Computing an Approximating Version of a Minimum-Size Explanation for Boolean Decision Tree Classifiers

Louenas Bounia

Computing an Approximating Version of a Minimum-Size Explanation for Boolean Decision Tree Classifiers

Louenas BOUNIA[1] [a]
[1]*LIPN-UMR CNRS 7030 Université Sorbonne Paris Nord, Villetaneuse, France*
{bounia@lipn.univ-paris13.fr}

Keywords: XAI (Explainable AI), Boolean Decision Tree, Abductive Explanations, Greedy Algorithm

Abstract: In this work, we tackle the problem of approximating minimum-size explanations for decision trees using a greedy algorithm with guarantees. Calculating a minimum-size abductive explanation can be a time-consuming task due to several factors. First, the combinatorial explosion of possible abductive explanations makes finding the minimum-size explanation extremely costly, even for restricted classifier families like decision trees. Indeed, finding a minimum-size abductive explanation for decision trees is an NP-**hard** problem, meaning that exact approaches can be very time-consuming, particularly for hard instances and high-dimensional inputs. This adds additional complexity and time to the process of finding the minimum-size explanation. Faced with these complexity challenges, approximate or heuristic approaches are often used to reduce computational load and obtain results more quickly, even if this comes at the cost of solution optimality. In this work, we propose a greedy algorithm to efficiently approximate minimum-size abductive explanations. Based on various experiments aimed at explaining decision tree predictions, we show that for difficult-to-explain instances, our greedy algorithm provides an effective alternative to exact approaches based on SAT encodings.

1 INTRODUCTION

Providing an explanation to someone involves delivering the necessary information for them to understand the reasons behind a decision, which is especially critical when it is generated by machine learning (ML) models such as decision trees, random forests, support vector machines, or deep neural networks (Adadi and Berrada, 2018; Miller, 2019; Molnar, 2019; Breiman, 2001). With the rise of applications utilizing these techniques, research into explainable artificial intelligence (XAI) has become increasingly important. XAI methods are generally categorized into model-agnostic approaches, with popular methods including LIME (Ribeiro et al., 2016), SHAP (Lundberg and Lee, 2017), and ANCHORS (Ribeiro et al., 2018). However, these methods have limitations regarding reliability. For instance, (Ignatiev et al., 2019) showed that the same explanation can be compatible with multiple predicted classes, which poses a problem in sensitive domains like medicine, cybersecurity, or even finance. This highlights the growing interest in methods offering stronger mathematical guarantees, such as formal methods, which address this weakness and provide valid explanations (Marques-Silva, 2023; Darwiche and Hirth, 2020; Shih et al., 2018).

In this paper, we focus on finding minimum-size explanations to explain the classification of an input instance $\boldsymbol{x}$ by a binary classifier. We emphasize abductive explanations, which answer the question why this classification?. Although there is no formal definition of interpretability (Lipton, 2018), decision trees (Breiman, 2001; Quinlan, 1986) are widely regarded as one of the most interpretable models for classification. They are often used to distill an opaque model into an understandable one (Frosst and Hinton, 2017) and form the basis of more complex models, such as random forests (RF) (Breiman, 2001) and boosted trees (XGBOOST) (Chen and Guestrin, 2016). The interpretability of decision trees lies in their transparency, as each node has a clear meaning, and in their ability to provide local explanations. From the instance to be classified, it is easy to extract an abductive explanation, called a direct reason (Audemard et al., 2022; Louenas, 2023). However, direct reasons may include redundant features (Izza et al., 2020, 2022), which justifies interest in non-redundant ex-

[a] https://orcid.org/0009-0006-8771-0401

planations, such as sufficient reasons (Darwiche and Hirth, 2020) (or PI-explanations (Shih et al., 2018)) or minimum-size sufficient reasons.

Explanations involving too many features can be difficult to interpret and use. Therefore, it is crucial to compute minimum-size sufficient reasons, as they offer more concise and understandable explanations for humain-users . By identifying the smallest subset of features sufficient to justify a decision, we improve the model's intelligibility and transparency while respecting users' cognitive limitations. However, computing a minimaum-size abductive explanation is generally very complex. For example, the problem of finding a minimum-size sufficient reason for random forests is DP-complete (Izza and Marques-Silva, 2021), and NP-hard for decision trees (Audemard et al., 2022). Consequently, this paper explores the approximation of these explanations when the classifier is a decision tree.

Contributions. In this work, we focus on approximating minimum-size sufficient reasons for an instance $\boldsymbol{x}$ given a classifier h represented by a decision tree T. While deriving explanations from a decision tree is computationally feasible (see (Audemard et al., 2021)), when the tree or the input instance is of high dimensionality, computing a minimum-size sufficient reason can become challenging in terms of computation time, due to the complexity of this NP-hard problem. That is, for difficult instances, deriving a minimum-size explanation may take significant time. To address this challenge, we formulate the problem of computing minimum-size abductive explanations as finding a minimal set of leaders for a supermodular function (the unnormalized error function $\mu_{h,x}(.)$) that reaches an error bound α. We also propose a greedy algorithm to compute efficient approximations of this problem, offering mathematical guarantees on the size of the produced solution. We empirically compare our approaches to two benchmark SAT encodings (Audemard et al., 2022; Arenas et al., 2022). Our results demonstrate the efficiency of our method in approximating minimum-size sufficient reasons, particularly for difficult instances, while highlighting the empirical performance of our algorithms.

2 Decision Tree, Abductive Explanations

Preliminaries. For an integer n, let $[n]$ denote the set $\{1,\ldots,n\}$. We denote by $\mathcal{F}_n$ the class of all Boolean functions from $\{0,1\}^n$ to $\{0,1\}$, and $X_n = \{x_1,\ldots,x_n\}$ refers to the set of Boolean variables. Any assignment $\boldsymbol{x} \in \{0,1\}^n$ is called an *instance*. A *literal* ℓ is a variable x_i or its negation $\bar{x}_i$. A *term* t is a conjunction of literals[1], and a *clause* c is a disjunction of literals. A DNF formula is a disjunction of terms, and a CNF formula is a conjunction of clauses. A formula f is *consistent* if and only if it has a model. Given an instance $z \in \{0,1\}^n$, the corresponding term is defined as follows: $t_z = \bigwedge_{i=1}^{n} x_i^{z_i} = \{x_1^{z_1},\cdots,x_n^{z_n}\}$ où $x_i^0 = \bar{x}_i$ et $x_i^1 = x_i$

An implicant of a Boolean function f is a term that implies f. A prime implicant of f is an implicant t of f such that no subset of t is an implicant of f. A partial instance is a vector $\boldsymbol{z} \in \{0,1,*\}^n$, where $z_i = *$ indicates that the i-th feature of $\boldsymbol{z}$ is undefined. An instance $\boldsymbol{x}$ is covered by $\boldsymbol{z}$ if $x_i = z_i$ for all features $i \in [n]$ such that $z_i \neq *$. For a subset S of features, the restriction of $\boldsymbol{x}$ to S, denoted x_S, is the partial instance in $\{0,1,*\}^n$ such that for each $i \in [n]$, $(x_S)_i = x_i$ if $i \in S$, and $(x_S)_i = *$ otherwise. Any instance $y \in \{0,1\}^n$ is covered by x_S if and only if $y_S = x_S$.

