$.

Example 4. *From the formulation of LAD-MSS as a 0/1 linear program (Example 3), we can derive the following set of sets, where each subset corresponds to the set of features associated with the set of variables involved in each 0/1 linear inequality:* $H(\mathcal{E}_\oplus) = \{\{x_1, x_3, x_4, x_5\}, \{x_1, x_3\}, \{x_1, x_3, x_5\}, \{x_1, x_3, x_4\}, \{x_2, x_4\}, \{x_2, x_5\}\}$. *Then* $MSSes(H(\mathcal{E}_\oplus)) = \{\{x_1, x_2\}, \{x_2, x_3\}, \{x_3, x_4, x_5\}, \{x_1, x_4, x_5\}\}$ *corresponds to the minimal supports sets in the Example 2.*

Note that the number of MSSes can grow exponentially in the worst case. For practical purposes, in this paper, we propose generating only a subset for integration into RFs.

3 Random Forest Based LAD

A main result proved in [Ing *et al.*, 2024] shows that for a minimal support set $\mathcal{S}$, it is possible to build a perfect decision tree (see Definition 3) that correctly discriminates between positive and negative examples using only the features of $\mathcal{S}$ for a dataset without redundancy (see Definition 2). Nevertheless, the number of MSSes can be exponentially large, and choosing the most relevant MSS is a computationally hard task. Moreover, finding the smallest MSSes with respect to cardinality is also NP-hard. For this reason, the authors considered generating minimal support sets (w.r.t. inclusion) instead of generating minimum ones (w.r.t. cardinality), and a heuristic scoring method has been designed to select the most effective MSS for building a perfect decision tree.

Definition 2. *Let $\mathcal{F}$ be a set of binary features and $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ a binary dataset over $\mathcal{F}$. $\mathcal{S} \subset \mathcal{F}$ is called redundant if $\mathcal{E}^+_\mathcal{S} \cap \mathcal{E}^-_\mathcal{S} \neq \emptyset$.*

Definition 3. *A decision tree is perfect if it correctly classifies all the observations of $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$.*

Let us now discuss how MSSes can be leveraged within the RF algorithm. First, we recall that the classical RF (Algorithm 1) creates a bootstrap sample of the same size as the training data using the `Bootstrap` function, where rows from the training data are randomly selected with replacement (line 8). For each bootstrap sample, by default, $\sqrt{t}$ features are randomly selected to form a feature subset, and a DT algorithm (e.g. `CART`) is then used to build a DT over the bootstrap sample based on that feature subset (line 9).

As stated in Section 2.3, a key advantage of LAD is its capability to generate minimal feature subsets that effectively distinguish between different classes. In contrast, one of the weaknesses of the classical RF is its sensitivity to noise, which can lead to the selection of irrelevant features when training on bootstrap samples, resulting in the use of too many features. To address this drawback, MSSes are valuable due to their discriminative power, focusing on relevant features. Algorithm 2 describes the proposed RF-LAD approach. In detail, $N_\mathcal{S}$ MSSes are generated using the training

Algorithm 1: Classical Random Forest

1: **Input:**
2: $\mathcal{E}$: Training dataset with n samples
3: K: Number of trees in the forest
4: t: Number of features
5: **Output:**
6: Ensemble of K decision trees
7: **for** $i = 1$ to K **do**
8: $\mathcal{E}_i \leftarrow$ Bootstrap($\mathcal{E}$)
9: $\mathcal{T}_i \leftarrow$ CART($\mathcal{E}_i, \sqrt{t}$)
10: **end for**
11: **return** $\mathcal{RF} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$

data $\mathcal{E}$ (line 7). Among these MSSes, K MSSes are randomly pre-selected without replacement (i.e. no redundancy among them) to be associated with each bootstrap sample (line 8). Over each bootstrap sample, the tree is built using the features of its associated MSS (line 11). Note that each $\mathcal{S}_i$ remains a support set of $\mathcal{E}_i$ but not necessarily minimal. In fact, since $\mathcal{E}_i$ is a subset of the training data $\mathcal{E}$, each support set of $\mathcal{E}$ is a support set of $\mathcal{E}_i$.

Since the number of MSSes is exponential in the worst case, limiting the enumeration to $N_\mathcal{S}$ ones is useful to control this complexity. Additionally, as K MSSes are chosen among $N_\mathcal{S}$, $N_\mathcal{S}$ must be large enough to ensure diversification and thus allow for a subset of MSSes that is representative of the considered data.

Let us note that using RF-LAD through MSSes ensures that important features have a high chance of being used. Moreover, using MSSes' features rather than randomly generated features for bootstrap samples can decrease the number of used features since minimality is required for support sets. Consequently, the abductive and contrastive explanations can also be significantly reduced accordingly.

As mentioned earlier, on data without redundancies, it is possible to classify all the training examples correctly by building perfect decision trees. Nonetheless, when using a classical Random Forest algorithm, we do not have guarantees in terms of perfect classification for each decision tree due to the randomization used to restrict the number of features for each bootstrap sample. Now, using RF-LAD, the trees are built over MSSes generated using the whole training data. Since MSSes remain support sets for the bootstraps, perfect trees can be built. For instance, this can be achieved by removing the restriction on depth in the CART classifier.

