Using Submodular Optimization to Approximate Minimum-Size Abductive Path Explanations for Tree-Based Models

Louenas Bounia[1]

[1]LIPN-UMR CNRS 7030., Université Sorbonne Paris Nord, Villetaneuse, France

Abstract

One of the key challenges of Explainable Artificial Intelligence (XAI) is providing concise and understandable explanations for classification model predictions. An abductive explanation for a given instance is a minimal set of features that justify the prediction. These minimal explanations are valuable for their interpretability, as they eliminate redundant or irrelevant information. However, computing these explanations is difficult, even for simpler classifiers like decision trees. Finding a minimum-size abductive explanation in decision trees is an NP-complete problem, and this complexity extends to random forests for minimum-size majoritary reasons. In this work, we focus on finding minimal sets of features along the paths leading to the decision, called *path-abductive explanations*. We show that the problem of finding *minimum-size path-abductive explanations* in decision trees and *minimum-size path-majoritary reasons* in random forests is also NP-complete. To address this, we reformulate the problem as a submodular optimization task and propose a greedy algorithm with optimality guarantees. Our experiments demonstrate that this algorithm produces near-optimal explanations efficiently and offers a strong alternative for difficult instances, where exact methods based on SAT encodings are computationally expensive. This approach is especially useful in resource-limited environments where modern SAT solvers are not feasible.

1 INTRODUCTION

The supervised classification problem involves deducing a model capable of predicting labels from annotated data. Common classifiers include decision trees [34], random forests [10], XGBOOST [11], support vector machines [15], and neural networks, which are widely used across fields like text and image classification, customer analysis, and medical diagnosis. However, with increasing use in critical sectors like healthcare and finance, the ability to explain model decisions is vital for transparency, trust, and regulatory compliance [31].

Formal explanations play a central role in Explainable Artificial Intelligence (XAI), as they provide mathematically validated justifications [29], which makes them particularly suitable for sensitive applications, such as the medical, financial, or legal domains. Unlike post-hoc agnostic methods such as LIME [35], SHAP [27], Anchors [36], or counterfactual explanations [18], which rely on local perturbations or game theory without considering the internal structure of the model, formal explanations are directly tied to the behavior of the studied model. This structural link grants them crucial properties of faithfulness, consistency, and robustness, often absent in agnostic methods. The latter can, indeed, generate identical explanations for opposite predictions [21], lack rigorous theoretical foundations [28], or be sensitive to input perturbations [1], undermining their reliability in critical contexts. Conversely, for a Boolean classifier h, a formal abductive explanation is characterized by the PI (Prime Implicant), which corresponds to a minimal subset of features I such that the restriction x_I of the input $\boldsymbol{x}$ is sufficient to guarantee the output $h(x)$ [21, 17]. Although finding such an explanation may be an NP-hard problem [14, 4], efficient solutions have been proposed for certain classes of models, notably decision trees and random forests [2, 3, 23]. Finally, we specify that our approximation proposed in this work is also distinguished by the use of formal approximations, which preserve the theoretical guarantees inherent to formal explanations, unlike empirical approximations used in agnostic approaches such as SHAP or LIME.

Conciseness is as important as validity in ensuring explanation comprehensibility. Human cognitive limits [30] justify the need for smaller explanations. However, finding minimum-size explanations is challenging, even for tree-

based models. Computing a minimum-size abductive explanation for decision trees is NP-hard [5, 6], and for random forests, computing a PI-explanation is DP-complete [23], with minimum-size abductive explanations being $\sum_2^p$-complete [2]. Majoritary reasons, introduced for random forests [2], are implicants of the majority of trees in the forest, but finding their minimum size is also NP-hard. Constraint optimization and modern SAT solvers have been applied to compute efficient explanations, such as the MUS method [23] for PI-explanations and the MaxSAT solver for minimum-size majoritary reasons [2].

The high computational complexity of these exact methods can become prohibitive, particularly for hard-to-explain instances or high-dimensional inputs, where computational time increases significantly. This issue is compounded in resource-constrained environments, where hardware and time limitations further restrict computation. To address this, we focus on approximating minimum-size explanations using submodularity, applied efficiently to decision trees and random forests. We aim to approximate minimum-size sufficient reasons (PI-explanations) for decision trees and minimum-size majoritary reasons (`minMAJ`) for random forests. For tree-based models, we focus on path-restricted explanations, which reflect the model's internal decision-making process. Our work proposes an efficient approximation method for minimum-size path-abductive explanations, which aligns with the internal workings of decision trees and ensures concise and relevant explanations.

Contributions and Main Motivation. In this work, we focus on approximating minimum-size explanations through the lens of *submodularity*, with an efficient application to *decision trees* and *random forests*. More specifically, we focus on *minimum-size path-abductive explanations*, which are based on decision paths (*path-explanations*) and are aligned with the internal workings of tree-based models. These explanations minimize the redundancy in path-explanations that are redundant [24, 25], while remaining consistent with the model's reasoning.

Our contributions include: First, we reformulate the problem of computing *minimum-size abductive explanations* as a submodular optimization task, applicable to a Boolean classifier h and a data instance $\boldsymbol{x}$. Second, we extend this reformulation to the computation of *minimum-size path-majoritary reasons* (denoted `PminMAJ`) for random forests F. We demonstrate that this problem remains NP-complete, even for a single tree ($F = \{T\}$), where majoritary reasons coincide with PI-explanations. Finally, we propose an efficient *greedy algorithm* with theoretical optimality guarantees, including an approximation bound on the size of the explanations.

Our main motivations are: We aim to provide an efficient alternative when computing minimum-size reasons becomes challenging, particularly for complex instances or high-dimensional inputs, where exact methods based on SAT solvers are computationally expensive. Additionally, we focus on delivering explanations in *resource-constrained environments*, where the use of SAT solvers or powerful machines is not feasible. Furthermore, we emphasize the importance of *path-explanations*, as the local decision-making process of tree-based models relies on these paths, offering concise justifications that align with the internal reasoning of the model.

In summary, this work proposes a practical and theoretically grounded solution for approximating minimum-size abductive explanations, addressing the limitations of existing methods in demanding contexts. *Path-explanations* play a central role in this approach, providing concise and interpretable justifications that are aligned with the internal reasoning of the model.

2 PRELIMINARIES

Classification problems. We assume that the reader is familiar with the basic concepts of machine learning, such as supervised learning, binary classification, random forests, and the principle of majority voting.