2.1 Decision Tree

Boolean Decision Tree. A binary decision tree on X_n is a binary tree T, where each internal node is labeled with one of the n input Boolean variables from X_n, and the leaves are labeled with 0 or 1. It is assumed that each variable appears at most once on any root-to-leaf path (read-once property). The value $T(\boldsymbol{x}) \in \{0,1\}$ of T on an input instance $\boldsymbol{x}$ is given by the label of the leaf reached from the root as follows at each node. The size of T, denoted $|T|$, is given by the number of its nodes. The class of decision trees is denoted $\mathtt{DT}_n$.

It is well known that any decision tree $T \in \mathtt{DT}_n$ can be transformed in linear time into an equivalent disjunction of terms, denoted $\mathtt{DNF}(T)$. This DNF is an orthogonal DNF, where each term corresponds to a path from the root to a leaf labeled 1. T can also be transformed into a conjunction of clauses, denoted $\mathtt{CNF}(T)$ (Audemard et al., 2022).

Definition 1 (DNF orthogonal)**.** *A* DNF $\phi = \{t_1,\cdots,t_m\}$ *is orthogonal if and only if* $\forall i,j \in [m]$*, if* $i \neq j$ *then* $t_i \cup t_j$ *is inconsistent.*

Definition 1 (Crama and Hammer, 2011) states that the terms t_i and t_j are mutually exclusive, meaning they do not share any common models where both are true. Due to the specific structure of orthogonal DNFs, counting the models can be done in linear time

[1] In this work, we specify that a term can be viewed as a set of literals

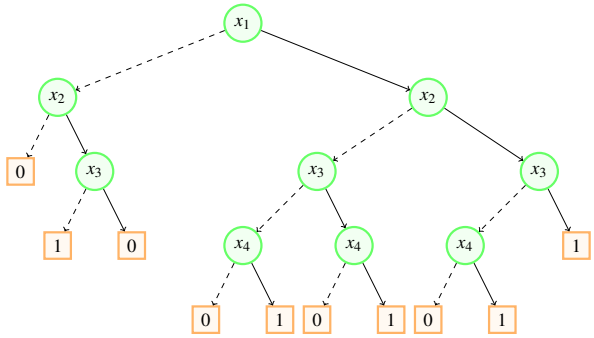


Figure 1: A decision tree T on the features $\{x_1, x_2, x_3, x_4\}$.

(in $O(m)$). The total number of models is given by: $w(\phi) = \sum_{k=1}^{m} 2^{n-|t_k|}$.

Example 1. *The decision tree in Figure 1 distinguishes Cattleya orchids from other orchids using the following features: x_1: "has fragrant flowers," x_2: "has one or two leaves," x_3: "has large flowers," and x_4: "is sympodial."*

For a tree $T \in \mathtt{DT}_n$, an instance $\boldsymbol{x} \in \{0,1\}^n$, and a set S of features, let t_{x_S} be the term associated with the partial instance x_S, that is:

$$t_{x_S} = \bigcup_{i=1}^{n} (\{x_i : (x_S)_i = 1\} \cup \{\overline{x_i} : (x_S)_i = 0\}).$$

By construction, the conditioning $\mathtt{DNF}(T) \mid t_{x_S}$ is an orthogonal DNF (Darwiche, 1999), and in conjunction with the fact that $\mathtt{DNF}(T) \mid t_{x_S}$ can be derived in time $O(|S| \cdot |T|)$, we conclude that the number of models of $\mathtt{DNF}(T) \mid t_{x_S}$ can be found in $O(|S| \cdot |T|)$ (Louenas, 2024).

2.2 Abductive Explanations

An abductive explanation for an instance $\boldsymbol{x}$ is a subset of features S such that the restriction of the instance $\boldsymbol{x}$ to S is sufficient to obtain the same prediction as $\boldsymbol{x}$. Decision trees are locally explainable. By construction, each instance $\boldsymbol{x}$ is associated with a unique path from the root to the leaf in a decision tree. A direct reason (Audemard et al., 2022) or path explanation (Izza et al., 2020) for $\boldsymbol{x}$, given the tree T, is a term from the $\mathtt{DNF}(T)$, denoted $p_{\boldsymbol{x}}^T$, corresponding to the unique path from the root to the leaf of T. However, these direct reasons may contain an arbitrary number of redundant features (Izza et al., 2020). This justifies the consideration of other types of non-redundant abductive explanations for decision trees, such as sufficient reasons (Darwiche and Hirth, 2020) and minimum-size sufficient reasons. Formally,

Definition 2 (Sufficient Reason)**.** *Let $h : \{0,1\}^n \to \{0,1\}$ be a classifier and $\boldsymbol{x} \in \{0,1\}^n$ such that $h(\boldsymbol{x}) = 1$. A sufficient reason for $\boldsymbol{x}$ given h is a prime implicant t of h that covers $\boldsymbol{x}$. **A minimum-size sufficient reason** for $\boldsymbol{x}$ given h is a sufficient reason for $\boldsymbol{x}$ given h that contains a minimal number of literals.*

Sufficient reasons (Darwiche and Hirth, 2020) (or PI-explanations (Shih et al., 2018)) are minimal abductive explanations concerning set inclusion, and it has been demonstrated that, for the class $\mathtt{DT}_n$, sufficient reasons can be found in polynomial time (see (Izza et al., 2020; Audemard et al., 2022)). It is often relevant to focus on the shortest ones (minimum-size sufficient reasons), as conciseness is generally a desirable property of explanations (the principle of Occam's razor). However, finding a minimum-size sufficient reason proves to be challenging, especially for difficult instances.

