On the Explanatory Power of Decision Trees

Anonymous Author(s)
Affiliation
Address
email

Abstract

Decision trees have long been recognized as models of choice in sensitive applications where interpretability is of paramount importance. In this paper, we examine the computational ability of Boolean decision trees in deriving, minimizing, and counting sufficient reasons and contrastive explanations. We prove that the set of all sufficient reasons of minimal size for an instance given a decision tree can be exponentially larger than the size of the input (the instance and the decision tree). Therefore, generating the full set of sufficient reasons can be out of reach. In addition, computing a single sufficient reason does not prove enough in general; indeed, two sufficient reasons for the same instance may differ on many features. To deal with this issue and generate synthetic views of the set of all sufficient reasons, we introduce the notions of relevant features and of necessary features that characterize the (possibly negated) features appearing in at least one or in every sufficient reason, and we show that they can be computed in polynomial time. We also introduce the notion of explanatory importance, that indicates how frequent each (possibly negated) feature is in the set of all sufficient reasons. We show how the explanatory importance of a feature and the number of sufficient reasons can be obtained via a model counting operation, which turns out to be practical in many cases. We also explain how to enumerate sufficient reasons of minimal size. We finally show that, unlike sufficient reasons, the set of all contrastive explanations for an instance given a decision tree can be derived, minimized and counted in polynomial time.

1 Introduction

In essence, explaining a decision to a person is to give the details or *reasons* that help a person (the explainee) understand why the decision has been made. This is a significant issue especially when decisions are made by Machine Learning (ML) models, such as random forests, Markov networks, support vector machines, and deep neural networks. Actually, with the growing number of applications that rely on ML techniques, researches on eXplainable AI (XAI) have become increasingly important, by providing efficient methods for interpreting ML models, and explaining their decisions (see for instance [10, 11, 12, 13, 16, 19, 22, 23, 24, 28, 30]).

When dealing with Boolean classifiers, which is what we do in this paper, two decisions are possible, only: 1 for the instances classified as positive instances, and 0 for the remaining ones (the negative instances). Whatever the way $\boldsymbol{x}$ has been classified, an explainee may seek for explanations from two distinct types [23]. On the one hand, abductive explanations for $\boldsymbol{x}$ are intended to explain why $\boldsymbol{x}$ has been classified in the way it has been classified by the ML model (thus, addressing the "Why?" question). On the other hand, the purpose of contrastive (also known as counterfactual) explanations for $\boldsymbol{x}$ is to explain why $\boldsymbol{x}$ has not been classified by the ML model as the explainee expected it (thus, addressing the "Why not?" question). In both cases, explanations that are as simple as possible are preferred (where simplicity is modeled as irredundancy, or even as size minimality).

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Although there is no formal notion of *interpretability* [21], for classification problems, *decision trees* [3, 26] are arguably among the most interpretable ML models. Because of their interpretability, decision trees are often considered as target models for distilling a black-box model into a comprehensible one [4, 10]. Furthermore, decision trees are often the components of choice for building (less interpretable, but potentially more accurate) ensemble classifiers, such as random forests [2] and gradient boosted decision trees [5].

The interpretability of decision trees is endowed with two key characteristics. On the one hand, decision trees are *transparent*: each node in a decision tree has some meaning, and the principles used for generating all nodes can be explained. On the other hand, decision trees are *locally explainable*: by construction of a decision tree T, any input instance $\boldsymbol{x}$ is mapped to a unique root-to-leaf path that yields to a decision label. The subset of (positive and negative) features $t_{\boldsymbol{x}}^T$ occurring in the path used to find the right label 1 or 0 for $\boldsymbol{x}$ in the decision tree T can be viewed as a "direct reason" for classifying $\boldsymbol{x}$ as a positive instance or as a negative instance. $t_{\boldsymbol{x}}^T$ is an abductive explanation for $\boldsymbol{x}$ given T, which explains why $\boldsymbol{x}$ has been classified by T as it has been classified. Indeed, every instance $\boldsymbol{x}'$ that coincides with $\boldsymbol{x}$ on $t_{\boldsymbol{x}}^T$ is classified by T in the same way as $\boldsymbol{x}$. However, such "direct reasons" can contain arbitrarily many redundant features [17]. This motivates to take account for other types of abductive explanations in the case of decision trees, namely, sufficient reasons [7] (also known as prime implicant explanations [29]), that are irredundant abductive explanations, and minimal sufficient reasons (i.e., those sufficient reasons of minimal size).

In this paper, we examine the computational ability of Boolean decision trees in deriving, minimizing and counting sufficient reasons and contrastive explanations. We prove that the set of all sufficient reasons of minimal size for an instance given a decision tree can be exponentially larger than the size of the input. When this is the case, generating the full set of sufficient reasons (i.e., the complete reason for the instance [7]) is typically out of reach. In addition, computing a single sufficient reason does not prove enough in general; indeed; two sufficient reasons for the same instance may differ on many features. To deal with this issue and generate synthetic views of the set of all sufficient reasons, we introduce the notions of relevant features and of necessary features that characterize the (possibly negated) features appearing in at least one or in every sufficient reason, and we show that they can be computed in polynomial time. We also introduce the notion of explanatory importance, that indicates how frequent each (possibly negated) feature is in the set of all sufficient reasons. Though deriving the explanatory importance of a feature in the set of sufficient reasons and determining the cardinality of this set are two computationally demanding tasks, we show how they can be achieved thanks to model counting operation, which turns out to be practical in many cases. We also explain how to enumerate sufficient reasons of minimal size, which is a way to count them when they are not too numerous. We finally show that, from a computational standpoint, contrastive explanations highly depart from sufficient reasons. Indeed, the set of all contrastive explanations for an instance given a decision tree can be computed in polynomial time. As a consequence, such explanations can also be minimized and counted in polynomial time.