Notations. Let n be an integer, and let $[n]$ denote the set $\{1, \ldots, n\}$. We denote by $\mathcal{F}_n$ the class of all Boolean functions mapping $\{0,1\}^n$ to $\{0,1\}$, and $X_n = \{x_1, \ldots, x_n\}$ refers to the set of Boolean variables. An assignment $\boldsymbol{x} \in \{0,1\}^n$ is called an *instance*. A *literal* ℓ is either a variable x_i or its negation $\overline{x}_i$. 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 at least one model (i.e., an assignment that satisfies it). Given an instance $z \in \{0,1\}^n$, the corresponding term t_z is defined as follows: $t_z = \bigwedge_{i=1}^n x_i^{z_i} = \{x_1^{z_1}, \ldots, x_n^{z_n}\}$, where $x_i^0 = \overline{x}_i$ and $x_i^1 = x_i$.

An *implicant* of a Boolean function f is a term t such that t implies f (i.e., every assignment satisfying t also satisfies f). A *prime implicant* of f is an implicant t of f such that no proper 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 $S \subseteq [n]$ of features, the restriction of $\boldsymbol{x}$ to S, denoted x_S, is the partial instance in $\{0,1,*\}^n$ such that: $(x_S)_i = x_i$ if $i \in S$, and $*$ otherwise. Any instance $\boldsymbol{y} \in \{0,1\}^n$ is covered by x_S if and only if $y_S = x_S$. The term t_{x_S} associated with the partial instance x_S is defined as:

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

[1] In this work, we treat a term as a set of literals for simplicity.

2.1 DECISION TREE AND RANDOM FOREST.

A binary decision tree on X_n is a binary tree T, where each internal node is labeled with one of the n Boolean input variables from X_n, and each leaf is labeled with either 0 or 1. Each variable is assumed to appear at most once on any path from the root to a leaf (read-once property). The value $T(x) \in \{0,1\}$ of T for an input instance $\boldsymbol{x}$ is determined by the label of the leaf reached from the root node.

A ***random forest*** on X_n is a set $F = \{T_1, \cdots, T_m\}$, where each T_i ($i \in [m]$) is a decision tree on X_n, and the value $F(\boldsymbol{x})$ is given by

$$F(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \frac{1}{m}\sum_{i=1}^{m} T_i(\boldsymbol{x}) > \frac{1}{2} \\ 0 & \text{otherwise.} \end{cases}$$

The *size* of F is given by $|F| = \sum_{i=1}^{m} |T_i|$, where $|T_i|$ is the number of nodes present in T_i. The class of decision trees on X_n is denoted by $\mathtt{DT}_n$, and the class of random forests with at most m decision trees (for $m \geq 1$) on $\mathtt{DT}_n$ is denoted by $\mathtt{RF}_{n,m}$. Finally, $\mathtt{RF}_n = \bigcup_{m\geq 1} \mathtt{RF}_{n,m}$ and $\mathtt{RF} = \bigcup_{n\geq 1} \mathtt{RF}_n$. It is well known that any decision tree $T \in \mathtt{DT}_n$ can be transformed in linear time into an equivalent DNF (or an equivalent CNF). This DNF is an orthogonal DNF (see [9] for more detail). However, when moving to random forests, the situation is quite different. Any formula in CNF or DNF can be converted in linear time into an equivalent random forest, but there is no polynomial space conversion from a random forest to CNF or DNF [2].

2.2 ABDUCTIVE EXPLANATIONS

Abductive Explanations and Decision Trees. An abductive explanation [2] for an instance $\boldsymbol{x}$ is a subset of features S such that the restriction of $\boldsymbol{x}$ to S is sufficient to obtain the same prediction. A *sufficient reason* (denoted as PI-explanation) is a minimal abductive explanation with respect to inclusion, while a *minimum-size sufficient reason* is an abductive explanation containing the smallest number of literals.

Decision trees are naturally interpretable, as each instance $\boldsymbol{x}$ can be explained by a unique direct path from the root to a decision leaf, called a *direct reason* (or *path-explanation*), denoted $P^h_{\boldsymbol{x}}$. However, these direct reasons may contain redundant features [24], justifying the use of more concise explanations, such as *sufficient reasons* and *minimum-size sufficient reasons*. Although sufficient reasons can be computed in polynomial time for decision trees, finding a minimum-size sufficient reason is an NP-hard problem [6].

Definition 1 (Path-Sufficient Reason ($\mathtt{Path-PI}$))**.** *Let h be a classifier represented by a tree $T \in \mathtt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$ an instance. A $\mathtt{Path-PI}$ for $\boldsymbol{x}$ given h is a set of features S such that the associated term t_{x_S} is a sufficient reason for $\boldsymbol{x}$ given h and $t_{x_S} \subseteq P^h_{\boldsymbol{x}}$. A **Minimum-Size Path-Sufficient Reason** (PminPI) is a $\mathtt{Path-PI}$ of minimum-size.*

[2]Unlike [21], we do not require abductive explanations to be minimal with respect to inclusion.

It is evident that it is always possible to derive a sufficient reason t (PI-explanation) from path explanation $P^h_{\boldsymbol{x}}$. However, a PminPI generally does not coincide with a minPI.

Remark 1. *When h is a classifier represented by a decision tree $T \in \mathtt{DT}_n$, it is important to note that the PminPI reason generally does not coincide with minPI, although all PminPI and $\mathtt{Path-PI}$ reasons are PI-explanations.*

Furthermore, for an instance $\boldsymbol{x}$ and a classifier h represented by a tree T, the number of $\mathtt{Path-PI}$ can be exponential in the size of the input.

Proposition 1. *There exists a decision tree $T \in \mathtt{DT}_n$ of depth $\log_2(n+1)$ such that, for any instance $\boldsymbol{x} \in \{0,1\}^n$, the number of $\mathtt{Path-PI}$ for $\boldsymbol{x}$ given T is at least $\left\lfloor \frac{3}{2}^{\frac{n+1}{2}} \right\rfloor$.*

Due to the large number of $\mathtt{Path-PI}$, finding a PminPI is not always straightforward.

Proposition 2. *Let h be a classifier represented by a decision tree $T \in \mathtt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$ an instance. Computing a **Minimum-size path-sufficient reason** (PminPI) for $\boldsymbol{x}$ given h is an NP-hard problem.*

Despite this result, it is possible to compute a PminPI in many practical cases. To achieve this, we use a slightly modified version of the encoding proposed in [5], which relies on PARTIAL MAXSAT solvers. However, these encodings require significant memory resources and powerful machines, especially for handling high-dimensional or challenging instances.