Next, we discuss how MSSes can be used to enforce some examples to be correctly classified.

Proposition 1. *Let $\mathcal{RF} = \{\mathcal{T}_1, ..., \mathcal{T}_K\}$ be a set of K perfect decision trees built over a set of bootstraps $\{\mathcal{E}_1, ..., \mathcal{E}_K\}$. If $\epsilon_q = (\mathbf{v}_q, c_q)$, s.t. $\mathbf{v}_q$ appears at least $\lfloor \frac{K}{2} \rfloor + 1$ times in $\{\mathcal{E}_1, ..., \mathcal{E}_K\}$, then $\mathcal{RF}(\mathbf{v}_q) = c_q$, using the majority vote.*

Proof. Assume that all the decision trees (DTs) of $\mathcal{RF}$ are perfect. Then, each $\mathcal{T}_i$ correctly classifies all the examples of $\mathcal{E}_i$. If $\epsilon_q = (\mathbf{v}_q, c_q)$ appears more than $\lfloor \frac{K}{2} \rfloor$ times, using the majority vote for $\mathcal{RF}$ will classify $\mathbf{v}_q$ as class c_q with more than $\lfloor \frac{K}{2} \rfloor$ DTs. Consequently, $\mathcal{RF}(\mathbf{v}_q) = c_q$. □

Proposition 1 ensures that frequent examples among bootstraps are guaranteed to be well classified. However, classical RF uses a random policy to choose features in each bootstrap, which leads to redundancies, i.e. identical examples that appear in both positive and negative classes (Definition 2), preventing perfect classification for all variants of DT classifiers. In contrast, by using MSSes, we will be able to ensure that RF-LAD can classify the majority of frequent examples into their corresponding classes.

Algorithm 2: Random Forest Based LAD (RF-LAD)

Input:
  $\mathcal{E}$: Training dataset with n samples
  K: Number of trees in the forest
  $N_{\mathcal{S}}$: Number of MSSes to generate
Output:
  $\mathcal{RF}$: Ensemble of K decision trees
$L_{\mathcal{S}} \leftarrow$ generate_MSSes$(\mathcal{E}, N_{\mathcal{S}})$
$\{\mathcal{S}_1, \ldots, \mathcal{S}_K\} \leftarrow$ random_select$(L_{\mathcal{S}}, K)$
for $i = 1$ to K **do**
  $\mathcal{E}_i \leftarrow$ Bootstrap$(\mathcal{E})$
  $\mathcal{T}_i \leftarrow$ CART$(\mathcal{E}_i, \mathcal{S}_i)$
end for
return $\mathcal{RF} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$

4 Experimental Results

In this section, we conduct comparative experiments between our Random Forest using MSSes of LAD (RF-LAD) and the state-of-the-art Random Forest (RF) (from the scikit-learn Python library [Pedregosa *et al.*, 2011]). The parameters of RFs are kept at their default values (i.e. in a forest, $K = 100$ DTs), except for the maximum depth, where we set different depths d $\in \{3, 4, 5\}$. The assessment is performed on 23 datasets, which are standard benchmarks originating from well-known repositories such as CP4IM (https://dtai-static.cs.kuleuven.be/CP4IM/datasets/), Kaggle (www.kaggle.com), OpenML (www.openml.org), and UCI (archive.ics.uci.edu/ml/). Some of them are original binary datasets, while others are numerical datasets that have been binarized using thresholds retrieved from CART [Breiman *et al.*, 1984]. Let us note that, for certain datasets, instances appearing in both $\mathcal{E}^+$ and $\mathcal{E}^-$ are removed to maintain consistency. For every benchmark, a Repeated Stratified 10-fold cross-validation with 3 repetitions have been achieved to maintain the class distribution (i.e. to address imbalanced datasets). To enumerate the MSSes, we use the multithreaded implementations provided by the algorithm "pMMCS" [Murakami and Uno, 2014], and set the number of threads to 20 to ensure diversification in terms of the generated MSSes. To control the complexity, we modified "pMMCS" by limiting the number of MSSes to 100K (i.e. $N_{\mathcal{S}} = 100$K) for all the considered datasets. For a fair comparison, we randomly select 100 different MSSes (without redundancy) from the 100K generated MSSes, resulting in 100 different DTs to build our RF-LAD. Two voting mechanisms are used for RF-LAD approach: the majority-vote and the soft-vote. Experiments are conducted on a computer equipped with Intel(R) Core(TM) i9-10900 CPU @ 2.80GHz with 62Gib of memory.