Proposition 1. *Let $T \in \mathtt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$. Calculating a minimum-size sufficient reason for $\boldsymbol{x}$ given T is an* NP*-hard problem.*

Despite this result, it is possible, in many practical cases, to compute a minimum-size sufficient reason. To do this, Audemard et al. (2022) have leveraged recent advances in combinatorial optimization, particularly through the use of PARTIAL MAXSAT solvers, which allow for the calculation of minimum-size sufficient reasons. However, when the size of the tree T and/or the dimension of the instance $\boldsymbol{x}$ is large, calculating a minimum-size sufficient reason can become out of reach. Even with modern solvers, the response time can be extremely long. This is why we turn to approximating minimum-size sufficient reasons using supermodularity.

Example 2. *Let ϕ be the* DNF *representing the tree in Figure 1 and the instance $\boldsymbol{x} = (1,1,1,1)$ such that $T(\boldsymbol{x}) = 1$. We have $\phi = (x_1 \wedge x_2 \wedge x_3) \vee (\overline{x_1} \wedge x_2 \wedge \overline{x_3}) \vee (x_1 \wedge x_2 \wedge \overline{x_3} \wedge x_4) \vee (x_1 \wedge \overline{x_2} \wedge x_3 \wedge x_4) \vee (x_1 \wedge \overline{x_2} \wedge \overline{x_3} \wedge x_4)$ Furthermore,*

$$w(T) = w(\phi) = 2^1 + 2^1 + 2^0 + 2^0 + 2^0 = 4 + 3 = 7.$$

For the instance $\boldsymbol{x} = (1,1,1,1)$, we observe that $T(\boldsymbol{x}) = 1$. The direct reason for $\boldsymbol{x}$ given T is $p_{\boldsymbol{x}}^T = x_1 \wedge x_2 \wedge x_3$. $x_1 \wedge x_4$ and $x_1 \wedge x_2 \wedge x_3$ are two sufficient reasons for $\boldsymbol{x}$. $x_1 \wedge x_4$ is the unique minimum-size sufficient reason for $\boldsymbol{x}$ given T.

3 Problem Formulation

The notion of a minimum-size sufficient reason t_{x_S} is often considered a natural concept for explaining the

output of a classifier. However, it imposes two strict restrictions: on the one hand, it requires that all completions of a partial instance $\boldsymbol{x}_S$ be classified in the same way, meaning that the probability of making an *explanation error* by using the partial instance $\boldsymbol{x}_S$ instead of the complete instance $\boldsymbol{x}$ is equal to zero. On the other hand, it imposes that the size of t_{x_S} (or simply S) be minimal in terms of cardinality, knowing that the number of abductive explanations for a given instance, in the case of a decision tree, can be exponential (Audemard et al., 2022).

We then define concepts that will be useful for the rest of this work, notably the explanation error function $\varepsilon_{h,\boldsymbol{x}}(S)$ given a classifier h and an instance $\boldsymbol{x}$, which can be interpreted as the probability of making an "explanation error" by using a subset S of features. Additionally, we introduce the notion of a supermodular function, which will be central to our approximation study. Given a classifier h and an instance $\boldsymbol{x}$ for which the prediction $h(\boldsymbol{x})$ needs to be explained, let $\varepsilon_{h,\boldsymbol{x}} : 2^{[n]} \rightarrow \mathbb{R}$ be the *explanation error function* (Bounia and Koriche, 2023) defined by

$$\varepsilon_{h,\boldsymbol{x}}(S) = \frac{|\{\boldsymbol{y} \in \{0,1\}^n : h(\boldsymbol{y}) \neq h(\boldsymbol{x}), \boldsymbol{y}_S = \boldsymbol{x}_S\}|}{|\{\boldsymbol{y} \in \{0,1\}^n : \boldsymbol{y}_S = \boldsymbol{x}_S\}|} \quad (1)$$

$$= \frac{\mu_{h,\boldsymbol{x}}(S)}{2^{n-|S|}} \quad (2)$$

As indicated above, $\varepsilon_{h,\boldsymbol{x}}(S)$ can be interpreted as the probability of making an *explanation error*, where $\mu_{h,\boldsymbol{x}}(S)$ is interpreted as the number of errors induced by the choice of S. For a subset of features S, t_{x_S} is an abductive explanation for $\boldsymbol{x}$, given h, if $\varepsilon_{h,\boldsymbol{x}}(S) = 0$. Moreover, t_{x_S} is a sufficient reason if $\varepsilon_{h,\boldsymbol{x}}(S) = 0$ and $\varepsilon_{h,\boldsymbol{x}}(S') > 0$ for every proper subset S' of S. Note that when h is represented by a decision tree T, the function $\mu_{h,\boldsymbol{x}}$ in (1) can be rewritten simply as follows:

$$\mu_{h,\boldsymbol{x}}(S) = \begin{cases} \sum\limits_{t \in \mathtt{DNF}(T) | t_{x_S}} 2^{n-|t|} & \text{si } h(\boldsymbol{x}) = 0 \\ 2^{n-|S|} - \sum\limits_{t \in \mathtt{DNF}(T) | t_{x_S}} 2^{n-|t|} & \text{si } h(\boldsymbol{x}) = 1 \end{cases}$$

The result shows that the evaluation of $\varepsilon_{h,\boldsymbol{x}}(S)$ can be performed in $O(|S| \cdot |T|)$ time when h is presented by a decision tree T, which is not always the case in general. Indeed, in the general case, the problem of evaluating $\varepsilon_{h,\boldsymbol{x}}(S)$ is **#-hard** (Wäldchen et al., 2021).

We now formulate the problem of finding a minimum-size sufficient reason. The idea behind this is based on the fact that a term t_{x_S} associated with a subset of features $S \subseteq [n]$ is an abductive explanation for $\boldsymbol{x}$ given h if $\varepsilon_{h,\boldsymbol{x}}(S) = 0$, which is equivalent to $\mu_{h,\boldsymbol{x}}(S) = 0$. Thus, by definition, a minimum-size abductive explanation[2] is simply the smallest set in cardinality that satisfies $\mu_{h,\boldsymbol{x}}(S) = 0$.

Approximation of a minimum-size sufficient reason. The problem of finding a minimum-size sufficient reason for an instance $\boldsymbol{x}$, given a classifier h, can be formulated as a leader selection problem (or k-leaders). More specifically, our goal is to select a subset S of leaders of minimum-size that satisfies a certain constraint on the explanation error function $\varepsilon_{h,\boldsymbol{x}}(.) \leq 0$, which is equivalent to writing $\mu_{h,\boldsymbol{x}}(.) \leq 0$ (or simply $\mu_{h,\boldsymbol{x}}(.) = 0$).