Abductive Explanations and Random Forests. The notions of *PI-explanation* (or sufficient reason) and minPI are general and applicable to any classifier h, including when h is a random forest $F \in \mathtt{RF}_{n,m}$. However, as mentioned in Section 1, computing minPI and *PI-explanations* for a random forest remains opaque. In this work, we focus on a type of explanation better suited to the internal workings of random forests, introduced by [2]: *direct reasons* and, more specifically, *majoritary reasons* (MAJ). A *direct reason* for a random forest is defined as the conjunction of the reasons for the features located on the paths of the trees that vote for the majority class.

Definition 2. *Let $F = \{T_1, \ldots, T_m\}$ be a random forest (random forest) in $\mathtt{RF}_{n,m}$, and $\boldsymbol{x} \in \{0,1\}^n$ be an instance. The **direct reason** for $\boldsymbol{x}$ given by F is the term P^F_x defined by $P^F_x = \bigwedge_{T_i \in F^x} P^{T_i}_x$ where $F^x = \{T_i \in F \mid T_i(\boldsymbol{x}) = F(\boldsymbol{x})\}$. By construction, P^F_x can be computed in time $O(n \cdot |F|)$.*

However, as with decision trees, P_x^F often contains redundant features [2]. We therefore focus on a stronger version of abductive explicitons than P_x^F: *majoritary reasons*. A majoritary reason, as defined in [2], is an implicant t of the majority of trees in F, where the removal of a single feature invalidates the majority condition.

Definition 3. *Let $F = \{T_1, \ldots, T_m\}$ be a random forest in $\mathtt{RF}_{n,m}$ and $\boldsymbol{x} \in \{0,1\}^n$ an instance. A* majoritary reason (MAJ) *for $\boldsymbol{x}$ given by F is a term t covering $\boldsymbol{x}$ such that t is an implicant of at least $\lfloor \frac{m}{2} \rfloor + 1$ decision trees T_i, and for every literal $l \in t$, the term $t \setminus l$ does not satisfy this condition.*

A Path-majoritary reason (PMAJ) *for $\boldsymbol{x}$ given by F is a* MAJ *such that $t \subseteq P_x^F$. A* minimum-size majoritary reason (minMAJ) *is a* MAJ *with the smallest number of literals, and a* minimum-size Path-majoritary reason (PminMAJ) *is a* PMAJ *with the smallest number of literals.*

In analogy with the proposition 1, and considering that a decision tree is a special case of a random forest where $F = \{T\}$, it is obvious that the number of reasons PMAJ can also be exponential. Moreover, in this special case, the reasons minMAJ coincide with the minPI. Note that all reasons PMAJ and PminMAJ are MAJ reasons, but they are not necessarily minMAJ. While deriving a reason MAJ or PMAJ is feasible in linear time, finding their minimum-size versions (minMAJ and PminMAJ) is computationally more complex. Finding a reason minMAJ has been shown to be an NP-complete problem [2], and the following proposition shows that computing PminMAJ is also hard.

Proposition 3. *Let $F \in \mathtt{RF}_{n,m}$, $\boldsymbol{x} \in \{0,1\}^n$, and $k \in \mathbb{N}$. Deciding whether there exists a reason* PminMAJ *t for $\boldsymbol{x}$ given F, such that t contains at most k features, is an* NP-*complete problem.*

Proposition 3 illustrates the difficulty of deriving explanations in a random forest, particularly when the size of F or the dimension of $\boldsymbol{x}$ is large. To overcome this complexity, we explore efficient approximations with few resources. Finally, we recall abductive explanations of minimum size in the context of the error function.

Error function. We now define a central concept for the rest of this work, the *explanation error function* $\epsilon_{h,\boldsymbol{x}}(S)$ for a classifier h and an instance $\boldsymbol{x}$, which can be interpreted as the probability of making an *explanation error* using a subset S of features. Given a classifier h and an instance $\boldsymbol{x}$ for which the prediction $h(\boldsymbol{x})$ must be explained, let $\epsilon_{h,\boldsymbol{x}} : 2^{[n]} \to \mathbb{R}$ be the *explanation error function* [8, 7] defined by: $\epsilon_{h,\boldsymbol{x}}(S) = \frac{\mu_{h,\boldsymbol{x}}(S)}{2^{n-|S|}}$ where $\mu(S) = |\{\boldsymbol{y} \in \{0,1\}^n : h(\boldsymbol{y}) \neq h(\boldsymbol{x}), \boldsymbol{y}_S = \boldsymbol{x}_S\}|$. As noted earlier, $\epsilon_{h,\boldsymbol{x}}(S)$ can be interpreted as the probability of making an *explanation error*, where $\mu_{h,\boldsymbol{x}}(S)$ represents the number of errors induced by the choice of S. For a feature subset S, t_{x_S} is an abductive explanation for $\boldsymbol{x}$, given h, if $\epsilon_{h,\boldsymbol{x}}(S) = 0$. Moreover, t_{x_S} is a PI-explanation if $\epsilon_{h,\boldsymbol{x}}(S) = 0$ and $\epsilon_{h,\boldsymbol{x}}(S') > 0$ for any proper subset S' of S. t_{x_S} is minPI if it is a PI-explanation that contains a minimal number of features. Note that when h is represented by a decision tree T, the function $\mu_{h,\boldsymbol{x}}$ can be simply rewritten as follows:

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

The result shows that the evaluation of $\epsilon_{h,\boldsymbol{x}}(S)$ can be achieved in time $O(|S| \cdot |T|)$ when h is represented by a decision tree T [8, 26], which is not always the case in general. Indeed, in the general case, the problem of evaluating $\epsilon_{h,\boldsymbol{x}}(S)$ is **#P-hard** (or #-SAT) [16]. We now formulate the problem of finding a minPI reason for $\boldsymbol{x}$ given h.

3 PROBLEM FORMULATION

Main idea. A term t_{x_S}, associated with a feature subset $S \subseteq V = [n]$, constitutes an abductive explanation for $\boldsymbol{x}$ given h if and only if $\epsilon_{h,\boldsymbol{x}}(S) = 0$, i.e. $\mu_{h,\boldsymbol{x}}(S) = 0$. Thus, a minimum-size abductive explanation corresponds to the smallest set S (in cardinality) satisfying $\mu_{h,\boldsymbol{x}}(S) = 0$.

3.1 APPROXIMATION OF A MINIMUM-SIZE ABDUCTIVE EXPLANATION

The problem of finding a minimum-size abductive explanation (or a minPI) for an instance $\boldsymbol{x}$, given a classifier h, can be formulated as an optimization problem. The objective is to select a subset S of minimum-size features satisfying an upper bound constraint $\alpha \geq 0$ on a function $g_{h,\boldsymbol{x}}(S)$, which depends on the classifier h and the instance $\boldsymbol{x}$.