The rest of the paper is organized as follows. Preliminaries about decision trees, abductive reasons, and contrastive explanations are given in Section 2. The computation of all sufficient reasons is considered in Section 3. Necessary and relevant features are presented in this section, as well as the approach for assessing the explanatory importance of a feature and for counting the number of sufficient reasons. We also explain there how minimal sufficient reasons can be enumerated. An algorithm for computing all the contrastive explanations for the instance given the decision tree is presented in Section 4. Experimental results are reported in Section 5. Finally, Section 6 concludes the paper. All the proofs and additional empirical results are reported as a supplementary material.

2 Decision Trees, Abductive and Contrastive Explanations

For an integer n, let $[n]$ be the set $\{1, \cdots, n\}$. By $\mathcal{F}_n$ we denote the class of all Boolean functions from $\{0,1\}^n$ to $\{0,1\}$, and we use $X_n = \{x_1, \cdots, x_n\}$ to denote the set of input Boolean variables, corresponding to the features under consideration. Any assignment $\boldsymbol{x} \in \{0,1\}^n$ is called an *instance*. If $f(\boldsymbol{x}) = 1$ for some $f \in \mathcal{F}_n$, then $\boldsymbol{x}$ is called a *model* of f. $\boldsymbol{x}$ is a *positive instance* when $f(\boldsymbol{x}) = 1$ and a *negative instance* when $f(\boldsymbol{x}) = 0$.

We refer to f as a *propositional formula* when it is described using the Boolean connectives $\wedge$ (conjunction), $\vee$ (disjunction) and $\neg$ (negation), together with the Boolean constants 1 (true) and 0

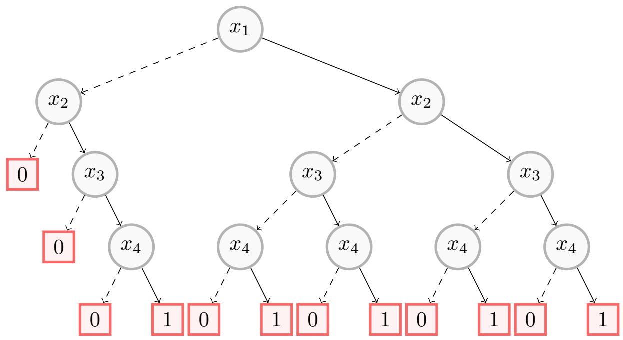

Figure 1: A decision tree T for recognizing *Cattleya* orchids. The left (resp. right) child of any decision node labelled by x_i corresponds to the assignment of x_i to 0 (resp. 1).

(false). As usual, a *literal* ℓ is a variable x_i (a positive literal) or its negation $\neg x_i$, also denoted $\overline{x}_i$ (a negative literal). A positive literal x_i is associated with a positive feature (i.e., x_i is set to 1), while a negative literal $\overline{x}_i$ is associated with a negative feature (i.e., x_i is set to 0). A *term* (or *monomial*) t is a conjunction of literals, 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. The set of variables occurring in a formula f is denoted $Var(f)$. A formula f is *consistent* if and only if it has a model. A `CNF` formula is *monotone* whenever every occurrence of a literal in the formula has the same polarity (i.e., if a literal occurs positively (resp. negatively) in the formula, then it does not have any negative (resp. positive) occurrence in the formula). A formula f_1 *implies* a formula f_2, noted $f_1 \models f_2$, if and only if every model of f_1 is a model of f_2. Two formulae f_1 and f_2 are *equivalent*, noted $f_1 \equiv f_2$ whenever they have the same models. The *conditioning* of a formula f by a literal ℓ, denoted $f \mid \ell$, is the formula obtained from f by replacing each occurrence of x_i with 1 (resp. 0) and each occurrence of $\overline{x}_i$ with 0 (resp. 1) if $\ell = x_i$ (resp. $\ell = \overline{x}_i$).

In what follows, we shall often treat assignments as terms, and terms and clauses as sets of literals. Given an assignment $\boldsymbol{z} \in \{0,1\}^n$, the corresponding term is defined as

$$t_{\boldsymbol{z}} = \bigwedge_{i=1}^{n} x_i^{z_i} \text{ where } x_i^0 = \overline{x}_i \text{ and } x_i^1 = x_i$$

A term t *covers* an assignment $\boldsymbol{z}$ if $t \subseteq t_{\boldsymbol{z}}$. 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 proper subset of t is an implicant of f. Dually, an *implicate* of a Boolean function f is a clause that is implied by f, and a *prime implicate* of f is an implicate c of f such that no proper subset of c is an implicate of f.

With these basic notions in hand, we shall focus on the following representation class of Boolean functions:

Definition 1 (Decision Tree)**.** *A (Boolean)* decision tree *is a binary tree T, each of whose internal nodes is labeled with one of n input Boolean variables, and whose leaves are labeled 0 or 1. Every variable is assumed (without loss of generality) to appear 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, go to the left or right child depending on whether the input value of the corresponding variable is 0 or 1, respectively. The size of T, denoted $|T|$, is given by the number of its nodes.*

The class of decision trees over X_n is denoted $\texttt{DT}_n$. It is well-known that any decision tree $T \in \texttt{DT}_n$ can be transformed in linear time into an equivalent disjunction of terms, denoted $\texttt{DNF}(T)$, where each term corresponds to a path from the root to a leaf labeled with 1. Dually, T can be transformed in linear time into a conjunction of clauses, denoted $\texttt{CNF}(T)$, where each clause is the negation of the term describing a path from the root to a leaf labeled with 0.