Problem 1. *Given a classifier h and an instance $\boldsymbol{x}$, the problem of finding a minimum-size sufficient reason $\boldsymbol{x}$ given $\boldsymbol{x}$ can be formulated as follows:*

$$\min_{S \subseteq [n]} \ |S| \qquad (\mathbb{P}_1)$$

$$\textit{s.t.} \quad \mu_{h,\boldsymbol{x}}(S) \leq 0$$

Proposition 2. *Let $h : \{0,1\}^n \rightarrow \{0,1\}$ be a classifier and $\boldsymbol{x} \in \{0,1\}^n$ be an instance, along with a subset $S^* \subseteq [n]$ of features. S^* is an optimal solution of problem 1 if and only if $t_{x_{S^*}}$ is a minimum-size sufficient reason for $\boldsymbol{x}$ given h.*

In general, selecting a minimum-size set of leaders for a supermodular function, as in problem 1, is NP-**hard** (Clark et al., 2014b). This is consistent with proposition 2 and the fact that calculating a minimum-size sufficient reason is also NP-**hard** when h is represented by a decision tree. Despite this intractability, it is possible to obtain an efficient approximation of the minimum-size sufficient reason by leveraging supermodular optimization. In particular, the function $\mu_{h,x}(.)$ is a decreasing supermodular function (Bounia and Koriche, 2023), which allows for the computation of a set of leader features. We propose an algorithm that effectively approximates this optimal set of leaders, guaranteeing a performance gap between the size of the set selected by the algorithm and that of the smallest set of leaders S^* that satisfies the error constraint $\mu_{h,x}(S^*) = 0$.

4 Approximation Algorithms

The main idea of this study is to relax the requirement of finding an optimal solution to problem 1 and

[2] A minimum-size abductive explanation is a minimum-size suffcient reason.

to settle for a **sufficiently good** solution by using supermodular minimization algorithms. We first introduce the basic concepts of supermodularity and then examine some useful properties.

4.1 Supermodular Functions

Let $f : 2^{[n]} \to \mathbb{R}$ be a real-valued set function. f is said to be non-decreasing if $f(S \cup \{i\}) \geq f(S) \quad$ for all $S \subseteq [n]$ and $i \in [n] \setminus S$, and non-increasing if $f(S \cup \{i\}) \leq f(S) \quad$ for all $S \subseteq [n]$ and $i \in [n] \setminus S$.

f is supermodular if it satisfies the following condition for all subsets A, B of $[n]$:

$$f(A \cup B) + f(A \cap B) \geq f(A) + f(B).$$

Dually, f is submodular if, for all subsets A and B of $[n]$, the following condition is satisfied:

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B).$$

For any $S \subseteq [n]$ and $i \in S$, f is supermodular if and only if $-f$ is submodular. Finally, f is modular if it is both submodular and supermodular.

In general, the error function $\varepsilon_{h,x}(\cdot)$ is neither supermodular nor submodular, and it is neither increasing nor decreasing (Bounia and Koriche, 2023). However, if we focus on the unnormalized version $\mu_{h,x}(\cdot)$, the following properties can be derived.

Proposition 3 (Bounia and Koriche (2023))**.** *Let $h : \{0,1\}^n \to \{0,1\}$ be a classifier, and let $x \in \{0,1\}^n$ be an instance. Then, $\mu_{h,x}$ is a positive, supermodular, and decreasing function.*

Proposition 3 implies that problem 1 is a supermodular optimization problem. Although supermodular optimization problems of this form are generally NP-hard (Clark et al., 2014b), a greedy algorithm will return a set S whose size of S is close to that of the size of the optimal solution S^*.

4.2 Greedy Algorithm

The supermodularity of $\mu_{h,x}(.)$ allows for the efficient computation of an approximate solution to problem 1 using a greedy algorithm. The algorithm begins by initializing the set of leader features to $S_0 = \emptyset$. At each iteration, the element e^i that maximizes $\mu_{h,x}(S) - \mu_{h,x}(S_{i-1} \cup \{e^i\})$ is added, such that $S_i = S_{i-1} \cup \{e^i\}$. The algorithm terminates when $\mu_{h,x}(S_k) \leq \alpha$ and returns the set $S = S_i$. A pseudo-code description of this algorithm is provided under the name `static-`α (Clark et al., 2014a).

Theorem 1 provides the approximation bounds for the solution returned by algorithm 1.

Algorithm 1

Input: a classifier h represented by a tree T, an instance $\boldsymbol{x}$, termination error α

Output: a set of leader features S (a justification of $\boldsymbol{x}$ given T)

Initialization: $S \leftarrow \emptyset$, error $\leftarrow \alpha$ ($\alpha > 0$), $V = [n]$

while *error* > 0 **do**
  $e^* \longleftarrow \underset{e \in V \setminus S}{\text{argmax}}\ \mu_{h,x}(S) - \mu_{h,x}(S \cup \{e\})$
  if $\mu_{h,x}(S) - \mu_{h,x}(S \cup \{e\}) \leq 0$ **then**
    return S; **exit**
  end
  else
    $S \longleftarrow S \cup \{e^*\}$
    error $\longleftarrow \mu_{h,x}(S)$
  end
end

return S; **exit**

Theorem 1. *Let $h : \{0,1\}^n \to \{0,1\}$ be a classifier and $\boldsymbol{x} \in \{0,1\}^n$ an instance. Let $|S^*| = k^*$ be the size of optimal solution of problem 1, and let $|S| = k$ be the set returned by algorithm 1. Then,*

$$\frac{|S|}{|S^*|} = \frac{k}{k^*} \leq 1 + \ln\left(\frac{\mu_{max}}{\mu_{h,x}(S_{k-1})}\right).$$

where $\mu_{max} = \max_{i \in [n]} \mu_{h,x}(\{i\})$.

Theorem 1 provides a size approximation bound, guaranteeing that the size of the solution S found by the algorithm 1 is close to the size of the optimal solution. However, even though this solution is an abductive (but not necessarily minimal for inclusion) explanation of the instance $\boldsymbol{x}$ given h, it may include irrelevant features, which means that it is not necessarily a sufficient reason (a minimal abductive explanation for inclusion). We also note that by the construction of the output S_k of algorithm 1, the value of $\mu_{h,x}(S_{k-1})$ cannot be zero because μ is a non-increasing function.