Problem 1. *Let h be a classifier, an instance $\boldsymbol{x}$ and a constant bound $\alpha \geq 0$. Let $g_{h,x}$ be a non-negative set function depending on h and $\boldsymbol{x}$. The problem studied consists in finding a subset of features $S \subseteq V$ solution of the problem:*

$$\begin{aligned} \min_{S \subseteq V} \quad & |S| \\ \textit{s.t.} \quad & g_{h,\boldsymbol{x}}(S) \geq \alpha \end{aligned}$$

Proposition 4. *Let $h : \{0,1\}^n \to \{0,1\}$ be a classifier, $\boldsymbol{x} \in \{0,1\}^n$ be an instance, and $S^* \subseteq V$ be a subset of features. For $V = [n]$, $g_{h,x}(S) = \mu_{h,x}(\emptyset) - \mu_{h,x}(S)$ and $\alpha = \mu_{h,x}(\emptyset)$, S^* is an optimal solution to the problem 1 if and only if $t_{x_{S^*}}$ is a* minPI *reason for $\boldsymbol{x}$ given at h.*

Application to decision trees. When a classifier h is represented by a decision tree, we restrict ourselves to the features present in the explanation based on the path P_x^h (denoted

V_x^{path}[3]) to generate a minimum-size explanation PminPI. To do this, it suffices to define $V = V_x^{\text{path}}$.

Proposition 5. *Let h be a classifier represented by a decision tree $T \in \texttt{DT}_n$ and an instance $\boldsymbol{x}$ to be explained. For $V = V_x^{path}$, $g_{h,x}(S) = \mu_{h,x}(\emptyset) - \mu_{h,x}(S)$ and $\alpha = \mu_{h,x}(\emptyset)$, S^* is an optimal solution to the problem 1 if and only if $t_{x_{S^*}}$ is a* PminPI *reason for $\boldsymbol{x}$ given h.*

In the case of decision trees, the evaluation of the error function $\mu_{h,x}(.)$ can be done in polynomial time (see [24]), and more precisely in linear time [9]. This evaluation is also feasible in polynomial time for linear classifiers [6], d-DNNF classifiers [20], and [19] decision diagrams. However, optimization problems, such as 1, are generally NP-hard in these cases (see [32], Chapter III).

This is consistent with the 4 and 5 propositions, as well as with the NP-hardness of computing the minimum explanations minPI and PminPI when h is a decision tree. In general, the evaluation of $\mu_{h,x}$ is #P-hard [16], making its computation intractable in polynomial time. For a random forest, this problem is equivalent to a #SAT, and the approximation or exact computation of minPI (of DP-complete complexity) remains out of reach. We therefore focus on another type of abductive explanation, majoritary reasons, and we will introduce a new error function adapted to random forests, efficiently computable in linear time.

3.2 APPROXIMATION OF MINIMUM-SIZE MAJORITARY REASONS

To circumvent the problem of evaluating the error function $\epsilon_{h,x}$ when h is represented by a random forest $F \in \texttt{RF}_{n,m}$, we focus on a subset of trees in F, namely those that vote for the majoritary class. This means that we consider the whole: $F^x = \{T_i \in F, i \in [m] \mid T_i(\boldsymbol{x}) = F(\boldsymbol{x})\}$. Inspired by the fact that a majoritary reason is an implicant of at least half of the trees in the forest and that the number of errors must be zero when considering an implicant of the majority of the trees (i.e. a MAJ-reason), we define the function: $\epsilon_{F^x}(S) = \frac{\mu_{F^x}(S)}{\#F^x \cdot 2^{n-|S|}}$ with:

$$\mu_{F^x}(S) = \left[\sum_{T_i \in F^x} \mu_{x,T_i}(S)\right] \mathbb{I}\left(\sum_{T_i \in F^x} \mathbb{I}_{\{\mu_{x,T_i}(S)=0\}} \leq \frac{m}{2}\right)$$

where $\#F^x$ denotes the number of trees in F^x, and $\mathbb{I}$ is the indicator function. The function ϵ_{F^x} can be interpreted as the average probability of making an explanation error by selecting a subset S of features. Note that when $F = \{T\}$, μ_{F^x} coincides with $\mu_{T,x}$.

[3]In the rest of the article, V_x^{path} denotes the set of features appearing in the direct paths leading to the decision for $\boldsymbol{x}$, whose classification must be explained.

Evaluation of μ_{F^x}. For any $T_i \in F$, the function $\mu_{x,T_i}(S)$ can be computed in $O(|S| \cdot |T_i|)$ [8, 26]. The evaluation of $\mu_{F^x}(S)$ then consists in computing $\mu_{x,T_i}(S)$ for each $T_i \in F^x$. Since $|F^x|$ represents the sum of the sizes of the trees that compose it, the total cost is $O(|S| \cdot |F^x|)$, with $|F^x| = \sum_{T_i \in F^x} |T_i|$.

Approximation of a mimumm-size majoritary reason. When a classifier h is a random forest and we seek to explain the prediction of a given instance $\boldsymbol{x}$, the problem 1 can be reformulated so that its optimal solution corresponds to a reason minMAJ for $\boldsymbol{x}$ given h.

Formally, this problem can be adapted as follows:

- Let a classifier h be represented by a random forest $F \in \texttt{RF}_{n,m}$.
- Let an instance $\boldsymbol{x}$ whose prediction is to be explained.
- Let $g_{h,x}(S) = \mu_{F^x}(\emptyset) - \mu_{F^x}(S), \alpha = \mu_{F^x}(\emptyset)$.

With these parameter adjustments in the formulation of the problem 1, the latter becomes a version adapted to the context of majority voting when the classifier h is a random forest F.

Proposition 6. *Let h be a classifier represented by a random forest $F \in \texttt{RF}_{n,m}$ and $\boldsymbol{x} \in \{0,1\}^n$ be an instance. If we set: $V = [n]$, $\alpha = \mu_{F^x}(\emptyset)$ and $g_{h,x}(S) = \mu_{F^x}(\emptyset) - \mu_{F^x}(S)$, then a set $S^* \subseteq V$ is an optimal solution to the problem 1 if and only if $t_{x_{S^*}}$ is a* minMAJ *reason for $\boldsymbol{x}$ given h. And if $V = V_x^{path}$ then S^* is an optimal solution to the problem 1 if and only if $t_{x_{S^*}}$ is a* PminMAJ *reason for $\boldsymbol{x}$ given h.*

4 APPROXIMATION ALGORITHMS

The main objective of this study is to relax the optimality constraint for the *Problem 1* (as well as its version adapted to the random forest context) by prioritizing obtaining a *sufficiently good* solution in terms of quality. To do so, we exploit submodular optimization techniques and a greedy algorithm, which have the advantage of being less resource-intensive than exact approaches based on SAT encodings or constraint optimization. This section begins with a reminder of the fundamental concepts of submodularity, followed by an analysis of the essential properties of the error functions $\mu_{h,x}$ and μ_{F^x}. Finally, we propose a greedy algorithm to obtain an approximate solution, accompanied by a study of its theoretical guarantees and experimental performances.