For illustration, the following toy example will be used throughout the paper as a running example:

Example 1. *The decision tree in Figure 1 separates 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".*

As a salient characteristic, decision trees convey a single explicit abductive explanation for classifying any input instance:

Definition 2 (Direct Reason)**.** *Let $T \in \mathtt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$. The* direct reason *for $\boldsymbol{x}$ given T is the term, denoted $t_{\boldsymbol{x}}^T$, corresponding to the unique root-to-leaf path of T that is compatible with $\boldsymbol{x}$.*

Another important notion of abductive explanations is the following concept of *sufficient reason*[7], that, unlike the notion of direct reason, is not specific to decision trees:

Definition 3 (Sufficient Reason)**.** *Let $f \in \mathcal{F}_n$ and $\boldsymbol{x} \in \{0,1\}^n$ such that $f(\boldsymbol{x}) = 1$ (resp. $f(\boldsymbol{x}) = 0$). A* sufficient reason *for $\boldsymbol{x}$ given f is a prime implicant t of f (resp. $\neg f$) that covers $\boldsymbol{x}$. $sr(\boldsymbol{x}, f)$ denotes the set of sufficient reasons for $\boldsymbol{x}$ given f.*

Thus, a sufficient reason [7] (also known as prime implicant explanation [29]) for an instance $\boldsymbol{x}$ given a class described by a Boolean function f is a subset t of the characteristics of $\boldsymbol{x}$ that is minimal w.r.t. set inclusion such that any instance $\boldsymbol{x}'$ sharing this set t of characteristics is classified by f as $\boldsymbol{x}$ is. Thus, when $f(\boldsymbol{x}) = 1$, t is a sufficient reason for $\boldsymbol{x}$ given f if and only if t is a prime implicant of f such that $\boldsymbol{x}$ implies t, and when $f(\boldsymbol{x}) = 0$, t is a sufficient reason for $\boldsymbol{x}$ given f if and only if t is a prime implicant of $\neg f$ such that t covers $\boldsymbol{x}$. Accordingly, sufficient reasons are suited to explain why the instance at hand $\boldsymbol{x}$ has been classified by f as it has been classified. Unlike direct reasons [17], sufficient reasons do not contain any redundant feature.

When considering the sufficient reasons of the input instance, one may be interested in focusing on the shortest ones, alias the minimal sufficient reasons. Those reasons are valuable since conciseness is often a desirable property of explanations (Occam's razor). Formally:

Definition 4 (Minimal Sufficient Reason)**.** *Let $f \in \mathcal{F}_n$ and $\boldsymbol{x} \in \{0,1\}^n$. A* minimal sufficient reason *for $\boldsymbol{x}$ given f is a sufficient reason for $\boldsymbol{x}$ given f that contains a minimal number of literals.*

Finally, unlike direct and (possibly minimal) sufficient reasons that aim to explain the classification of the instance $\boldsymbol{x}$ under consideration as achieved by the classifier f, contrastive explanations are valuable when $\boldsymbol{x}$ has not been classified by f as expected by the explainee. In this case, one looks for minimal subsets of the features that when switched in $\boldsymbol{x}$ are enough to get instances that are classified positively (resp. negatively) by f if $\boldsymbol{x}$ is classified negatively (resp. positively) by f. Formally, a *contrastive explanation* for $\boldsymbol{x}$ given f [15] is a subset t of the characteristics of $\boldsymbol{x}$ that is minimal w.r.t. set inclusion among those such that at least one instance $\boldsymbol{x}'$ that coincides with $\boldsymbol{x}$ except on the characteristics from t is not classified by f as $\boldsymbol{x}$ is.

Definition 5 (Contrastive Explanation)**.** *Let $f \in \mathcal{F}_n$ and $\boldsymbol{x} \in \{0,1\}^n$ such that $f(\boldsymbol{x}) = 1$ (resp. $f(\boldsymbol{x}) = 0$). A* contrastive explanation *for $\boldsymbol{x}$ given f is a term t over X_n such that $t \subseteq t_{\boldsymbol{x}}$, $t_{\boldsymbol{x}} \setminus t$ is not an implicant of f (resp. $\neg f$), and for every $\ell \in t$, $t \setminus \{\ell\}$ does not satisfy this last condition.*

Example 2. *Based on our running example, we can observe that $T(\boldsymbol{x}) = 1$ for the instance $\boldsymbol{x} = (1,1,1,1)$. The direct reason for $\boldsymbol{x}$ given T is the term $t_{\boldsymbol{x}}^T = x_1 \wedge x_2 \wedge x_3 \wedge x_4$. $x_1 \wedge x_4$ and $x_2 \wedge x_3 \wedge x_4$ are the sufficient reasons for $\boldsymbol{x}$ given T. $x_1 \wedge x_4$ is the unique minimal sufficient reason for $\boldsymbol{x}$ given T. x_4, $x_1 \wedge x_2$, and $x_1 \wedge x_3$ are the contrastive explanations for $\boldsymbol{x}$ given T. Thus, the instance $(1,1,1,0)$ that differs with $\boldsymbol{x}$ only on x_4 is not classified by T as $\boldsymbol{x}$ is ($(1,1,1,0)$ is classified as a negative instance).*

We mention in passing that when dealing with decision trees T, we could have focused only on explanations for the *positive* instances $\boldsymbol{x}$ given T. This comes from the fact that $\mathtt{DT}_n$ is closed under negation, in the sense that for any $T \in \mathtt{DT}_n$, $\neg T$ can be obtained by just replacing from T the label of each leaf with its complement. So, for any instance $\boldsymbol{x} \in \{0,1\}^n$, a direct reason (resp. sufficient reason, minimal sufficient reason, contrastive explanation) explaining why $T(\boldsymbol{x}) = 0$ is precisely the same as a direct reason (resp. sufficient reason, minimal sufficient reason, contrastive explanation) explaining why $(\neg T)(\boldsymbol{x}) = 1$. Considering T or its negation $\neg T$ has no computational impact since $\neg T$ can be computed in time linear in the size of T.