Proposition 4. *Let h be a classifier represented by a decision tree $T \in \mathtt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$ an instance. Algorithm 1 runs in time $O(n^3 \cdot |T|)$.*

Example 3. *For the decision tree in Figure 1. Let $\boldsymbol{x} = (1,1,1,1)$. We intialize $S \leftarrow \emptyset$ and run Algorithm 1. The steps of Algorithm 1:*

$$x_1 = \underset{e \in \{x_1, x_2, x_3, x_4\}}{\text{argmax}}\ \mu_{h,x}(\emptyset) - \mu_{h,x}(\{e\})$$

Then $S = \{x_1\}$ and:

$$x_4 = \underset{e \in \{x_2, x_3, x_4\}}{\operatorname{argmax}} \ \mu_{h,x}(S) - \mu_{h,x}(S \cup \{e\})$$

We obtain $S = \{x_1, x_4\}$ *and* $\frac{\mu_{h,x}(\{x_1,x_4\})}{2^{4-2}} = 0$. *Thus, Algorithm 1 has captured the unique minimum-size sufficient reason for* $\boldsymbol{x}$ *given h.*

Example 3 shows that the algorithm 1 captured the optimal solution to problem $\mathbb{P}_1$, namely a minimum-size sufficient reason. The output of algorithm 1 is not even guaranteed to be a sufficient reason for $\boldsymbol{x}$ given h. However, we propose another greedy algorithm to derive a sufficient reason from the output of algorithm 1 as a sort of improvement.

Improvement of the output from algorithm 1. We know that the output of algorithm 1 does not necessarily constitute a sufficient reason. To improve the output of our algorithm, we can extract a sufficient reason from the solution S returned by algorithm 1 using a simple greedy algorithm described as follows.

Algorithm 2 : Deriving a Sufficient Reason

Input: a classifier h represented by a tree T, $\boldsymbol{x} \in \{0,1\}^n$, a set S

Output: a sufficient reason for $\boldsymbol{x}$ given T

$I \leftarrow S$ /*S is the output of algorithm 1 */

for $l \in I$ **do**
  if $\mu_{h,x}(S) = 0$ **then**
    $S \leftarrow S - \{l\}$
  end
end

return S

This algorithm 2 aims to derive a sufficient reason from the output of a previous algorithm (Algorithm 1). It iterates through the elements of the set I (initialized to S, the output of Algorithm 1), and removes elements that do not contribute to a sufficient reason. Specifically, it removes any element l if the measure $\mu_{h,x}(S)$ indicates that the set S without l still holds a valid reason. The final result is a sufficient reason for the classification of $\boldsymbol{x}$ under h.

Lemma 1. *Let h be a classifier represented by a binary decision tree* $T \in \mathtt{DT}_n$ *and let* $\boldsymbol{x} \in \{0,1\}^n$ *be an instance. Let S denote the output of Algorithm 1. Algorithm 2 runs in time* $O(|S| \cdot |T|)$, *and the term* t_{x_S} *associated with its output is a sufficient reason for* $\boldsymbol{x}$ *given h.*

5 Experiments

In order to validate the efficiency of our algorithms, we considered various instances of Problem 1, where the input classifier h is described by a decision tree T. The code was written in Python. All experiments were conducted on a computer equipped with an Intel(R) Core i9-9900 processor at 3.1 GHz and 64 GiB of RAM.

We conducted several experiments primarily to evaluate the performance of Algorithm 1. Our objectives are as follows:

- We measure the accuracy of the results from Algorithm 1 in calculating an efficient approximation of the optimal solution to Problem 1 (a minimum-size sufficient reason). Specifically, our goal is to compare the average size of the optimal solution to Problem 1, calculated using an exact method, with the approximate solution provided by Algorithm 1 and its improved version, Algorithm 2.
- We also demonstrate the effectiveness of our approach on hard instances by comparing the computation times between SAT methods and Algorithm 1. The results clearly show that Algorithm 1 is a very efficient alternative, particularly when SAT methods become inefficient in terms of computation time.

5.1 Experimental Protocol

We studied a set of $B = 25$ datasets from recognized sources such as Kaggle[3], OpenML[4], and UCI[5]. Categorical features were treated as integers, and numerical features were binarized during the training of decision trees. All datasets pertain to binary classification tasks, with the number of features ranging from 10^1 to 10^5. Multi-label classification tasks were converted to binary classification by considering the dominant label versus all other labels. In order to present a case with numerous challenging instances to demonstrate the effectiveness of our algorithms, a specific preprocessing step for *sentiment140* was carried out.

For each instance $\boldsymbol{x}$ from the test set of a benchmark b, an explanation task consists of a pair $(T_b, \boldsymbol{x})$, where T_b is the decision tree representing a classifier h, learned from the training set of b using the CART algorithm via the *Scikit-Learn* implementation (Pedregosa et al., 2011). The accuracy of T_b is evaluated on the test set of b. To assess the performance

[3] www.kaggle.com
[4] www.openml.org
[5] archive.ics.uci.edu/ml/

dataset			decision tree			|Explanation|		
name	#F	#I	%A	$\|T\|$	Depth	S^*	S_{algo1}	S_{improve}
placement	10	215	92.31	11	6	3.71 ($\pm$ 0.99)	3.71 ($\pm$ 0.99)	3.71 ($\pm$ 0.99)
letter	91	20000	99.08	144	15	6.76 ($\pm$ 1.33)	6.82 ($\pm$ 1.41)	6.81 ($\pm$ 1.4)
diabetes t	101	768	68.83	107	15	7.85 ($\pm$ 2.36)	7.85 ($\pm$ 2.36)	7.85 ($\pm$ 2.36)
student.por	27	649	89.23	30	8	4.22 ($\pm$ 0.92)	4.22 ($\pm$ 0.92)	4.22 ($\pm$ 0.92)
ad-data	115	3279	96.44	135	83	28.31 ($\pm$ 9.77)	29.06 ($\pm$ 10.13)	29.03 ($\pm$ 10.12)
balance	16	625	85.64	88	11	4.89 ($\pm$ 0.84)	5.04 ($\pm$ 0.95)	5.04 ($\pm$ 0.95)
breast. c	31	286	62.79	80	16	7.15 ($\pm$ 2.58)	7.51 ($\pm$ 3.03)	7.17 ($\pm$ 2.58)
christine	327	5418	62.12	328	38	15.77 ($\pm$ 9.87)	15.77 ($\pm$ 9.87)	15.77 ($\pm$ 9.87)
haberman	55	306	69.57	75	13	6.57 ($\pm$ 1.33)	6.59 ($\pm$ 1.33)	6.59 ($\pm$ 1.33)
spambase	219	4601	90.51	222	32	16.46 ($\pm$ 6.87)	16.46 ($\pm$ 6.87)	16.46 ($\pm$ 6.87)
meta	40	528	90.57	66	15	4.13 ($\pm$ 1.42)	4.14 ($\pm$ 1.43)	4.14 ($\pm$ 1.43)
bank	805	41188	88.93	2090	25	14.06 ($\pm$ 2.21)	14.1 ($\pm$ 2.23)	14.08 ($\pm$ 2.22)
heart	32	303	73.63	35	7	5.29 ($\pm$ 0.75)	5.29 ($\pm$ 0.75)	5.29 ($\pm$ 0.75)
mnist49	238	13782	95.60	240	30	21.08 ($\pm$ 7.31)	21.08 ($\pm$ 7.31)	21.08 ($\pm$ 7.31)
weather	1843	39716	79.49	2930	24	14.45 ($\pm$ 2.95)	14.46 ($\pm$ 2.95)	14.45 ($\pm$ 2.95)
christine	330	5418	64.82	331	26	14.91 ($\pm$ 5.89)	14.91 ($\pm$ 5.89)	14.91 ($\pm$ 5.89)
diabetes	94	768	73.16	98	13	6.59 ($\pm$ 1.98)	6.59 ($\pm$ 1.98)	6.59 ($\pm$ 1.98)
gina	136	3153	86.05	139	22	14.06 ($\pm$ 6.02)	14.06 ($\pm$ 6.02)	14.06 ($\pm$ 6.02)
tic-tac-toe	9	958	98.96	77	9	4.21 ($\pm$ 1.04)	4.39 ($\pm$ 1.15)	4.29 ($\pm$ 1.1)
vote	14	434	96.95	22	7	2.74 ($\pm$ 0.9)	2.89 ($\pm$ 1.02)	2.89 ($\pm$ 1.02)
horse	24	299	73.33	27	8	5.22 ($\pm$ 1.36)	5.23 ($\pm$ 1.37)	5.22 ($\pm$ 1.36)