The results are reported in Table 1. The first column shows the name of the dataset, the size of the dataset (#s), the number of features (#f), and the number of classes (#c). The second column displays different depth settings (d). The column $\text{RF}_{\text{sklearn}}$ represents the RF algorithm from sklearn (it utilizes the soft-voting mechanism), while the columns RF_{LAD}(majority-vote) and RF_{LAD}(soft-vote) represent our proposed RF-LAD using majority-vote and soft-vote, respectively. The column "Acc." stands for the testing or prediction accuracy along with its standard deviation, and the column "#nbNodes" is the average total number of nodes in a forest. The column "Time(s)" measures the average runtime for constructing our RF-LAD. This runtime includes encoding the dataset into an MHS problem, generating 100K MSSes, randomly selecting 100 MSSes, and building 100 DTs. Note that we do not report exact runtime for the $\text{RF}_{\text{sklearn}}$ because it takes only a few seconds. According to the experiments, $\text{RF}_{\text{sklearn}}$ and RF-LAD(s) perform similarly across most benchmarks in terms of accuracy, with differences often within only 1% to 2%. The higher accuracies with more than 2% are highlighted in bold. For the standard deviations, they seem to be very similar, with only slight differences (between 1% and 2%) for some datasets. In terms of #nbNodes, RF-LAD(s) are able to produce smaller forests on 10 benchmarks across all depths (e.g. anneal, drilling, fetal-health, ...).

Our second comparison concerns the average number of features used in a forest between $\text{RF}_{\text{sklearn}}$ and RF-LAD(s). The top histogram of Figure 2 displays the average number of features used in $\text{RF}_{\text{sklearn}}$ and in RF-LAD(s) for d=4. The x-axis displays all the considered datasets, and the y-axis displays the average number of features used. As we can see in that histogram, for all the considered datasets, RF-LAD(s) used fewer features to build the entire forest compared to $\text{RF}_{\text{sklearn}}$. Statistically, for all the benchmark datasets, RF_{LAD}(majority-vote) used from 4.72% to 78% less than $\text{RF}_{\text{sklearn}}$, whereas RF_{LAD}(soft-vote) used from 3.92% to 78.83% less than $\text{RF}_{\text{sklearn}}$. Furthermore, for the datasets with more than 200 features, such as "ad", "cnae", "pd-speech", "reuters", and "spambase", the reduction exceeds 50%, showing strong reduction. These results show that RF-LAD(s) are beneficial, notably for datasets containing numerous features.

Our final comparison focuses on the size of the explanations derived from $\text{RF}_{\text{sklearn}}$ and RF-LAD(s). To derive such explanations from those RFs, we utilized the recent Random Forest explanation tool (RFxpl: https://github.com/izzayacine/RFxpl) proposed by Izza and Marques-Silva [Izza and Marques-Silva, 2021]. The authors proposed a propositional encoding for computing explanations of RFs, enabling the extraction of abductive explanations (AXps) and contrastive explanations (CXps) using a SAT solver. First, to obtain a model for each RF algorithm, we saved the best trained model based on the highest testing accuracy during the evaluation process of all 30 iterations (10-fold cross-validation with 3 repetitions). Then, to choose the examples for explanation, we followed the procedure in [Izza and Marques-Silva, 2021] by randomly picking fractions of the dataset, depending on its size. More precisely, for datasets contain-