3 Computing All Sufficient Reasons

Sufficient reasons can be exponentially numerous. When switching from the direct reason for an instance (that is unique but not always redundancy-free) to its sufficient reasons, a main obstacle to be dealt with lies in the number of reasons to be considered. Indeed, even for the restricted class

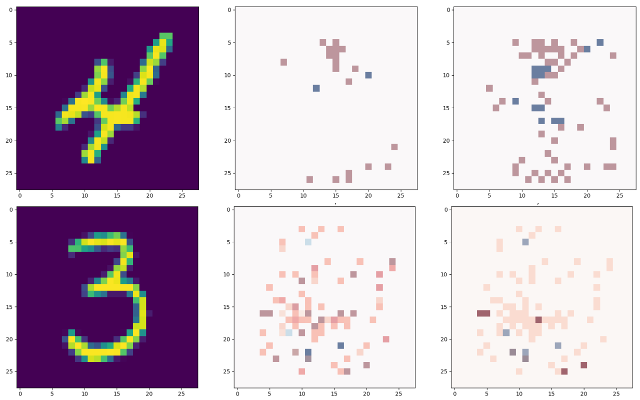

Figure 2: Two sufficient reasons for an `mnist` instance (top), and an explanatory heat map and the explanatory features for an `mnist` instance (bottom).

of decision trees with logarithmic depth, an input instance can have exponentially many sufficient reasons:

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

By definition, the minimal sufficient reasons for $\boldsymbol{x}$ given T cannot be more numerous than its sufficient reasons. However, focusing on minimal sufficient reasons does not solve the problem since an instance can also have exponentially many minimal sufficient reasons:

Proposition 2. *For every* $n \in \mathbb{N}$ *such that* n *is odd, there is a decision tree* $T \in \mathtt{DT}_n$ *of depth* $\frac{n+1}{2}$ *such that* T *contains* $2n+1$ *nodes and there is an instance* $\boldsymbol{x} \in \{0,1\}^n$ *such that the number of minimal sufficient reasons for* $\boldsymbol{x}$ *given* T *is equal to* $2^{\sqrt{n-1}}$.

In many practical cases, the number of sufficient reasons for an instance given a decision tree can be very large. Figure 2 (top) shows an `mnist` instance (the leftmost subfigure) that has 482 185 073 664 sufficient reasons. Among them there are very dissimilar sufficient reasons. As an illustration, the two rightmost subfigures present two sufficient reasons for this instance, and they differ on many features (blue (resp. red) dots correspond to pixels on (resp. off)).

For such datasets, computing the set of all the sufficient reasons for a given instance is not always feasible. Furthermore, if the computation succeeds but the number of sufficient reasons is huge, their (disjunctively interpreted) set, alias the complete reason for the instance [7], can hardly be considered as intelligible by the explainee. Finally, due to the number of sufficient reasons and their diversity, deriving one of them is not informative enough. Thus, one needs to design approaches to synthesizing their set while avoiding the two pitfalls (the computational one and the informational one).

Synthesizing the set of sufficient reasons. In this objective, the following notions of *necessary / (ir)relevant features* appear useful. These notions of necessity and relevance echo the ones that have been considered in [9] for logic-based abduction.

Definition 6 (Explanatory Features)**.** *Let* $f \in \mathcal{F}_n$*, and* $\boldsymbol{x} \in \{0,1\}^n$ *be an instance. Let* e *be an explanation type.*[1]

- *A literal* ℓ *over* X_n *is a* necessary feature *for the family* e *of explanations for* $\boldsymbol{x}$ *given* f *if and only if* ℓ *belongs to every explanation* t *for* $\boldsymbol{x}$ *given* f *such that* t *is of type* e*.* $Nec_e(\boldsymbol{x}, f)$ *denotes the set of all necessary features for the family* e *of explanations for* $\boldsymbol{x}$ *given* f*.*
- *A literal* ℓ *over* X_n *is a* relevant feature *for the family* e *of explanations for* $\boldsymbol{x}$ *given* f *if and only if* ℓ *belongs to at least one explanation* t *for* $\boldsymbol{x}$ *given* f *such that* t *is of type* e*.* $Rel_e(\boldsymbol{x}, f)$ *denotes the set of all relevant features for the family* e *of explanations for* $\boldsymbol{x}$

[1] For instance, e can be s when the sufficient reasons for $\boldsymbol{x}$ given f are targeted or c when the contrastive explanations for $\boldsymbol{x}$ given f are targeted.

given f. $Irr_e(\boldsymbol{x}, f)$, which is the complement of $Rel_e(\boldsymbol{x}, f)$ in the set of all literals over X_n, denotes the set of all irrelevant features for the family e of explanations for $\boldsymbol{x}$ given f.

The necessary (resp. irrelevant) features for the family s of sufficient reasons for $\boldsymbol{x}$ given f are the most (resp. less) important features for explaining the classification of $\boldsymbol{x}$ by f, since they belong to every (resp. no) sufficient reason for $\boldsymbol{x}$ given f.

When a single sufficient reason t for $\boldsymbol{x}$ given f has been computed, the cardinality of t deprived from the features of $Nec_s(\boldsymbol{x}, f)$ is small, and the cardinality of the symmetric difference between t and $Rel_s(\boldsymbol{x}, f)$ is small as well, t can be viewed as a good representative of the complete reason for $\boldsymbol{x}$ given f in the sense that a sufficient reason t' for $\boldsymbol{x}$ given f that differs a lot from t cannot exist.