Table 1: Statistics on the reliability of the approximate solutions to problem $\mathbb{P}_{\mathbb{1}}$ generated by Algorithm 1.

of Algorithm 1 in deriving an approximate solution to Problem 1, we randomly select m instances $\boldsymbol{x}$ from the test set of b (with $m = \min(q, 1000)$, where q is the size of the test set). We then compare the average sizes of the approximate solutions to those of the various types of abductive explanations for decision trees (see 1 and 2).

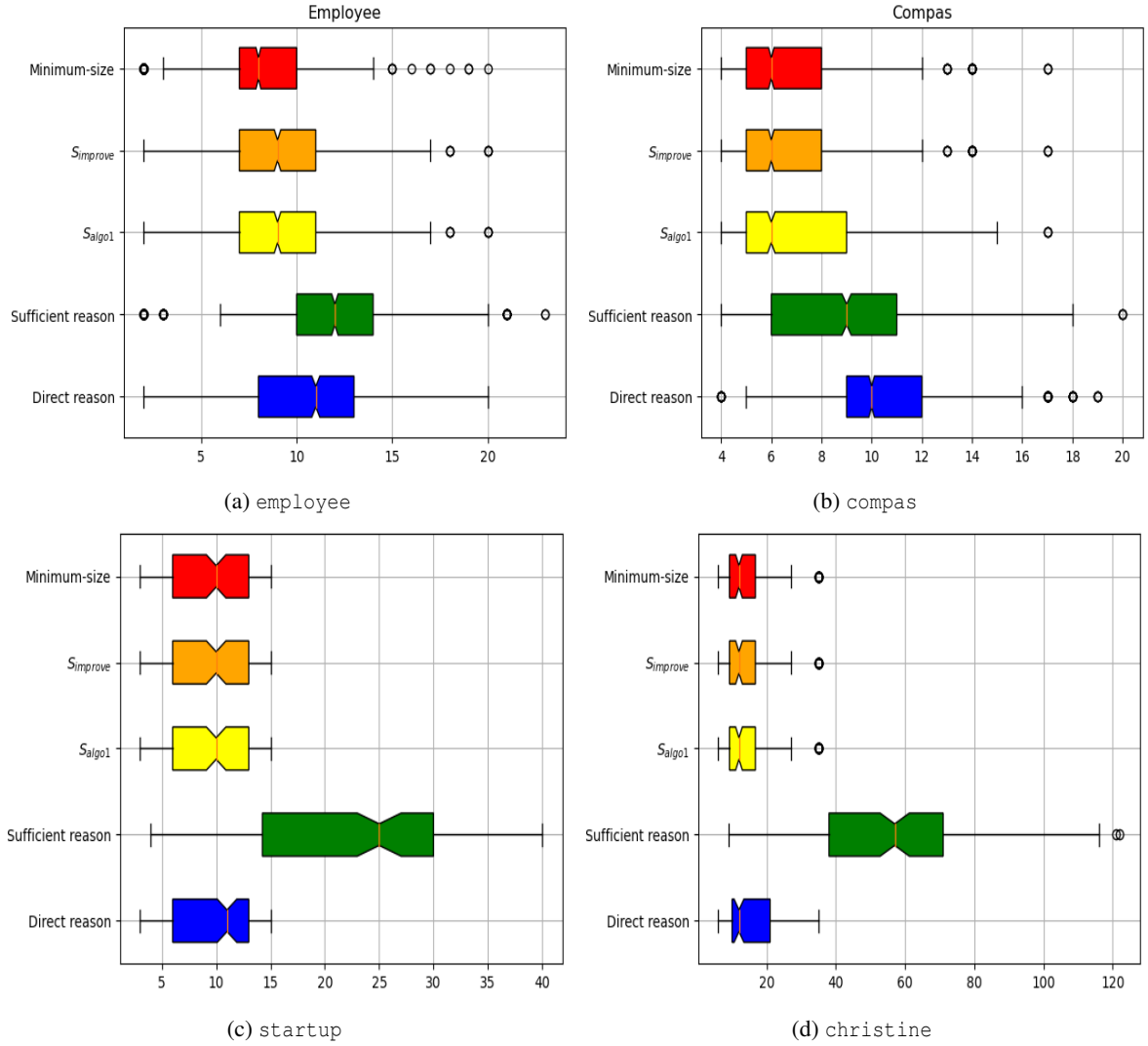
To evaluate the performance of our greedy algorithms, we compared the solutions returned by Algorithm 1 and Algorithm 2 to those from an exact method for solving Problem 1. We used the method described in (Audemard et al., 2022), which relies on the PARTIAL MAXSAT solver *RC*2, via its implementation in the **Pyxai** library (Audemard et al., 2023), to obtain a minimum-size sufficient reason for each instance $\boldsymbol{x}$ given T_b. To assess the reliability of our approximate solutions, we compared the average sizes of the optimal solution S^* to those of the arbitrary sufficient reason S_r, the direct reason (Audemard et al., 2022) (or path explanation (Izza et al., 2020)) P_x^T, as well as the solutions provided by Algorithm 1 (S_{alg1}) and its improved version Algorithm 2 (S_{improve}), and compared their respective sizes (see Table 1 and Figure 2).

