On the Computation of Contrastive Explanations for Boosted Regression Trees

Gilles Audemard[a,*], **Jean-Marie Lagniez**[a,**] **and Pierre Marquis**[a,b,***]

[a]Univ. Artois, CNRS, CRIL, Lens, France
[b]Institut Universitaire de France, France

Abstract. A contrastive explanation is a local explanation that is looked for when the prediction achieved by an ML model on an input instance $\boldsymbol{x}$ differs from what was foreseen. A contrastive explanation indicates how to change $\boldsymbol{x}$ to another instance $\boldsymbol{x}_c$ from which a prediction that complies with the user's expectations can be obtained. In this paper, we present a constraint-based approach to the generation of contrastive explanations that are suited to regression functions represented by boosted trees. We show how to compute the smallest interval containing all the regression values that are attainable given a set of characteristics of $\boldsymbol{x}$ that are protected (i.e., not amenable to change). We also show how to generate minimal contrastive explanations for $\boldsymbol{x}$ given a target interval, i.e., instances with regression values within the specified interval and that are as close as possible to $\boldsymbol{x}$. Closeness is captured using user-dependent mappings reflecting preferences about value change for the attributes (or combinations of attributes) considered in the representation of $\boldsymbol{x}$.

1 Introduction

Machine Learning (ML) models became pervasive in real-world applications because of their astonishing predictive power. However, the most efficient ML models are opaque. This opacity constitutes a significant obstacle for leveraging ML models in high-stake applications. eXplainable AI (XAI) has emerged for the past five years as a new research field, with the goal of developing techniques for gaining trust in ML models and the predictions obtained from them (see for instance [12, 1, 17, 13, 18, 25, 2, 8, 22]).

In the following, we present an XAI framework for *boosted trees*. When dealing with tabular data, boosted trees are among the state-of-the art ML models in terms of predictive power [7]. However, the generation of explanations in the case of boosted trees is challenging. Indeed, boosted trees can hardly be seen as explainable by design [19], or even computationally intelligible [3].

Our very objective is to show how to compute *contrastive explanations* suited to *regression functions* represented by boosted trees. A regression function is a mapping from a set $\boldsymbol{X}$ of instances to $\mathbb{R}$. The regression value $f(\boldsymbol{x})$ of an instance $\boldsymbol{x}$ may represent any piece of information that can be encoded as a real value. Most of the time, the precise value $f(\boldsymbol{x})$ taken by f on instance $\boldsymbol{x}$ is irrelevant for the user, so that predicting $f(\boldsymbol{x}) \pm \epsilon$ (for a sufficiently small ϵ) instead of $f(\boldsymbol{x})$ would not make any difference. The extent to which getting another prediction than $f(\boldsymbol{x})$ would matter can be made precise by specifying an interval $I_{\boldsymbol{x}}$ such that $f(\boldsymbol{x}) \in I_{\boldsymbol{x}}$. Any value within $I_{\boldsymbol{x}}$ would be satisfying for the user as the regression value of $\boldsymbol{x}$ for f.

Example 1. *Consider the following loan application scenario, to be used as a running example throughout the paper. Suppose that $\boldsymbol{x}$ represents a bank client, say Bob, applying for a loan in the objective of buying the boat of his dreams, that costs $\$110k$. The bank uses an ML model f to predict the amount of money that can be safely granted its clients. Suppose that the corresponding regression value $f(\boldsymbol{x}) = \$100k$ represents the amount of money that the bank is ready to grant Bob. If Bob has savings up to $\$20k$ that he is okay to use for buying the boat, he will surely be happy with the bank offer since he will have enough money to get the boat. Thus, in Bob's case, $I_{\boldsymbol{x}} = [90k, +\infty)$ would do the job. However, it could be the case that $f(\boldsymbol{x}) = \$80k$. In this situation, we do not have $f(\boldsymbol{x}) \in I_{\boldsymbol{x}}$, and Bob would like to know what he could do to be granted a loan of at least $\$90k$.*

The quest for contrastive explanations precisely aims to find answers to such questions. Bob would like to know how to change his description (given by the input instance $\boldsymbol{x}$) to get a prediction belonging to a preset target interval $I_{\boldsymbol{x}}$.

In general, $I_{\boldsymbol{x}}$ is such that $f(\boldsymbol{x}) \notin I_{\boldsymbol{x}}$ since in the case when $f(\boldsymbol{x}) \in I_{\boldsymbol{x}}$, the user does not need a contrastive explanation. Most of the time, the expected change must be minimal in a certain sense, and the corresponding notion of minimality is *user-dependent*. Informally, it reflects the minimal effort the user is ready to make (in terms of the characteristics to be changed) in order to reach a regression value within an expected interval.

Several criteria can be used for evaluating explanations (and/or the XAI methods used to produce them) [20, 26]. Correctness (aka faithfulness or soundness) indicates to which extent explanations capture the actual behaviour of the ML model under consideration. It is paramount since the goal pursued is to account for the predictions made by the model used, not those that could be obtained by a different model. Generalizability is also valuable. It reflects the number of instances covered by the explanation that is produced: the larger this number the more general the explanation.