Instance #s, #f, #c	d	$RF_{sklearn}$		RF_{LAD}(majority-vote)			RF_{LAD}(soft-vote)		
		Acc.	#nbNodes	Acc.	#nbNodes	Time (s)	Acc.	#nbNodes	Time (s)
ad	3	86.81(± 1.31)	1017.73	**92.36(± 1.81)**	1346.27	244.27	**92.53(± 1.82)**	1348.2	246.04
	4	88.36(± 1.45)	1511.4	**93.01(± 1.73)**	2119.6	249.39	**93.15(± 1.82)**	2104.65	247.65
2108, 1558, 2	5	89.92(± 1.39)	2052	**93.39(± 1.63)**	2885.6	251.84	**93.50(± 1.47)**	2879.94	253.46
anneal	3	87.26(± 2.53)	1163	85.58(± 2.34)	928	1.59	86.09(± 2.29)	924.93	1.97
	4	88.33(± 2.52)	1885.14	87.26(± 2.28)	1503.27	1.98	87.31(± 2.41)	1506.94	1.59
712, 93, 2	5	89.51(± 2.22)	2755.65	88.71(± 3.03)	2273	1.61	88.76(± 2.61)	2290.47	1.58
australian-credit	3	86.16(± 4.52)	1458	84.68(± 4.77)	1491.94	2.7	85.55(± 4.50)	1490.33	2.72
	4	86.42(± 4.50)	2750.6	85.96(± 4.96)	2860.65	2.76	86.32(± 4.37)	2853.94	2.7
653, 125, 2	5	87.33(± 4.31)	4702.46	86.26(± 4.63)	4874.4	2.7	86.42(± 4.58)	4855.8	2.75
breast-cancer	3	96.73(± 1.83)	1192.94	95.07(± 2.20)	1302.67	2.98	95.22(± 1.84)	1306.86	3.01
	4	96.63(± 1.77)	1921.6	95.66(± 2.10)	2125.27	2.98	95.61(± 2.18)	2142.2	2.97
683, 89, 2	5	96.78(± 1.74)	2784.4	95.85(± 2.15)	3077.65	3.89	95.85(± 1.91)	3078	3.02
cnae	3	**87.05(± 3.09)**	832.73	83.31(± 3.75)	1169.33	76.33	84.71(± 2.53)	1160.86	76.58
	4	**89.22(± 2.44)**	1168.8	86.30(± 3.82)	1697	76.19	86.98(± 2.64)	1694.67	77.28
1027, 856, 2	5	**90.36(± 2.77)**	1561.6	87.79(± 3.12)	2238.65	75.17	88.38(± 3.11)	2225.53	77.43
contraceptive	3	54.99(± 3.56)	1463.73	54.88(± 4.38)	1234	86.18	56.49(± 4.25)	1238.47	85.76
	4	56.52(± 4.19)	2895.47	57.44(± 4.60)	2215	85.33	57.95(± 4.34)	2205.87	85.46
1262, 90, 3	5	56.81(± 3.93)	5405.26	**58.90(± 4.24)**	3949.4	85.90	57.97(± 4.78)	3926.8	85.61
drilling	3	67.83(± 7.72)	1088	**76.38(± 8.15)**	1024.94	0.78	**84.39(± 5.67)**	1034.67	0.79
	4	77.50(± 8.59)	1767.53	**85.13(± 7.62)**	1649	0.8	**91.25(± 3.94)**	1628.4	0.79
285, 56, 2	5	87.61(± 4.54)	2628.47	**91.35(± 5.13)**	2432	0.78	**93.27(± 4.85)**	2421.27	0.78
fetal-health	3	87.61(± 1.79)	1477.94	**89.70(± 1.56)**	1480.53	91.09	**89.99(± 1.85)**	1480.33	91.39
	4	90.36(± 1.75)	2862.13	90.58(± 1.72)	2868.94	90.16	90.66(± 1.67)	2864.8	91.53
1806, 93, 3	5	91.60(± 1.89)	5079.93	91.67(± 1.31)	4895.74	91.99	91.82(± 1.31)	4907.74	91.77
german-credit	3	70.59(± 1.22)	1482.2	70.48(± 0.80)	1489.73	12.46	70.60(± 1.72)	1491.6	12.39
	4	72.00(± 1.69)	2911.35	70.94(± 1.67)	2952.94	12.35	71.20(± 1.57)	2946.2	12.35
1000, 112, 2	5	72.70(± 2.46)	5222.26	71.47(± 2.18)	5331.87	12.45	71.56(± 2.26)	5321	12.4
heart-cleveland	3	81.59(± 8.34)	1470.27	81.85(± 7.85)	1487.86	0.87	81.51(± 5.90)	1488.4	0.78
	4	**82.84(± 7.53)**	2775.87	79.93(± 6.95)	2849.53	0.79	80.85(± 7.23)	2846.73	0.82
296, 95, 2	5	82.72(± 7.04)	4543.93	80.84(± 6.75)	4646	0.79	81.06(± 6.44)	4646	0.82
hepatitis	3	82.83(± 7.44)	1335	81.33(± 6.28)	1326.4	0.43	81.28(± 7.83)	1319.6	0.46
	4	82.85(± 8.09)	2128.94	82.05(± 7.88)	2077.87	0.45	81.33(± 8.37)	2085.73	0.47
137, 68, 2	5	82.12(± 9.14)	2899.47	82.08(± 8.52)	2798.65	0.47	81.81(± 8.20)	2791.27	0.44