In the case when f is a decision tree T, though the set of all sufficient reasons for $\boldsymbol{x}$ given T cannot be generated when it is too large, $Nec_s(\boldsymbol{x}, f)$, $Rel_s(\boldsymbol{x}, f)$, and $Irr_s(\boldsymbol{x}, f)$ can be derived efficiently:

Proposition 3. *Let $T \in \mathtt{DT}_n$, and $\boldsymbol{x} \in \{0,1\}^n$. Computing $Nec_s(\boldsymbol{x}, T)$, $Rel_s(\boldsymbol{x}, f)$, and $Irr_s(\boldsymbol{x}, T)$ can be done in $\mathcal{O}((n + |T|) \times |T|)$ time.*

Going a step further consists in evaluating the explanatory importance of every (positive or negative) feature:

Definition 7 (Explanatory Importance)**.** *Let $f \in \mathcal{F}_n$, and $\boldsymbol{x} \in \{0,1\}^n$ be an instance. Let e be an explanation type, and $E_e(\boldsymbol{x}, f)$ the set of all explanations for $\boldsymbol{x}$ given f that are of type e. The* explanatory importance *of a literal ℓ over X_n for $\boldsymbol{x}$ given f w.r.t. e is given by*

$$Imp_e(\ell, \boldsymbol{x}, f) = \frac{\#(\{t \in E_e(\boldsymbol{x}, f) : \ell \in t\})}{\#(E_e(\boldsymbol{x}, f))}.$$

Example 3. *On the running example, we have $Nec_s(\boldsymbol{x}, T) = \{x_4\}$, and $Rel_s(\boldsymbol{x}, T) = \{x_1, x_2, x_3, x_4\}$. We also have $Imp_s(x_4, \boldsymbol{x}, T) = 1$, $Imp_s(x_1, \boldsymbol{x}, T) = Imp_s(x_2, \boldsymbol{x}, T) = Imp_s(x_3, \boldsymbol{x}, T) = \frac{1}{2}$, and $Imp_s(\ell, \boldsymbol{x}, T) = 0$ for every other literal ℓ (the negative ones over $\{x_1, x_2, x_3, x_4\}$).*

The notion of explanatory importance must not be confused with the notions of feature importance (which can be defined and assessed in many different ways): the former is local (i.e., relative to an instance) and not global, it concerns literals and not variables (polarity matters), and it is about the explanation task, not the prediction one.

In order to compute the explanatory importance of a literal, a straightforward approach consists in enumerating the explanations of $E_e(\boldsymbol{x}, f)$. This is feasible when this set is not too large, which is not always the case for sufficient reasons even when f is a decision tree T. Thus, for dealing with the remaining case, an alternative approach must be looked for.

We designed such an approach for computing $Imp_s(\ell, \boldsymbol{x}, T)$. We know that $sr(\boldsymbol{x}, T)$ is by construction the set of prime implicants of $g = \{c \cap t_{\boldsymbol{x}} : c \in \mathtt{CNF}(T)\}$. Thus, we exploited the translation presented in [18] showing how to associate in polynomial time with a given `CNF` formula (here, g) another formula (over a distinct set of variables), let us say h, such that the models of h are in one-to-one correspondence with the prime implicants of g. In our case, the translation can be simplified because g is a monotone `CNF` formula. Since h is not primarily a `CNF` formula, leveraging Tseitin transformation [31], we turned h in linear time into a query-equivalent `CNF` formula i. Note that every auxiliary variable that is introduced in i is defined from the other variables (those occurring in h), so that the number of models of i is the same as the number of models of h. Finally, we took advantage of the compilation-based model counter `D4` [20] to compile i into a `d-DNNF` circuit [6], and this enabled us to compute in time polynomial in the size of i both the number of sufficient reasons and the explanatory importance of every literal (indeed, the `d-DNNF` language supports in polytime the model counting query and the conditioning transformation [8]). We show in Section 5 that, despite a high complexity in the worst case (the size of i can be exponential in $|T|$), this approach based on knowledge compilation proves quite efficient in practice.

Clearly enough, when $Imp_e(\ell, \boldsymbol{x}, T)$ has been computed for every ℓ, one can easily generate explanatory heat maps. Figure 2 (bottom) shows an `mnist` instance (the leftmost subfigure) that has 19 115 685 sufficient reasons, 6 necessary literals, and 94 relevant literals. The central subfigure is the corresponding heat map. Blue (resp. red) pixels correspond to positive (resp. negative) literals in the instance, and the intensity of the color aims to reflect the explanatory importance of the corresponding literal. The rightmost subfigure gives the explanatory features (dark pixels are associated with necessary literals, and light pixels to relevant literals).

Enumerating the minimal sufficient reasons. An approach to synthesizing the set of sufficient reasons consists in focusing on the minimal ones. Indeed, though the set of minimal sufficient reasons for an instance given a decision tree can be exponentially large, the number of minimal sufficient reasons cannot exceed the number of sufficient reasons, and it can be significantly lower in practice.

However, unlike sufficient reasons that can be generated in polynomial time using a greedy algorithm (see e.g., [17]), computing minimal reasons is not an easy task:

Proposition 4. *Let $T \in \texttt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$. Computing a minimal sufficient reason for $\boldsymbol{x}$ given T is* NP*-hard.*

Despite this intractability result, minimal sufficient reasons can be generated in many practical cases. A common approach for handling NP-optimization problems is to rely on modern constraint solvers. One follows this direction here and casts the task of finding minimal sufficient reasons as a Boolean constraint optimization problem. We first need to recall that a PARTIAL MAXSAT problem consists of a pair $(C_{\text{soft}}, C_{\text{hard}})$ where C_{soft} and C_{hard} are (finite) set of clauses. The goal is to find a Boolean assignment that maximizes the number of clauses c in C_{soft} that are satisfied, while satisfying all clauses in C_{hard}.