4.1 SUPERMODULAR FUNCTIONS

Let $f : 2^{[n]} \to \mathbb{R}$ be a function with real parts. We say that f is non-decreasing if $f(S \cup \{i\}) \geq f(S)$ for all $S \subseteq [n]$ and $i \in [n] \setminus S$, and non-increasing if $f(S \cup \{i\}) \leq f(S)$ 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 $[n]$: $f(A \cup B) + f(A \cap B) \geq f(A) + f(B)$. On the other hand, 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 all $S \subseteq [n]$ and $i \in S$, a function f is supermodular if and only if $-f$ is submodular. Moreover, f is modular if it is both submodular and supermodular.

In general, the error function $\epsilon_{h,x}(\cdot)$ is neither supermodular nor submodular, and it is neither non-increasing nor non-decreasing [8]. However, by considering the non-normalized version $\mu_{h,x}(\cdot)$, useful properties can be derived. For any classifier h and an instance $x \in \{0,1\}^n$, the function $\mu_{h,x}$ is non-negative, supermodular, and non-increasing [8]. Consequently, since μ_{F^x} is a linear combination of supermodular, non-negative, and non-increasing functions, it inherits these properties: μ_{F^x} remains supermodular, non-negative, and non-increasing.

Proposition 7. *Let $h : \{0,1\}^n \to \{0,1\}$ be a classifier, and let $\boldsymbol{x} \in \{0,1\}^n$ be an instance. Then, the function $g_{h,x}(S) = \mu_{h,x}(\emptyset) - \mu_{h,x}(S)$ is submodular, non-decreasing, and non-negative, with $g_{h,x}(\emptyset) = 0$.*

When h is represented by a random forest $F \in \mathtt{RF}_{n,m}$ and $g_{h,x}(S) = \mu_{F^x}(\emptyset) - \mu_{F^x}(S)$, Proposition 7 shows that $g_{h,x}$ is a non-negative, submodular, non-decreasing function satisfying $g_{h,x}(\emptyset) = 0$. Therefore, Problem 1 can be reformulated as a submodular optimization problem. Although such problems are often NP-hard (cf. Chapter III of [32]), the use of a greedy algorithm provides an approximate solution with performance guarantees close to optimal.

4.2 GREEDY ALGORITHM

A natural approach to minimize a supermodular and non-increasing function f under the strong constraint of minimizing $|S|$ consists of formalizing the problem as a leader selection problem for f, of minimum-size, reaching an error bound α. Greedy algorithms can then be used to compute an approximate solution. As shown in [13, 12], this greedy method benefits from mathematical guarantees on solution quality. In our study, the function $\mu_{h,x}(\cdot)$ is supermodular and non-increasing. However, the approach in [13] is also based on the work of [32] and on the fact that $g_{h,x}(\emptyset) = 0$, but [13] does not emphasize the details, being rather generic. Therefore, using supermodular minimization algorithms is not possible, as most rely on the assumption $\mu_{h,x}(\emptyset) = 0$, which in our case ($\mu_{h,x}(\emptyset) \neq 0$). The same applies to μ_{F^x}.

To circumvent this limitation, we are interested in a modified version of $\mu_{h,x}$, defined by the function $g_{h,x}(.)$. This function is submodular, non-negative, and non-decreasing, and it satisfies $g(\emptyset) = 0$. These properties allow the use of the greedy algorithm 1, described as follows:

Algorithm 1: Greedy Approximation Algorithm

Input: A submodular funciton g, termination bound α

Output: A set of feateurs S

$S \leftarrow \emptyset$

error $\leftarrow 0$

V {A set of features, by default $V \leftarrow [n]$}

while error $< \alpha$ **do**

  $e^* \leftarrow \underset{e \in V \setminus S}{\operatorname{argmax}}\, g(S \cup \{e\}) - g(S)$

  if $g(S \cup \{e\}) - g(S) \leq 0$ **then**

    return S

  else

    $S \leftarrow S \cup \{e^*\}$

    error $\leftarrow g(S)$

  end if

end while

return S

Theorem 1. *Let $h : \{0,1\}^n \to \{0,1\}$ be a classifier and $\boldsymbol{x} \in \{0,1\}^n$ an instance. For $g(S) = \mu_{h,x}(\emptyset) - \mu_{h,x}(S)$, let $|S^*| = k^*$ be the size of the 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 $\mu_{\max} = \max_{i \in V} \mu_{h,x}(\{i\})$

Theorem 1 establishes an approximation guarantee for the greedy algorithm, by comparing the size of the set S to that of the optimal set S^*. This bound remains valid for the adaptive version of the problem 1 applied to random forests, i.e. when h is represented by a random forest F. Moreover, by constructing S_{k-1} via the algorithm 1, we have $\mu_{h,x}(S_{k-1}) \neq 0$ ($\mu_{h,x}$ is non-increasing), which guarantees the validity of the approximation bounds. In the worst case, the algorithm 1 runs in time $O(n^3 \cdot |h|)$ (i.e. $O(n^3 \cdot |T|)$ for decision trees and $O(n^3 \cdot |F|)$ for random forests).

Discussion on the approximation bound. The approximation bounds of theorem 1 depend on the value of $\mu_{h,x}(S_{k-1})$, which varies depending on the instance considered. However, we believe that it is possible to obtain fixed and more precise bounds by using other greedy algorithms. Our experimental results support this intuition: by comparing the numerical values of the bounds described in theorem 1 with $\log(n)$ (see Table 1), we observe that the average value of the bound is typically a multiplicative factor $\gamma \cdot \log(n)$, with γ being non-negative constant. This suggests that the bounds could be reduced to a form proportional to $\log(n)$, without dependence on $\mu_{h,x}(S_{k-1})$. However, this

remains a conjecture, as no theoretical result yet rigorously supports the bound being in $\mathcal{O}(\log(n))$, although we have strong reasons to believe so. We think that it is possible to obtain a fixed and more precise bounds inspired in particular by the work of [22, 37], by reformulating the problem 1 as a bicriteria submodular optimization problem with submodular cover and submodular knapsack constraints.