Next, we focus on instances where computing the optimal solution to problem 1 becomes difficult, even with encodings based on a SAT solver (Arenas et al., 2022) or PARTIAL MAXSAT (Audemard et al., 2022). These instances, characterized by their complex structure or large size, are particularly challenging to solve. To address this, we trained a decision tree on high-dimensional benchmarks b (with high n) and significant depth (**Depth**). We then counted the number of instances among the m selected in the test set for which the exact methods do not find a solution within a fixed limited time of 1, 3, and 5 minutes, respectively. These results were compared to the average computation times required by algorithm 1 (see Table 2).

Sentiment140. *Sentiment140* (see (Go, 2009) for more information) is a dataset containing 1.6 million textual documents, aimed at determining the sentiment expressed in each text: positive, negative, or neutral. In our process, we selected a portion of this dataset by limiting ourselves to texts that reflect only negative or positive emotions, in order to transform the problem into a binary classification.

To convert from text format to a numerical dataset, we followed the following steps:

1. **Text Preprocessing**: We normalized the texts by removing punctuation, uppercase letters, etc. This preprocessing reduces complexity and improves the quality of text analysis.
2. **Dictionary Creation**: We then generated a dictio-



(a) `employee` (b) `compas`

(c) `startup` (d) `christine`

Figure 2: Box plots for (*Employee*, *Startup*) on the left and (*Compas*, *Christine*) on the right, representing the sizes of direct reasons, sufficient reasons, minimum-size sufficient reasons, approximate solutions returned by algorithm 1, as well as their improvement found by algorithm 2.

nary of 47877 entries, corresponding to the characters and words potentially present in the texts.

3. **Projection of Texts into the Dictionary**: Each text was projected into this dictionary by encoding the words using *one-hot* encoding (Samuels, 2024), thereby transforming the texts into binary numerical vectors.

Remark 1 (**Tree Construction**). *For the learning of decision trees on the Sentiment140 dataset, a* CART *algorithm was used with an increased maximum depth to create a complex and deep decision tree.* ***Pruning was deliberately omitted*** [6] *in order to obtain a deeper tree, thus capturing a greater number of interactions between the features.*

Remark 2 (**SAT Encoding**). *We specify that in this work, the encoding* SAT *proposed by (Arenas et al., 2022) for computing a probabilistic explanation for a confidence threshold* $\delta \in (0, 1]$ *has been adapted to the case* $\delta = 1$ *to derive a minimum-size sufficient reason.*

5.2 Experimental Results

Table 1 shows a sample of our results for 21 datasets. For each dataset, we measured the average time re-

[6] This step was also applied during the learning process for both the *Adult* and *Gisette* datasets.

quired to obtain a minimum-size sufficient reason (denoted $|S^*|$), which does not exceed 10 seconds. The first column indicates the name of the dataset b, followed by the number of binary features ($\#F$) and the number of instances ($\#I$) in each dataset. Next, we display the accuracy of the tree T_b ($\%A$), which reflects its effectiveness in classifying the data, as well as the size of the tree ($|T|$) and its depth (**Depth**), providing an overview of the model's complexity.

The column |Explanation| presents the average size of the explanations computed for the m instances considered from the test set, with distinct notations for $|S^*|$, $|S_{\text{algo1}}|$, and $|S_{\text{improve}}|$. These notations correspond, respectively, to the average size of the approximate solutions provided by algorithm 1 and those after improvement by algorithm 2. We observe that the average size of the solutions generated by 1 is very close to that of the minimum-size sufficient reasons, indicating highly effective performance from the algorithm.

In general, for all the datasets studied, the average error $||S_{\text{algo1}}| - |S^*||$ is on the order of 10^{-2}, demonstrating the precision of the approximate solutions. This error slightly decreases with the improved solutions S_{improve}, reaching a precision of 10^{-3}. These results highlight the fact that algorithm 1 already provides excellent performance for approximating minimum-size sufficient reasons (approximate solution of problem 1).

The table 2 presents experimental results for three datasets that contain a large number of features, trained to generate decision trees of considerable size and maximum depth. The columns $\#I$, $\#F$, $\%A$, $|T|$, and **Depth** indicate respectively the number of instances, the number of binary features, the accuracy of the tree, the size of the tree, and its depth, thus providing an overview of the complexity of the trees.

The columns «1-min», «3-min», and «5-min» show the number of instances for which the solvers (PARTIAL MAXSAT and SAT) failed to provide solutions within the allotted times of 1, 3, and 5 minutes, respectively. This criterion allows for a comparison of the solvers' performance based on computation time. It is important to note that algorithm 1 manages to generate a solution in less than 30 seconds for all datasets and tested instances, highlighting its efficiency in these experiments.

Although the PARTIAL MAXSAT solver manages to return results for a large number of instances in under a minute, some so-called "hard" instances do not receive a response even after 5 minutes. For example, out of a total of 1000 instances for the *adult* dataset, only three instances remain unsolved after 5 minutes, compared to 557 for the SAT solver. In the case of the *sentiment140* dataset, the SAT solver encounters major difficulties, particularly due to the large size of the input and the complexity of the tree, which causes an explosion in the number of clauses in the SAT encoding. This phenomenon, as mentioned in (Bounia and Koriche, 2023; Arenas et al., 2022), typically occurs when the tree is too complex or when the dataset is of very high dimension. In these situations, the approximation provided by algorithm 1 becomes particularly valuable for circumventing the time limitations of traditional solvers. These results highlight the effectiveness of the algorithm in scenarios where exact approaches fail to produce a solution within reasonable time limits.

To illustrate the gain and loss of intelligibility of the approximations generated by algorithms 1 and 2, we created boxplots for the compas and employee datasets, showing the distribution of sizes of different abductive explanations: direct reasons in blue, sufficient reasons in green, outputs of algorithm 1 in yellow, outputs of algorithm 2 in orange, and the minimum-size sufficient reason in red. Figure 2 presents these boxplots. We observe a significant reduction in the number of features used in the direct and sufficient reasons thanks to the approximate solutions of 1, especially in its improved version, which constitutes a sufficient reason. This confirms that approximate minimum-size sufficient reasons can be concise enough to serve as an effective alternative in cases where calculating a minimum-size sufficient reason is out of reach.