Proposition 5. *Let T be a decision tree in $\texttt{DT}_n$ and $\boldsymbol{x} \in \{0,1\}^n$ be an instance such that $T(\boldsymbol{x}) = 1$. Let $(C_{\text{soft}}, C_{\text{hard}})$ be an instance of the* PARTIAL MAXSAT *problem such that:*

$$C_{\text{soft}} = \{\overline{x_i} : x_i \in t_{\boldsymbol{x}}\} \cup \{x_i : \overline{x}_i \in t_{\boldsymbol{x}}\} \text{ and } C_{\text{hard}} = \{c \cap t_{\boldsymbol{x}} : c \in \texttt{CNF}(T)\}.$$

The intersection of $t_{\boldsymbol{x}}$ with $t_{\boldsymbol{x}^*}$ where $\boldsymbol{x}^*$ is an optimal solution of $(C_{\text{hard}}, C_{\text{soft}})$, is a minimal sufficient reason for $\boldsymbol{x}$ given T.

Clearly enough, if $\boldsymbol{x}$ is such that $T(\boldsymbol{x}) = 0$, then it is enough to consider the same instance of PARTIAL MAXSAT as above, except that $C_{\text{hard}} = \{c \cap t_{\boldsymbol{x}} : c \in \texttt{CNF}(\neg T)\}$.

Finally, one can take advantage of this PARTIAL MAXSAT characterization for generating a preset number of minimal sufficient reasons (basically, one generates a first reason t, then one adds to C_{hard} the negation of t as a clause as well as a `CNF` encoding of a cardinality constraint for ensuring that the next reasons to be generated have the same size as the one of t, and we resume until the bound is reached or no solution exists).

4 Computing All Contrastive Explanations

Interestingly, it has been shown that sufficient reasons and contrastive explanations are connected by a minimal hitting set duality [15]. This duality can be leveraged to derive one of the two sets of explanations from the other one using algorithms for computing minimal hitting sets [27, 32].

However, in the case of decision trees, a more direct and much more efficient approach to derive all the contrastive explanations for $\boldsymbol{x} \in \{0,1\}^n$ given $T \in \texttt{DT}_n$ can be designed. Indeed, unlike what happens for sufficient reasons (see Section 3), the set of *all* contrastive explanations for $\boldsymbol{x} \in \{0,1\}^n$ given a decision tree $T \in \texttt{DT}_n$ can be computed in polynomial time from $\boldsymbol{x}$ and T:

Proposition 6. *The set of all contrastive explanations for $\boldsymbol{x} \in \{0,1\}^n$ given a decision tree $T \in \texttt{DT}_n$ can be computed in time polynomial in $n + |T|$ as $min(\{c \cap t_{\boldsymbol{x}} : c \in \texttt{CNF}(f)\}, \subseteq)$.*

Example 4. *On the running example, we have $\texttt{CNF}(T) = \{x_1 \vee x_2, x_1 \vee \overline{x_2} \vee x_3, x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4, \overline{x_1} \vee x_2 \vee x_3 \vee x_4, \overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4, \overline{x_1} \vee \overline{x_2} \vee x_3 \vee x_4, \overline{x_1} \vee \overline{x_2} \vee \overline{x_3} \vee x_4\}$. Thus, with $\boldsymbol{x} = (1,1,1,1)$, we have $min(\{c \cap t_{\boldsymbol{x}} : c \in \texttt{CNF}(f)\}, \subseteq) = \{x_1 \vee x_2, x_1 \vee x_3, x_4\}$, which corresponds to the contrastive explanations $x_1 \wedge x_2$, $x_1 \wedge x_3$, x_4 for $\boldsymbol{x}$ given T (viewing clauses and terms as sets of literals).*

As straightforward consequences of Proposition 6, computing necessary / relevant features and computing the explanatory importance of features w.r.t. contrastive explanations can be achieved in time polynomial in $n + |T|$. Similarly, statistics about the size of contrastive explanations can be easily established, and contrastive explanations can be easily minimized and counted.

5 Experiments

Empirical setting. We have considered 90 datasets, which are standard benchmarks from the well-known repositories Kaggle (`www.kaggle.com`), OpenML (`www.openml.org`), and UCI (`archive.`

Table 1: Empirical results based on 12 datasets.

Dataset	Decision Tree			|Sufficient|		|Minimal|		#Nec. Features		#Rel. Features	
	%A	#N	#B	med	max	med	max	med	max	med	max
recidivism	63.41	13828.80	147.60	14	22	13	22	6	19	60	98
adult	81.36	12934.00	2974.80	16	36	16	36	7	22	263	543
bank marketing	87.40	6656.40	1432.60	14	21	14	21	3	16	247	398
bank	88.99	5523.60	977.80	13	24	13	24	4	15	200	330
lending loan	73.49	2610.40	1131.40	16	31	16	31	8	25	226	442
contraceptive	50.44	1252.20	88.60	11	20	11	20	8	17	25	47
compas	65.98	1230.00	46.20	6	14	6	14	3	12	16	33
christine	63.36	853.20	426	12	47	12	47	8	41	92	202
farm-ads	86.75	544.80	264.60	20	99	20	99	16	92	73	192
mnist49	95.47	539.60	267.90	22	30	22	30	9	19	91	166
spambase	91.94	536.40	264.80	15	29	15	29	9	24	68	146
mnist38	96.07	506.60	251.40	19	28	19	28	8	20	93.50	157