Dataset	$\log(n)$	**Bound**
compas	3.78	5.84
titanic	4.43	5.86
yeast	3.30	9.19
malware	3.43	4.86
gisette	4.88	10.31
tae	3.40	4.26
spambase	5.37	10.70
mnist38	5.32	12.16
letter	4.43	6.62
meta	3.61	4.42

Table 1: Additional experimental results on several datasets when h is represented by a decision tree T. $\log(n)$ denotes the logarithm of the number of binary features n, and *Bound* is the average value of the approximation bound from Theorem 1, computed over at most $m = 250$ instances.

Improvement of the output from algorithm 1. The output of algorithm 1 does not necessarily guarantee a minimal abductive explanation for inclusion (sufficient or majoritary reason) for $\boldsymbol{x}$ given h. To refine this result, we extract a less redundant explanation from the solution S returned by algorithm 1, using a simple greedy procedure. Algorithm 2[4] takes as input the set S and iteratively removes elements that do not contribute to a minimal abductive explanation for the inclusion. Specifically, it discards any element ℓ such that $\mu_{h,\boldsymbol{x}}(S \setminus \{\ell\})$ still yields a valid explanation. The final output is a minimal abductive explanation for the inclusion for $\boldsymbol{x}$ given h.

Algorithm 2: Improving Solution Parsimony

Input: a classifier h, instance $\boldsymbol{x} \in \{0,1\}^n$, a set S

Output: a minimal abductive explanation for inclusion

$I \leftarrow S$ {S is the output of the algorithm 1}

for $\ell \in I$ **do**

  if $\mu_{h,x}(S) = 0$ **then**

    $S \leftarrow S - \{\ell\}$

  end if

end for

return S

[4]Recall that algorithm 2 runs in time $O(|S| \cdot |h|)$.

The algorithm 2 iteratively traverses the elements of S, eliminating those that do not affect the validity of the explanation. The output of the algorithm 2 constitutes a minimal explanation for inclusion.

5 EXPERIMENTS

In this section, we evaluate the performance of our approach by comparing the solutions returned by our greedy algorithm with exact methods based on Partial MaxSAT solver. Our goal is to measure the efficiency of our algorithm in approximating the optimal solution to Problem 1. We demonstrate that our method is a high-performing alternative, particularly for challenging instances where exact approaches become inefficient due to their high computational cost. Finally, we discuss the relevance of our approach in resource-constrained environments and explain why it represents a more suitable solution than existing exact methods.

5.1 EXPERIMENTAL PROTOCOL

We conducted experiments on various instances of Problem 1. Since, when $F = \{T\}$, the concepts `minPI`, `PminPI`, `Path` $-$ `PI`, and P_x^T coincide with `minMAJ`, `PminMAJ`, `PMAJ`, and P_x^F, respectively, we focus on the case where h is represented by a random forest $F \in \texttt{RF}_{n,m}$. The experiments were performed using Python code executed on a machine equipped with an Intel(R) Core $i9-9900$ processor clocked at 3.1 GHz and 64 GiB of RAM.

George Nemhause We studied a set of $B = 52$ datasets from well-known sources such as Kaggle (`www.kaggle.com`), OpenML (`www.openml.org`), and UCI (`archive.ics.uci.edu/ml/`). Categorical features were encoded as integers, while numerical features were binarized during the training of random forests. All datasets used are related to binary classification tasks.

Methodology. For each instance $\boldsymbol{x}$ in the test set of a dataset b, an explanation task is defined by the pair $(F_b, \boldsymbol{x})$, where F_b denotes the random forest representing a classifier h, trained on the training set of b using the *Scikit-Learn* library [33]. The training of F_b was performed with default hyperparameters, except for the *nb_estimator* parameter, which controls the number of trees in the forest. This parameter was adjusted to ensure high performance while avoiding an explosion in the size of the forest and the encodings, while maintaining good accuracy. A time limit of 60 minutes per instance was defined.

To evaluate the performance of the algorithm 1 in approximating a solution to Problem 1, we randomly select $m = \min(q, 250)$ instances $\boldsymbol{x}$ from the test set of b, where q is the size of this set. And due to space constraints, we limit ourselves to comparing the average sizes of the approximate solutions to those of the explanations (`PminMAJ`).

Comparison with exact methods. To compare the performance of the algorithm 1 with an exact solution, we used an approach based on a PARTIAL MAXSAT solver described in [2]. Concretely, given a random forest F associated with a classifier h and an instance $\boldsymbol{x}$, the *hard* clauses (C_{hard}) of the encoding represent the clause CNF of the forest, while the *soft* clauses encode the literals of the instance $\boldsymbol{x}$. The optimal solution of this instance PARTIAL MAXSAT corresponds to a minMAJ reason for $\boldsymbol{x}$ given h [2] (i.e. the optimal solution of Problem 1 with $V = [n]$ and $\alpha = \mu_{F^x}(\emptyset)$).

In the case of minimum-size majoritary reasons restricted to paths (PminMAJ), the above-mentioned encoding has been extended by adding the clause: $\bigvee\{x_i : (x_I)_i = 1\} \vee \bigvee\{x_i : (x_I)_i = 0\}$ where $I = V_x^{\text{path}}$ to the *hard* clauses (C_{hard}). Thus, the optimal solution of the extended PARTIAL MAXSAT problem corresponds to a PminMAJ for $\boldsymbol{x}$ given h (optimal solution of Problem 1 with $V = V_x^{\text{path}}$, $g(S) = \mu_{F^x}(\emptyset) - \mu_{F^x}(S),\ \alpha = \mu_{F^x}(\emptyset)$).

Analysis of Hard Instances. We focus here on instances for which the optimal resolution of Problem 1 becomes difficult for the PARTIAL MAXSAT solver. These situations arise when the size of the random forest F is large, or when the input instance's dimension is large, leading to an explosion in the size of the Boolean circuit encoding (CNF formula). However, even in datasets with lower dimensionality, complex instances may arise. This difficulty can also stem from the random and complex structure of the considered instances, making convergence to an optimal solution more difficult for the PARTIAL MAXSAT solver, thus posing computational challenges. Due to space constraints, our analysis is limited to the two datasets *Placement* and *Cars*.

5.2 EXPERIMENTAL RESULTS

Table 2 presents a sample of our results for 15 datasets. The columns include the dataset name, the number of binary features ($\#F$), the number of instances ($\#I$), the accuracy of the forest F_b ($\%A$), and the size of the forest ($|F|$). The column |Path-Reason| indicates the average size of the explanations computed for the m selected instances. The columns S^*, S_{algo1}, and S_{improve} respectively present the average sizes of the PminMAJ reasons, the output of Algorithm 1, and its improvement obtained with Algorithm 2. Similarly, the column |Times-Reason| shows the average time required to derive PminMAJ (sub-column S^*) and its approximation (sub-column S_{algo1}).