hypothyroid	3	96.47(± 1.22)	1139.67	98.37(± 0.47)	1123.8	77.02	98.38(± 0.51)	1126	87.08
	4	97.14(± 1.22)	1850.67	98.64(± 0.49)	1800.67	76.9	98.62(± 0.47)	1800.86	77.4
3199, 88, 2	5	97.94(± 0.75)	2822	98.59(± 0.59)	2579	77.04	98.59(± 0.53)	2574.8	76.86
indian-diabetes	3	75.65(± 4.09)	1482.94	75.48(± 4.47)	1491.94	4.27	75.74(± 4.43)	1492.86	5.42
	4	76.30(± 4.13)	2876.13	76.69(± 3.98)	2928.47	5.39	77.04(± 3.70)	2925.27	4.15
768, 97, 2	5	77.42(± 3.94)	5088.54	77.73(± 3.95)	5205.87	4.21	77.25(± 3.58)	5211.74	4.11
indian-liver	3	70.58(± 2.89)	1462.53	70.58(± 2.56)	1458.67	1.94	69.92(± 3.61)	1458.8	1.94
	4	69.98(± 3.55)	2732.53	70.28(± 4.08)	2685	1.89	70.17(± 4.02)	2682.87	1.88
554, 84, 2	5	70.29(± 4.78)	4586	70.16(± 4.33)	4417.13	1.88	69.01(± 4.48)	4417.87	1.88
loan	3	72.93(± 4.25)	1358.4	**81.29(± 4.43)**	1404.2	1.30	**81.36(± 4.44)**	1402.14	1.26
	4	78.61(± 5.58)	2402.6	**81.07(± 4.15)**	2440.6	1.28	**81.08(± 4.49)**	2438.2	1.27
474, 68, 2	5	79.25(± 4.65)	3888.2	80.87(± 4.88)	3822.35	1.27	80.79(± 4.64)	3807.47	1.27
lymph	3	80.12(± 9.43)	1313.94	80.07(± 9.28)	1357.8	0.52	80.76(± 9.59)	1352.4	0.5
	4	81.73(± 7.81)	2145.73	81.42(± 9.23)	2217.4	0.5	82.36(± 8.77)	2223.4	0.5
148, 68, 2	5	**83.53(± 9.14)**	3087.8	81.24(± 10.33)	3130.4	0.5	**83.26(± 9.31)**	3137.73	0.52
pd-speech	3	**82.00(± 3.61)**	1497.47	79.75(± 2.56)	1487.6	9.58	80.45(± 2.80)	1488.2	9.57
	4	82.79(± 3.42)	2986.27	81.34(± 2.56)	2873.87	9.68	81.08(± 2.84)	2875.53	9.54
754, 754, 2	5	83.19(± 3.50)	5347	81.47(± 3.04)	4993.2	9.83	81.82(± 2.59)	4991.2	9.62
pima	3	76.17(± 4.30)	1447	75.22(± 5.22)	1495.6	2.75	75.22(± 5.43)	1495.4	2.75
	4	76.35(± 5.41)	2724	75.45(± 5.74)	2951.2	2.78	75.85(± 6.09)	2952.87	2.74
740, 160, 2	5	76.26(± 5.20)	4643.4	75.72(± 4.98)	5293.13	3.09	76.12(± 6.08)	5293.26	3.32
reuters	3	88.02(± 0.10)	1373.27	**92.33(± 1.68)**	1363.47	43.32	**92.74(± 1.51)**	1362.73	43.5
	4	88.91(± 0.76)	2525.53	**93.20(± 1.43)**	2369.73	43.31	**93.11(± 1.53)**	2390.13	43.15
1834, 246, 2	5	90.58(± 0.91)	4084.6	**93.67(± 1.57)**	3577.4	43.55	**94.02(± 1.41)**	3535.73	43.27
soybean	3	86.02(± 0.71)	1226.33	**90.24(± 2.23)**	1344	1.06	**90.56(± 2.19)**	1344.53	1
	4	90.08(± 2.45)	2013.4	91.89(± 2.30)	2257.47	1	**92.69(± 2.25)**	2254.94	1.06
625, 50, 2	5	92.37(± 2.62)	2975	92.95(± 2.35)	3385.2	1.03	93.54(± 2.55)	3374.47	1.06
spambase	3	90.24(± 1.30)	1479.67	88.67(± 1.79)	1359.6	612.68	89.01(± 1.62)	1354.67	595.9
	4	91.43(± 1.37)	2906.65	90.40(± 1.64)	2557.35	613.48	90.57(± 1.73)	2557.73	620.32
3471, 236, 2	5	91.78(± 1.11)	5222.2	91.33(± 1.47)	4432.67	626.24	91.40(± 1.57)	4418.33	620.88
startup	3	78.54(± 3.50)	1383.14	76.84(± 3.93)	1444.4	10.47	77.67(± 3.39)	1449.6	10.43
	4	**79.22**(± 3.55)	2587	77.01(± 3.28)	2733.87	10.56	77.70(± 3.37)	2733.6	10.45
921, 108, 2	5	79.15(± 3.39)	4508.8	77.85(± 3.07)	4802.4	10.56	77.77(± 3.36)	4821.33	10.55
wdbc	3	95.60(± 2.59)	1246.14	95.49(± 2.10)	1125	1.57	95.31(± 2.49)	1117	1.56
	4	96.13(± 2.32)	2081.87	96.48(± 2.26)	1849.8	1.6	96.48(± 2.44)	1821.86	1.62
569, 88, 2	5	96.66(± 2.41)	3056	96.54(± 2.69)	2769.13	1.7	96.89(± 2.42)	2762.35	1.64