Dataset	#Sufficient		#Contrastive		|Contrastive|		#Minimal	
	med	max	med	max	med	max	med	max
recidivism	10387	9734080	54	145	3	16	2	144
adult	-	$\geq$ 157383572260730000000000	201	470	4	16	3	256
bank marketing	-	$\geq$ 746037521348435000000	189	337	4	13	8	432
bank	-	$\geq$ 743395197901850000	150	277	4	13	4	168
lending loan	459258918095775	9432432428162030000000000000	157	311	3	12	3	192
contraceptive	20,50	4272	21	52	2	11	2	48
compas	16	444	13	33	2	11	2	21
christine	63108	2167735434744	71	151	3	8	2	4096
farm-ads	1177,50	921895392	59	166	2	10	-	$\geq$ 10000
mnist49	7392384	715892613696000	61	106	2	12	-	$\geq$ 10000
spambase	15712	2535069312	50	107	2	11	4	384
mnist38	14849376	16922386736640	62	107	3	11	32	3072

`ics.uci.edu/ml/`). `mnist38` and `mnist49` are subsets of the `mnist` dataset, restricted to the instances of 3 and 8 (resp. 4 and 9) digits. Because some datasets are suited to the multi-label classification task, we used the standard "one versus all" policy to deal with them: all the classes but the target one are considered as the complementary class of the target. Categorical features have been treated as arbitrary numbers (the scale is nominal). As to numeric features, no data preprocessing has taken place: these features have been binarized on-the-fly by the decision tree learning algorithm that has been used.

For every benchmark b, a 10-fold cross validation process has been achieved. Namely, a set of 10 decision trees T_b have been computed and evaluated from the labelled instances of b, partitioned into 10 parts. One part was used as the test set and the remaining 9 parts as the training set for generating a decision tree. This tree is thus in 1-to-1 correspondence with the test set chosen within the whole dataset b. The classification performance for b was measured as the mean accuracy obtained over the 10 decision trees generated from b. The CART algorithm, and more specifically its implementation provided by the Scikit-Learn library [25] has been used to learn decision trees. All hyper-parameters of the learning algorithm have been set to their default value. Notably, decision trees have been learned using the Gini criterion, and without any maximal depth or any other manual limitation.

For each benchmark b, each decision tree T_b, and a subset of at most 100 instances $\boldsymbol{x}$ picked up at random in the test set following a uniform distribution, we computed a sufficient reason for $\boldsymbol{x}$ given T_b (using the standard greedy algorithm run on the direct reason $t_{\boldsymbol{x}}^{T_b}$), and a minimal sufficient reason for $\boldsymbol{x}$ given T_b using the PARTIAL MAXSAT encoding presented in Proposition 5. This enabled us to draw some statistics (median, maximum) about the sizes of the reasons that have been generated. Using the algorithm presented in the proof of Proposition 3, we also derived the necessary and relevant explanatory features for each $\boldsymbol{x}$, and again drew some statistics about them. Exploiting the model counter `D4`, we computed the number of sufficient reasons for $\boldsymbol{x}$ given T_b, as well as the explanatory importance of every feature. Taking advantage of the algorithm given in Proposition 4, we computed the number of contrastive explanations for $\boldsymbol{x}$ given T_b, and drew some statistics about those numbers and about the sizes of the contrastive explanations. Finally, using the approach described in Section 3, we enumerated all the minimal sufficient reasons for $\boldsymbol{x}$ given T_b up to a limit of 10 000, and again drew some statistics about the numbers of minimal sufficient reasons. Of course, for each computation, we measured the corresponding runtimes since this is fundamental to determine the extent to which the algorithms are practical (details are provided as a supplementary material).

All the experiments have been conducted on a computer equipped with Intel(R) XEON E5-2637 CPU @ 3.5 GHz and 128 GiB of memory. `D4` [20] was run with its default parameters. For computing minimal reasons, we used the Pysat library [14], which provides the implementation of the RC2 PARTIAL MAXSAT solver. This solver was run using the parameters corresponding to the "Glucose" setting. A time-out of 100s per instance was set for `D4`.

Results. Table 1 (top and bottom) reports an excerpt of our results, focusing on 12 benchmarks out of 90 (the selected datasets are among those containing many instances and/or many features). The leftmost column gives the name of the dataset b. Columns $\%A$, $\%N$, and $\#B$ give, respectively, the mean accuracy over the 10 decision trees, the average number of nodes in those trees, and the average number of binary features they are based on. The next columns give statistics (median, maximum) about, respectively, the size of the sufficient reasons (|Sufficient|) and of the minimal sufficient reasons (|Minimal|) that have been computed, as well as about the number of necessary ($\#$Nec. Features) and relevant ($\#$Rel. Features) features that appear in the full set of sufficient reasons for the instance. Table 1 (bottom) give statistics (median, maximum) about, respectively, the number of sufficient reasons ($\#$Sufficient), the number of contrastive explanations ($\#$Contrastive) and their sizes (|Contrastive|), and finally the number of minimal sufficient reasons ($\#$Minimal).

As to the computation times, it turns out that all the algorithms described in the previous sections proved as efficient in practice. This is not surprising for those algorithms having a polytime worst-case complexity (the greedy algorithm for computing a sufficient reason, the one for deriving explanatory features, and the one for computing all the contrastive explanations). It was less obvious at first sight for the algorithms used for counting the number of sufficient reasons and for computing the explanatory importance of features. However, all the computations that have been run have terminated in due time, except for 3 datasets out of 90, namely `adult`, `bank_marketing`, and `bank`. For these datasets, the time limit of 100s has been reached for, respectively, 203, 150, and 336 instances out of 1000 (in this case, the median number of sufficient reasons has not been reported). Notably, for all the 90 datasets but those 3, the median time required for counting the number of sufficient reasons and computing the explanatory importance of features never exceeded 1s. Computing a minimal sufficient reason, and more generally all such reasons looked challenging as well, due to both the intrinsic complexity of computing a minimal sufficient reason and to their number. Nevertheless, our enumeration algorithm succeeded in deriving *all the minimal sufficient reasons* for every dataset except 3 out of 90, namely `farm-ads`, `mnist49`, and `gisette`. For these datasets, the limit of 10 000 reasons has been reached for, respectively, 5, 16, and 3 instances out of 1000. Interestingly, the median time needed to derive all the minimal sufficient reasons for the instances for which the computation has been successful exceeded 1s only for 2 datasets (`adult` and `bank_marketing`).