For the column |Path-Reason|, we observe that the average size of the approximate solutions to Problem 1 (with $V = [n]$) generated by Algorithm 1 is close to that of the optimal solutions (PminMAJ). For most of the studied datasets, the maximum average error $||S_{\text{algo1}}| - |S^*||$ is on the order of 2 on average, demonstrating the accuracy of the approximate solutions. This error slightly decreases with the improved solutions S_{improve}, reaching a precision close to 1.2. This error can be as low as 10^{-3} for some datasets, such as *monk*, *vote*, *tic-tac-toe*, *compas*, and *heart*. Moreover, the average size of the outputs of Algorithm 1 is significantly smaller than that of MAJ and PMAJ (see the supplementary material in https://github.com/Lounesbo). The column |Times-Reason| shows the average time required to find PminMAJ and its approximation. For the datasets in the table, the computation time is almost identical. However, this is not always the case, as shown by our study on *Placement* and *Cars*. The results in Table 2 demonstrate that the algorithm can compute an efficient approximation solution.

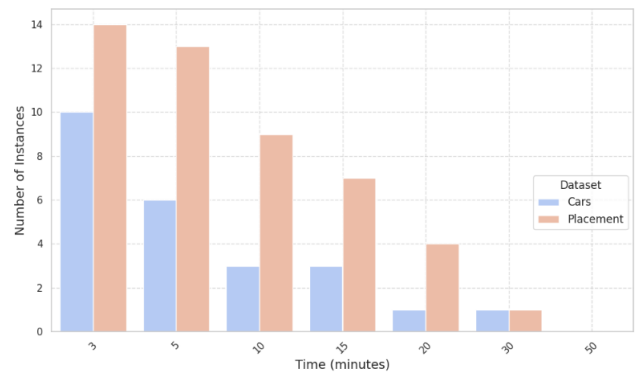

Figure 1: Placement and Cars

Table 3 presents experimental results for 2 datasets. The columns $\#I$, $\#F$, $\%A$, and $|F|$ respectively indicate the number of instances, the number of binary features, the accuracy of the forest, and its size. The column Path-Times shows the average computation times for the $m = 65$ selected instances. The sub-columns S^* and S_{alg1} reflect the average computation times for the exact and approximate solutions. We note that the average computation time of the exact solvers often exceeds 10 minutes (as already shown in the experimental part of the work by [2]), while Algorithm 1 generates a solution in less than 5 seconds.

Figure 1 illustrates the number of instances for which the PARTIAL MAXSAT solvers failed to find a PminMAJ reason within the allotted time limits of 3, 5, 10, 15, 20, 30, and 50 minutes (Placement in orange, Cars in blue). These results show that exact methods can take a significant amount of computation time, while the greedy algorithm requires negligible time. Although these results highlight the slowness of solvers in producing results, Figure 1 shows that some instances deemed *difficult* remain unresolved even after 50 minutes. For example, for the *Placement* dataset, out of 65 instances in the test set, 14 remain unsolved after 3 minutes, 12 after 5 minutes, 9 after 10 minutes, and 4 after 20 minutes, with 1 remaining after 30 minutes. Similar results are observed for the *Cars* dataset. This phenomenon typically occurs when the forest consists of complex, large, and deep trees, and/or when the dataset is high-dimensional. These results highlight the utility of our greedy algorithms.

dataset			random forest			|Path-Reason|				Times-Reason	
name	#F	#I	%A	$\#T$	$\|F\|$	$\|P_x^F\|$	S^*	S_{algo1}	S_{improve}	S^*	S_{algo1}
tic-tac-toe	9	958	100.0	53	10265	9.00 ($\pm$ 0.00)	5.75 ($\pm$ 1.02)	5.79 ($\pm$ 1.03)	5.75 ($\pm$ 1.07)	0.0640	0.0595
monk	16	601	66.85	33	9047	12.26 ($\pm$ 0.96)	8.31 ($\pm$ 1.61)	8.89 ($\pm$ 2.16)	8.42 ($\pm$ 2.09)	0.0672	0.0686
titanic	498	623	79.68	23	3631	61.98 ($\pm$ 15.08)	27.59 ($\pm$ 9.13)	30.17 ($\pm$ 10.82)	28.16 ($\pm$ 10.82)	0.8162	0.8206
biomed	267	209	90.48	23	777	73.54 ($\pm$ 15.09)	32.08 ($\pm$ 8.52)	38.75 ($\pm$ 10.64)	36.05 ($\pm$ 10.04)	0.8134	0.4040
vote	16	434	94.66	25	1321	15.56 ($\pm$ 0.57)	5.87 ($\pm$ 1.23)	5.97 ($\pm$ 1.30)	5.90 ($\pm$ 1.23)	0.0361	0.0135
compas	63	6172	65.77	31	28303	21.75 ($\pm$ 6.31)	11.14 ($\pm$ 4.09)	12.15 ($\pm$ 4.86)	11.43 ($\pm$ 4.28)	0.9147	0.7103
vehicle	272	846	98.43	17	839	53.90 ($\pm$ 8.63)	21.46 ($\pm$ 6.08)	24.94 ($\pm$ 6.97)	24.94 ($\pm$ 6.97)	0.1162	0.1037
heart	400	303	85.71	33	2447	64.53 ($\pm$ 12.32)	27.98 ($\pm$ 8.50)	31.21 ($\pm$ 9.54)	30.19 ($\pm$ 9.1)	9.9494	0.4648
hepatitis	172	142	86.05	35	931	51.16 ($\pm$ 5.96)	18.70 ($\pm$ 7.12)	21.30 ($\pm$ 6.69)	21.30 ($\pm$ 6.69)	0.5841	0.1043
horse	394	299	84.44	29	1771	80.21 ($\pm$ 15.04)	37.99 ($\pm$ 10.14)	42.98 ($\pm$ 11.93)	42.98 ($\pm$ 11.93)	11.4389	0.7641
student.por	142	649	91.79	23	1861	48.44 ($\pm$ 3.97)	18.35 ($\pm$ 5.02)	20.25 ($\pm$ 5.42)	20.21 ($\pm$ 5.36)	1.0457	0.1747
haberman	154	306	69.57	31	3329	56.39 ($\pm$ 8.28)	28.50 ($\pm$ 7.31)	31.29 ($\pm$ 7.18)	30.76 ($\pm$ 7.65)	4.1101	0.5890
employee	72	4653	84.67	19	23063	32.82 ($\pm$ 6.61)	15.03 ($\pm$ 4.99)	17.85 ($\pm$ 7.41)	16.92 ($\pm$ 6.77)	0.1571	0.1790

Table 2: Statistics on the approximation of `PminMAJ` reasons when h is a random forest.