Table 1: Experimental results for Random Forests.

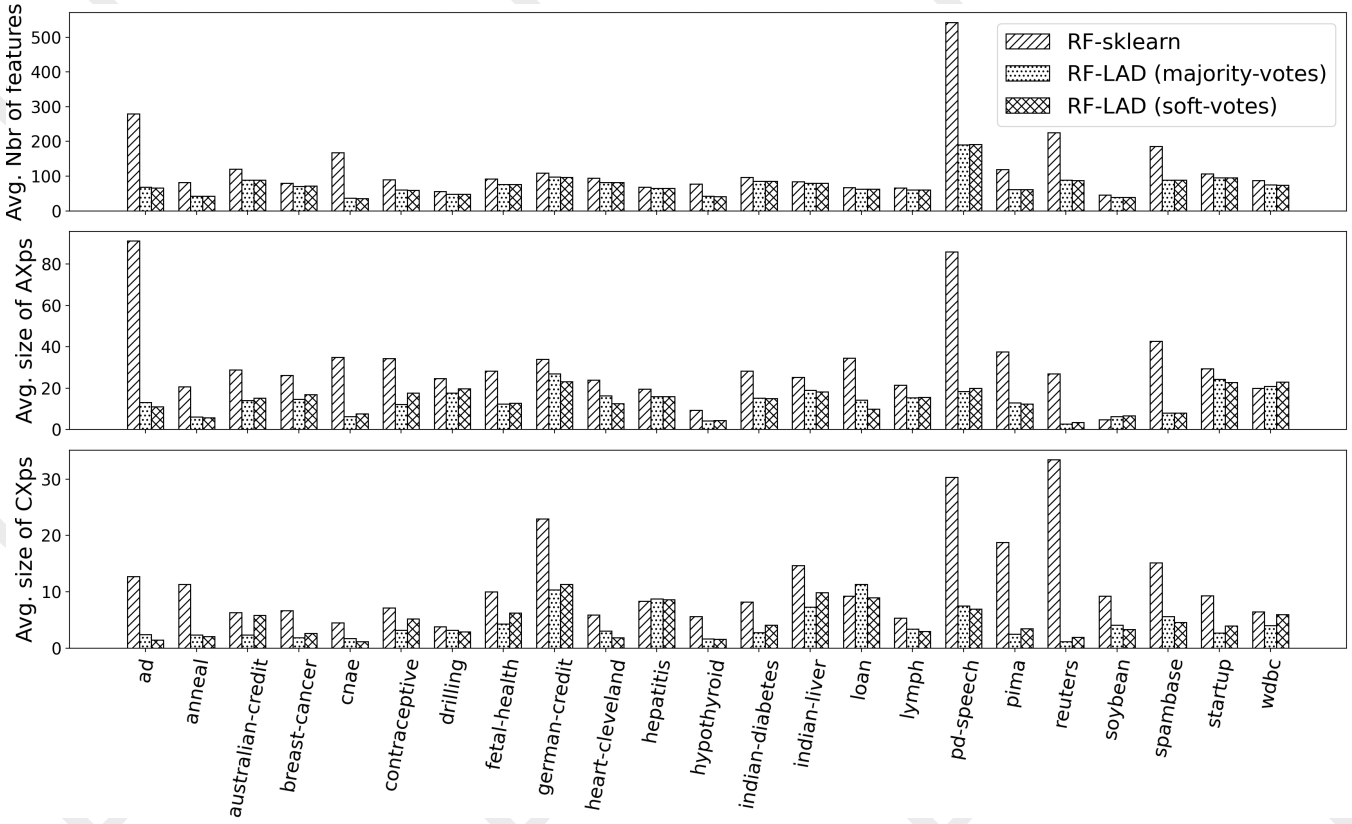

Figure 2: The average number of features (top histogram), the average size of AXps (middle histogram), and the average size of CXps (bottom histogram), when d=4.

ing fewer than 200, 200–999, and 1000–9999 examples, we randomly picked 40%, 20%, and 10% of the examples, respectively, to explain. To ensure a fair comparison in terms of the sizes of the explanations, for all datasets, we set a fixed randomized seed so that $RF_{sklearn}$ and RF-LAD(s) explain only the same examples. Finally, we report the average size of AXps and the average size of CXps. The middle histogram and the bottom histogram of Figure 2 represent the average size of AXps and the average size of CXps, respectively. The x-axis displays all the considered datasets, and the y-axis displays the average size of AXps (middle histogram) and the average size of CXps (bottom histogram). In these histograms, we can see that shorter AXps and CXps can be derived from RF-LAD(s) compared to $RF_{sklearn}$ for most considered datasets, except for a few datasets like "soybean" and "wdbc", where slightly longer AXps were derived from RF-LAD(s). Similarly, for datasets like "hepatitis" and "loan", slightly longer CXps were also derived from RF-LAD(s). Statistically, for RF-LAD(s), the average size of AXps are reduced by 17.72% to 90.28% and by 18.96% to 87.86% for RF_{LAD}(majority-vote) and RF_{LAD}(soft-vote), respectively. In a similar vein, the average size of CXps are reduced by 17.44% to 96.53% and by 3.23% to 94.14% for RF_{LAD}(majority-vote) and RF_{LAD}(soft-vote), respectively. Additionally, for at least 10 datasets and at least 14 datasets, the reduction in the average size of AXps and CXps exceeds 50%, respectively, which also shows a significant reduction.

To summarize, this section shows that building RFs using MSSes (RF-LAD) allows for a reduction in the set of used features while maintaining similar results in terms of prediction accuracy, produces forests of similar size with sometimes higher accuracy, and enables us to derive shorter explanations compared to the classical RF approach.