Beyond providing evidence that the number of reasons can be huge, our experiments have highlighted that the greedy algorithm for deriving a sufficient reason computes in practice a minimal sufficient reason in many cases. They have also shown that the number of explanatory relevant features for an instance is typically much lower than the number of binary features used to describe it, and that the number of explanatory necessary features is also significantly lower than the number of explanatory relevant features. The gap between the two explains the possibly enormous number of sufficient reasons. When considering the full set of reasons, a considerable difference between the number of sufficient reasons and the number of minimal sufficient reasons can also be observed. Finally, like minimal sufficient reasons, the number of contrastive explanations appears in many cases not very large, which is a good point from an intelligibility perspective.

6 Conclusion

In light of our results, it turns out that the explanatory power of decision trees goes far beyond its ability to generate direct reasons. From a decision tree, the explanatory importance of features and the minimal sufficient reasons for an instance can be computed efficiently most of the time. For decision trees, fully addressing the "Why not?" question also appears as easier than fully addressing the "Why?" question: computing the full set of sufficient reasons for the instance at hand is typically out of reach, while computing its full set of contrastive explanations is tractable.

Accordingly, the language of decision trees appears not only as appealing for the learning purpose, but also as a good target when one needs to reason on the various forms of explanations (abductive and contrastive ones) associated with the predictions made. This coheres with (and completes) the results reported in [1], showing that many other explanation and verification tasks are tractable for decision tree classifiers.

References

[1] G. Audemard, F. Koriche, and P. Marquis. On tractable XAI queries based on compiled representations. In *Proc. of KR'20*, pages 838–849, 2020.

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

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[4] L. Breiman and N. Shang. Born again trees. Technical report, https://www.stat.berkeley.edu/ breiman/BAtrees.pdf, 1996.

[5] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proc. of KDD'16*, page 785–794, 2016.

[6] A. Darwiche. Decomposable negation normal form. *Journal of the Association for Computing Machinery*, 48(4):608–647, 2001.

[7] A. Darwiche and A. Hirth. On the reasons behind decisions. In *Proc. of ECAI'20*, pages 712–720, 2020.

[8] A. Darwiche and P. Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[9] Th. Eiter and G. Gottlob. The complexity of logic-based abduction. *Journal of the Association for Computing Machinery*, 42(1):3–42, 1995.

[10] N. Frosst and G. E. Hinton. Distilling a neural network into a soft decision tree. In *Proc. of the First International Workshop on Comprehensibility and Explanation in AI and ML*, volume 2071 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.

[11] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):93:1–93:42, 2019.

[12] S. Hooker, D. Erhan, P-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks. In *Proc. of NeurIPS'19*, pages 9737–9748, 2019.

[13] J. Huysmans, K. Dejaeger, C. Mues, J. Vanthienen, and B. Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decis. Support Syst.*, 51(1):141–154, 2011.

[14] A. Ignatiev, A. Morgado, and J. Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *Proc. of SAT'18*, pages 428–437, 2018.

[15] A. Ignatiev, N. Narodytska, N. Asher, and J. Marques-Silva. On relating 'why?' and 'why not?' explanations. *CoRR*, abs/2012.11067, 2020.

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

[17] Y. Izza, A. Ignatiev, and J. Marques-Silva. On explaining decision trees. *CoRR*, abs/2010.11034, 2020.

[18] S. Jabbour, J. Marques-Silva, L. Sais, and Y. Salhi. Enumerating prime implicants of propositional formulae in conjunctive normal form. In *Proc. of JELIA'14*, pages 152–165, 2014.

[19] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *Proc. of ICML'18*, pages 2668–2677, 2018.

[20] J.-M. Lagniez and P. Marquis. An Improved Decision-DNNF Compiler. In *Proc. of IJCAI'17*, pages 667–673, 2017.

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

[22] S. Lundberg and S-I. Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Proc. of NIPS'17*, pages 4765–4774, 2017.

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

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

[25] 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.

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

[27] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[28] M. Ribeiro, S. Singh, and C. Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proc. of KDD'16*, pages 97–101, 2016.

[29] A. Shih, A. Choi, and A. Darwiche. A symbolic approach to explaining Bayesian network classifiers. In *Proc. of IJCAI'18*, pages 5103–5111, 2018.

[30] A. Shih, A. Darwiche, and A. Choi. Verifying binarized neural networks by Angluin-style learning. In *Proc. of SAT'19*, pages 354–370, 2019.

[31] G.S. Tseitin. *On the complexity of derivation in propositional calculus*, chapter Structures in Constructive Mathematics and Mathematical Logic, pages 115–125. Steklov Mathematical Institute, 1968.

[32] F. Wotawa. A variant of Reiter's hitting-set algorithm. *Inf. Process. Lett.*, 79(1):45–51, 2001.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] One cannot expect any negative impact (the paper is about explaining predictions).
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] As a supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] But the results we obtained have been averaged over a number of trials.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The pieces of software we used are furnished as a supplementary material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] This issue is irrelevant for this paper.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The datasets we used are anonymized and do not contain personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [No] We did not use crowdsourcing or conducted research with human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [No] We did not use crowdsourcing or conducted research with human subjects.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [No] We did not use crowdsourcing or conducted research with human subjects.