6 Conclusion

In this article, we addressed the problem of approximating minimum-size abductive explanations, formulated as the selection of a minimal set of leaders achieving an error bound for a supermodular function (problem 1). Finding a minimum-size abductive explanation for an instance $\boldsymbol{x}$ given classifier h represents a considerable challenge, especially when h is modeled by a decision tree T. This challenge becomes even more complex for intricate instances where exact methods fail to provide efficient results, rendering the computation of minimum-size explanations impractical.

To address this difficulty, we proposed a greedy algorithm (algorithm 1) that includes an approximation bound. This algorithm allows for the efficient

Dataset	#I	#F	%A	$\|T\|$	Depth	1-min	3-min	5-min
sentiment140	$\approx 10^6$	47877	83.28	14537	93	$(41,-)$	$(35,-)$	$(33,-)$
adult	48842	2395	81.68	5014	54	$(11,712)$	$(7,623)$	$(3,557)$
gisette	13500	5000	93.14	436	32	$(3,11)$	$(0,3)$	$(0,0)$

Table 2: Table of results for hard instances of 3 datasets.

computation of abductive explanations whose size approaches the minimum-size while maintaining a certain level of computational efficiency. Our experiments demonstrated that this algorithm, when combined with algorithm 2, generates explanations that are not only close to the minimum-size but also more compact than those provided by an arbitrary sufficient reason or a direct reason for a instance $\boldsymbol{x}$ given a decision trees T.

By carefully balancing computation time and explanation size, these algorithms prove to be particularly effective for tackling difficult instances where traditional approaches encounter limitations. Indeed, the ability to produce compact and informative explanations constitutes a significant advancement in the field of machine learning model interpretability, which is essential for enhancing user trust in these systems.

For our future work, we plan to extend this approach to other types of classifiers, including random forests, boosted trees, and neural networks (including multilayer perceptrons, or MLP). This extension will be particularly relevant in cases where evaluating the unnormalized error function $\mu_{h,x}$ proves costly, which is often the case for more complex models. By integrating these new methods, we hope to further improve the efficiency and relevance of the generated explanations, thereby contributing to better interpretability of artificial intelligence systems.

REFERENCES

Adadi, A. and Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, 6:52138–52160.

Arenas, M., Barceló, P., Romero Orth, M., and Subercaseaux, B. (2022). On computing probabilistic explanations for decision trees. *Advances in Neural Information Processing Systems*, 35:28695–28707.

Audemard, G., Bellart, S., Bounia, L., Koriche, F., Lagniez, J., and Marquis, P. (2021). On the computational intelligibility of boolean classifiers. In *Proc. of KR'21*, pages 74–86.

Audemard, G., Bellart, S., Bounia, L., Koriche, F., Lagniez, J.-M., and Marquis, P. (2022). On the explanatory power of boolean decision trees. *Data Knowledge Engineering*, 142:102088.

Audemard, G., Bellart, S., Bounia, L., Lagniez, J.-M., Marquis, P., and Szczepanski, N. (2023). Pyxai : calculer des explications pour des modèles d'apprentissage supervisé. EGC.

Bounia, L. and Koriche, F. (2023). Approximating probabilistic explanations via supermodular minimization (corrected version). In *Uncertainty in Artificial Intelligence (UAI 2023).*, volume 216, pages 216–225.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 785–794, New York, NY, USA.

Clark, A., Alomair, B., Bushnell, L., and Poovendran, R. (2014a). Minimizing convergence error in multi-agent systems via leader selection: A supermodular optimization approach. *IEEE Transactions on Automatic Control*, 59(6):1480–1494.

Clark, A., Bushnell, L., and Poovendran, R. (2014b). A supermodular optimization framework for leader selection under link noise in linear multi-agent systems. *IEEE Transactions on Automatic Control*, 59(2):283–296.

Crama, Y. and Hammer, P. L. (2011). Boolean functions - theory, algorithms, and applications. In *Encyclopedia of mathematics and its applications*.

Darwiche, A. (1999). Compiling devices into decomposable negation normal form. pages 284–289.

Darwiche, A. and Hirth, A. (2020). On the reasons behind decisions. In *Proc. of ECAI'20*.

Frosst, N. and Hinton, G. E. (2017). Distilling a neural network into a soft decision tree. *ArXiv*, abs/1711.09784.

Go, A. (2009). Twitter sentiment classification using distant supervision.

Ignatiev, A., Narodytska, N., and Marques-Silva, J. (2019). Abduction-based explanations for machine learning models. In *Proc. of AAAI'19*, pages 1511–1519.

Izza, Y., Ignatiev, A., and Marques-Silva, J. (2020). On explaining decision trees. *ArXiv*, abs/2010.11034.

Izza, Y., Ignatiev, A., and Marques-Silva, J. (2022). On tackling explanation redundancy in decision trees. *J. Artif. Intell. Res.*, 75:261–321.

Izza, Y. and Marques-Silva, J. (2021). On explaining random forests with SAT. In *Proc. of IJCAI'21*, pages 2584–2591.

Lipton, Z. C. (2018). The mythos of model interpretability. *ACM*, 61(10):36–43.

Louenas, B. (2023). *Modéles formels pour l'IA explicable: des explications pour les arbres de décision*. PhD thesis, Université d'Artois.

Louenas, B. (2024). Enhancing the Intelligibility of Boolean Decision Trees with Concise and Reliable Probabilistic Explanations. In *20th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Lisbao, Portugal.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting m(ijcaiodel predictions. In *Proc. of NIPS'17*, pages 4765–4774.

Marques-Silva, J. (2023). Logic-based explainability in machine learning. *ArXiv*, abs/2211.00541.

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Molnar, C. (2019). *Interpretable Machine Learning - A Guide for Making Black Box Models Explainable*. Leanpub.

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

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should I trust you?": Explaining the predictions of any classifier. In *Proc. of SIGKDD'16*, pages 1135–1144.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proc. of AAAI'18*, pages 1527–1535.

Samuels, J. (2024). One-hot encoding and two-hot encoding: An introduction.

Shih, A., Choi, A., and Darwiche, A. (2018). A symbolic approach to explaining bayesian network classifiers. In *Proc. of IJCAI'18*, pages 5103–5111.

Wäldchen, S., Macdonald, J., Hauch, S., and Kutyniok, G. (2021). The computational complexity of understanding binary classifier decisions. *J. Artif. Intell. Res.*, 70:351–387.