5 Conclusion and Future Works

In this paper, we have shown how to integrate the Logical Analysis of Data (LAD) framework to enhance classical random forests (RFs) using minimal support sets (MSSes) based on their discriminating property. Concretely, we have proposed to modify the classical RF algorithm by randomly choosing MSSes rather than randomly choosing feature subsets that likely contain irrelevant features for training bootstrap samples. Experiments on benchmark datasets show that our RF-LAD(s) are competitive with the classical RF in terms of prediction accuracy, while producing a forest of similar size, using limited features that lead to the extraction of shorter explanations compared to the classical RF.

In the future, we plan to extend LAD to handle numerical datasets directly, without relying on binarization techniques. Finally, it would be interesting to integrate LAD into other tree-based algorithms to evaluate whether it can also produce shorter and more concise explanations.

Acknowledgments

Many thanks to the anonymous reviewers for their insightful comments. This work has benefited from the support of the ANR HYCI Project (ANR-22-CE55-0010) and the support of the National Research Agency under France 2030 MAIA Project (ANR-22-EXES-0009).

References

[Audemard *et al.*, 2020] Gilles Audemard, Frédéric Koriche, and Pierre Marquis. On Tractable XAI Queries based on Compiled Representations. In *17th International Conference on Principles of Knowledge Representation and Reasoning (KR'20)*, 2020.

[Audemard *et al.*, 2022a] Gilles Audemard, Steve Bellart, Louenas Bounia, Frédéric Koriche, Jean-Marie Lagniez, and Pierre Marquis. On Preferred Abductive Explanations for Decision Trees and Random Forests. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 643–650. ijcai.org, 2022.

[Audemard *et al.*, 2022b] Gilles Audemard, Steve Bellart, Louenas Bounia, Frédéric Koriche, Jean-Marie Lagniez, and Pierre Marquis. Trading Complexity for Sparsity in Random Forest Explanations. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 5461–5469. AAAI Press, 2022.

[Audemard *et al.*, 2023] Gilles Audemard, Jean-Marie Lagniez, Pierre Marquis, and Nicolas Szczepanski. On Contrastive Explanations for Tree-Based Classifiers. In *ECAI 2023*, pages 117–124. IOS Press, 2023.

[Bailey *et al.*, 2003] James Bailey, Thomas Manoukian, and Kotagiri Ramamohanarao. A Fast Algorithm for Computing Hypergraph Transversals and its Application in Mining Emerging Patterns. In *Third IEEE International Conference on Data Mining*, pages 485–488. IEEE, 2003.

[Barredo Arrieta *et al.*, 2020] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020.

[Boros *et al.*, 2000] E. Boros, P.L. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik. An Implementation of Logical Analysis of Data. *IEEE Transactions on Knowledge and Data Engineering*, 12(2):292–306, 2000.

[Breiman *et al.*, 1984] L. Breiman, Jerome H. Friedman, Richard A. Olshen, and C. J. Stone. Classification and Regression Trees. 1984.

[Breiman, 2001] Leo Breiman. Random Forests. *Machine learning*, 45:5–32, 2001.

[Chen *et al.*, 2018] Chaofan Chen, Kangcheng Lin, Cynthia Rudin, Yaron Shaposhnik, Sijia Wang, and Tong Wang. An Interpretable Model with Globally Consistent Explanations for Credit Risk. *arXiv preprint arXiv:1811.12615*, 2018.

[Crama *et al.*, 1988] Yves Crama, Peter L. Hammer, and Toshihide Ibaraki. Cause-Effect Relationships and Partially Defined Boolean Functions. *Annals of Operations Research*, 16:299–325, 1988.

[Ding *et al.*, 2018] Lieyun Ding, Weili Fang, Hanbin Luo, Peter ED Love, Botao Zhong, and Xi Ouyang. A deep hybrid learning model to detect unsafe behavior: Integrating convolution neural networks and long short-term memory. *Automation in construction*, 86:118–124, 2018.

[Eiter *et al.*, 2008] Thomas Eiter, Kazuhisa Makino, and Georg Gottlob. Computational aspects of monotone dualization: A brief survey. *Discrete Applied Mathematics*, 156(11):2035–2049, 2008.

[Fayyad and Irani, 1993] Usama M Fayyad and Keki B Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *Ijcai*, volume 93, pages 1022–1029. Citeseer, 1993.

[Fredman and Khachiyan, 1996] Michael L. Fredman and Leonid Khachiyan. On the Complexity of Dualization of Monotone Disjunctive Normal Forms. *Journal of Algorithms*, 21(3):618–628, 1996.

[Gainer-Dewar and Vera-Licona, 2016] Andrew Gainer-Dewar and Paola Vera-Licona. The minimal hitting set generation problem: algorithms and computation. *CoRR*, abs/1601.02939, 2016.

[García *et al.*, 2013] Salvador García, Julián Luengo, José Antonio Sáez, Victoria López, and Francisco Herrera. A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):734–750, 2013.

[Guidotti *et al.*, 2018] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A Survey of Methods for Explaining Black Box Models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[Hammer and Bonates, 2006] Peter L Hammer and Tibérius O Bonates. Logical analysis of data—An overview: From combinatorial optimization to medical applications. *Annals of Operations Research*, 148(1):203–225, 2006.