Dataset	#I	#F	%A	$\#T$	$\|F\|$	Path-Times	
						S^*	S_{alg1}
Placement	215	371	95.38	47	1947	239.74	0.61
Cars	406	611	91.8	53	2685	199.55	1.35

Table 3: Experimental results on Placement and Cars

6 CONCLUSION

This work addressed the challenge of generating minimum-size abductive explanations for classification models, focusing on decision trees and random forests. By formulating the problem as a submodular optimization, we leveraged structural properties that enable high-quality approximate solutions. We showed that computing minimum-size abductive explanations for these classifiers is an NP-complete problem, even when restricted to the features of direct paths in decision trees or random forest trees, highlighting the complexity of providing concise explanations. To address this, we developed efficient greedy algorithms with theoretical optimality guarantees, producing near-optimal explanations in reasonable time. Our experiments demonstrated that the greedy algorithm is as effective as exact methods, and sometimes more computationally efficient, generating intelligible and relevant explanations, as shown in our case study on the *placement* benchmark. This makes our approach a viable alternative, particularly in resource-limited environments where modern solvers are costly. Our method, based on submodular optimization, is well-suited for hard-to-explain instances, offering a robust alternative to exact solvers. Future work could explore new formulations of the problem, develop more sophisticated algorithms, and extend our approach to other classification models, including neural networks, to broaden its applicability in diverse contexts.

References

[1] David Alvarez-Melis and Tommi S. Jaakkola. On the robustness of interpretability methods, 2018. URL `https://arxiv.org/abs/1806.08049`.

[2] G. Audemard, S. Bellart, L. Bounia, F. Koriche, J-M. Lagniez, and P. Marquis. Trading complexity for sparsity in random forest explanations. In *Proc. of AAAI'22*, 2022.

[3] G. Audemard, S. Bellart, L. Bounia, F. Koriche, J.M. Lagniez, and P. Marquis. On preferred abductive explanations for decision trees and random forests. In *Proc. of IJCAI'22*, 2022.

[4] Gilles Audemard, Steve Bellart, Louenas Bounia, Frédéric Koriche, Jean-Marie Lagniez, and Pierre Marquis. On the computational intelligibility of boolean classifiers. In *Proc. of KR'21*, pages 74–86, 2021.

[5] Gilles Audemard, Steve Bellart, Louenas Bounia, Frédéric Koriche, Jean-Marie Lagniez, and Pierre Marquis. On the explanatory power of boolean decision trees. *Data Knowledge Engineering*, 142, 2022.

[6] Pablo Barcelo, Mikaël Monet, Jorge A. Perez, and Bernardo Subercaseaux. Model interpretability through the lens of computational complexity. abs/2010.12265, 2020.

[7] Louenas Bounia. *Modèles formels pour l'IA explicable : des explications pour les arbres de décision*. Thèse de doctorat, Université d'Artois, 2023.

[8] Louenas Bounia and Frederic Koriche. Approximating probabilistic explanations via supermodular minimization (corrected version). In *Uncertainty in Artificial Intelligence (UAI 2023).*, pages 216–225, 2023.

[9] Louenas Bounia and Insaf Setitra. Enhancing the intelligibility of decision trees with concise and reliable probabilistic explanations. *Data & Knowledge Engineering*, 2024.

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

[11] Tianqi Chen and Carlos Guestrin. 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, 2016. ISBN 9781450342322.

[12] Andrew Clark, Basel Alomair, Linda Bushnell, and Radha Poovendran. Minimizing convergence error in multi-agent systems via leader selection: A supermodular optimization approach. *IEEE Transactions on Automatic Control*, 59(6):1480–1494, 2014. doi: 10.1109/TAC.2014.2303236.

[13] Andrew Clark, Linda Bushnell, and Radha Poovendran. A supermodular optimization framework for leader selection under link noise in linear multi-agent systems. *IEEE Transactions on Automatic Control*, 59 (2):283–296, 2014. doi: 10.1109/TAC.2013.2281473.

[14] Martin C. Cooper and João Marques-Silva. Tractability of explaining classifier decisions. *Artificial Intelligence*, 2023. doi: https://doi.org/10.1016/j.artint.2022.103841.

[15] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[16] Yves Crama and Peter L. Hammer. Boolean functions - theory, algorithms, and applications. In *Encyclopedia of mathematics and its applications*, 2011.

[17] A. Darwiche and A. Hirth. On the reasons behind decisions. In *Proc. of ECAI'20*, 2020.

[18] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Pai-Shun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Neural Information Processing Systems*, 2018.

[19] Hao Hu, Marie-José Huguet, and Mohamed Siala. Optimizing binary decision diagrams with maxsat for classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, page AAAI Technical Track on Constraint Satisfaction and Optimization, 2022. doi: 10.1609/aaai.v36i4.20291.

[20] Xuanxiang Huang, Yacine Izza, Alexey Ignatiev, Martin Cooper, Nicholas Asher, and João Marques-Silva. Tractable explanations for d-dnnf classifiers. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI)*, pages 5719–5728, 2022.

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

[22] Rishabh K. Iyer and Jeff A. Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *Proc. of NeurIPS'13*, 2013.

[23] Y. Izza and J. Marques-Silva. On explaining random forests with SAT. In *Proc. of IJCAI'21*, 2021.

[24] Yacine Izza, Alexey Ignatiev, and Joao Marques-Silva. On explaining decision trees. *ArXiv*, abs/2010.11034, 2020. URL `https://api.semanticscholar.org/CorpusID:224814214`.

[25] Yacine Izza, Alexey Ignatiev, and Joao Marques-Silva. On tackling explanation redundancy in decision trees. *J. Artif. Intell. Res.*, 75:261–321, 2022.

[26] Bounia Louenas. Enhancing the Intelligibility of Boolean Decision Trees with Concise and Reliable Probabilistic Explanations. In *IPMU 2024*, Lisbao, Portugal, 2024.

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

[28] Joao Marques-Silva and Xuanxiang Huang. Explainability is not a game. *Communications of the ACM*, 2024.

[29] Joao Marques-Silva and Alexey Ignatiev. Delivering trustworthy ai through formal xai. In *AAAI Conference on Artificial Intelligence*, 2022.

[30] G. A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*, 63, 1956.

[31] Ch. Molnar. *Interpretable Machine Learning - A Guide for Making Black Box Models Explainable*. 2019.

[32] George Nemhauser and Laurence Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, Inc., New York, 1988.

[33] 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.

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

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

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

[37] L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.