[Hammer *et al.*, 2007] Peter L Hammer, Alexander Kogan, and Miguel A Lejeune. Reverse-Engineering Banks' Financial Strength Ratings Using Logical Analysis of Data. *Rutgers Center for OR*, 2007.

[Hammer, 1986] Peter L. Hammer. The Logic of Cause-effect Relationships. Lecture at the international conference on Multi-Attribute Decision Making via operation research based ExpertcSystems, Passau, Germany, 1986.

[Ideker *et al.*, 1999] Trey E Ideker, VESTEINN THORSSONt, and Richard M Karp. Discovery of regulatory interactions through perturbation: inference and experimental design. In *Biocomputing 2000*, pages 305–316. World Scientific, 1999.

[Ignatiev and Marques-Silva, 2021] Alexey Ignatiev and Joao Marques-Silva. SAT-Based Rigorous Explanations for Decision Lists. In *Theory and Applications of Satisfiability Testing–SAT 2021: 24th International Conference, Barcelona, Spain, July 5-9, 2021, Proceedings 24*, pages 251–269. Springer, 2021.

[Ignatiev *et al.*, 2019] Alexey Ignatiev, Nina Narodytska, and Joao Marques-Silva. Abduction-Based Explanations for Machine Learning Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1511–1519, 2019.

[Ignatiev *et al.*, 2020] Alexey Ignatiev, Nina Narodytska, Nicholas Asher, and Joao Marques-Silva. From Contrastive to Abductive Explanations and Back Again. In *International Conference of the Italian Association for Artificial Intelligence*, pages 335–355. Springer, 2020.

[Ing *et al.*, 2024] David Ing, Saïd Jabbour, Lakhdar Sais, and Fabien Delorme. LAD-based Feature Selection for Optimal Decision Trees and Other Classifiers. In Pierre Marquis, Magdalena Ortiz, and Maurice Pagnucco, editors, *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024*, 2024.

[Izza and Marques-Silva, 2021] Yacine Izza and João Marques-Silva. On Explaining Random Forests with SAT. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 2584–2591. ijcai.org, 2021.

[Izza *et al.*, 2023] Yacine Izza, Xuanxiang Huang, Alexey Ignatiev, Nina Narodytska, Martin C. Cooper, and João Marques-Silva. On Computing Probabilistic Abductive Explanations. *Int. J. Approx. Reason.*, 159:108939, 2023.

[Kotsiantis and Kanellopoulos, 2006] Sotiris Kotsiantis and Dimitris Kanellopoulos. Discretization Techniques: A recent survey. *GESTS International Transactions on Computer Science and Engineering*, 32(1):47–58, 2006.

[Lipton, 2018] Zachary C Lipton. The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[Miller, 1956] George A Miller. The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information. *Psychological review*, 63(2):81, 1956.

[Mortada *et al.*, 2012] Mohamad-Ali Mortada, Thomas Carroll, Soumaya Yacout, and Aouni Lakis. Rogue components: their effect and control using logical analysis of data. *Journal of Intelligent Manufacturing*, 23:289–302, 2012.

[Murakami and Uno, 2014] Keisuke Murakami and Takeaki Uno. Efficient algorithms for dualizing large-scale hypergraphs. *Discrete Applied Mathematics*, 170:83–94, 2014.

[Pedregosa *et al.*, 2011] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[Perner and Trautzsch, 1998] Petra Perner and Sascha Trautzsch. Multi-interval Discretization Methods for Decision Tree Learning. In Adnan Amin, Dov Dori, Pavel Pudil, and Herbert Freeman, editors, *Advances in Pattern Recognition*, pages 475–482, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[Ragab *et al.*, 2018] Ahmed Ragab, Mohamed El-Koujok, Bruno Poulin, Mouloud Amazouz, and Soumaya Yacout. Fault diagnosis in industrial chemical processes using interpretable patterns based on Logical Analysis of Data. *Expert Systems with Applications*, 95:368–383, 2018.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[Rudin and Ustun, 2018] Cynthia Rudin and Berk Ustun. Optimized Scoring Systems: Towards Trust in Machine Learning for Healthcare and Criminal Justice. *Interfaces*, 48(5):449–466, 2018.

[Rudin, 2019] Cynthia Rudin. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[Shone *et al.*, 2018] Nathan Shone, Tran Nguyen Ngoc, Vu Dinh Phai, and Qi Shi. A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1):41–50, 2018.

[Tian *et al.*, 2018] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars. In *Proceedings of the 40th International Conference on Software Engineering*, pages 303–314, 2018.

[Weng *et al.*, 2017] Stephen F Weng, Jenna Reps, Joe Kai, Jonathan M Garibaldi, and Nadeem Qureshi. Can machine-learning improve cardiovascular risk prediction using routine clinical data? *PloS one*, 12(4):e0174944, 2017.

[Yacout *et al.*, 2017] Soumaya Yacout, David Salamanca, and Mohamad-Ali Mortada. Tool and method for fault detection of devices by condition based maintenance, November 21 2017. US Patent 9,824,060.