When dealing with tree-based models F, every instance $\boldsymbol{x} \in \boldsymbol{X}$ can be associated with a unique instance, noted $\boldsymbol{x}^F$, that is based on the set of (usually non-independent) Boolean conditions $\mathcal{B}$ occurring in F. $\boldsymbol{x}^F$ can be obtained by rewriting the instance $\boldsymbol{x}$ considered primarily, the description of which being based on attributes that are not solely Boolean (numerical or categorical attributes are usually in-

* Email: audemard@cril.fr.
** Email: lagniez@cril.fr.
*** Corresponding Author. Email: marquis@cril.fr.

volved at start). The rewrite process can be achieved efficiently (i.e., using a linear-time algorithm in the size of the input instance and the size of F), and it ensures that $f(\boldsymbol{x}^F) = f(\boldsymbol{x})$, where f is the regression function represented by F.

Example 1 [cont'ed] Considering the loan application scenario again, suppose that the three attributes used to describe instances from $\boldsymbol{X}$ are the annual income A_1 of the applicant (in $\$k$), his/her age A_2, and his/her level of education A_3. Then, depending on the values of A_1, A_2, and A_3 that are selected by the algorithm used to learn the boosted tree, F may contain the following Boolean conditions $(A_1 > 80)$, $(A_1 > 65)$, $(A_1 > 30)$, $(A_2 > 50)$, $(A_2 > 30)$, $(A_3 = bachelor)$, $(A_3 = master)$, $(A_3 = Ph.D.)$ (taken in this order and forming the set $\mathcal{B}$). The instance $\boldsymbol{x} = (A_1 = 70, A_2 = 29, A_3 = master)$ corresponding to Bob is associated with the instance $\boldsymbol{x}^F = (0, 1, 1, 0, 0, 0, 1, 0)$ over $\mathcal{B}$ since $\boldsymbol{x}$ satisfies $(A_1 > 65)$, $(A_1 > 30)$, and $(A_3 = master)$ while it does not satisfy any other conditions from $\mathcal{B}$. Of course, $\boldsymbol{x}^F$ depends on F, and more precisely on the Boolean conditions (especially, the thresholds used in them) that have been considered when F has been learned. Thus, if the thresholds occurring in F about A_2 were 30 and 25 (instead of 50 and 30), the instance corresponding to Bob would have been $\boldsymbol{x}^F = (0, 1, 1, 0, 1, 0, 1, 0)$.

Whatever the thresholds retained, the Boolean conditions in $\mathcal{B}$ are not independent one another, but are related by a domain theory Σ that indicates how the conditions are logically connected. $\boldsymbol{X}^F$ denotes the set of all Boolean instances $\boldsymbol{x}^F$ over $\mathcal{B}$ that satisfy Σ.

Example 1 [cont'ed] For the loan application scenario, the domain theory Σ associated with $\mathcal{B}$ is equivalent to

$$
\begin{array}{ll}
 & ((A_1 > 80) \Rightarrow (A_1 > 65)) \\
\wedge & ((A_1 > 65) \Rightarrow (A_1 > 30)) \\
\wedge & ((A_2 > 50) \Rightarrow (A_2 > 30)) \\
\wedge & ((A_3 = bachelor) \Rightarrow \neg(A_3 = master)) \\
\wedge & ((A_3 = bachelor) \Rightarrow \neg(A_3 = Ph.D.)) \\
\wedge & ((A_3 = master) \Rightarrow \neg(A_3 = Ph.D.)).
\end{array}
$$

As shown in [5], considering rewritten instances $\boldsymbol{x}^F$ is valuable to derive explanations that are *at least as general* as explanations that could be obtained from the instances $\boldsymbol{x}$ considered at start.[1] Contrastive explanations given as instances from $\boldsymbol{X}^F$ (and not as instances from $\boldsymbol{X}$) better reflect the behaviour of the tree ensemble that is used to achieve the regression task. The improved generality of instances from $\boldsymbol{X}^F$ also reduces the risk to report contrastive instances that are outliers. Finally, when all the attributes used are Boolean ones, there is no need to indicate the *values* of the attributes that must be changed to get a contrastive explanation for $\boldsymbol{x}^F$: it is enough to indicate the attributes the values of which must be changed. As a consequence, the contrastive explanations generated by **C** can be represented by terms t, i.e., conjunctions of literals, over the Boolean conditions in $\mathcal{B}$.

Example 1 [cont'ed] For example, if the regression value $f(\boldsymbol{x}^F)$ of f for $\boldsymbol{x}^F = (0, 1, 1, 0, 0, 0, 1, 0)$ (associated with Bob) is less than $90k$, Bob is not involved in the preparation of a Ph.D. and is not ready to wait (i.e., to increase his age), and $\boldsymbol{x}_c^F = (1, 1, 1, 0, 0, 0, 1, 0)$ satisfies $f(\boldsymbol{x}_c^F) \geq 90k$, then the contrastive instance $\boldsymbol{x}_c^F$ for $\boldsymbol{x}^F$ given f and $I_{\boldsymbol{x}^F} = [90k, +\infty)$ can be represented by the term $t = (A_1 > 80)$. Using words, to get a loan of at least $\$90k$, it is enough for Bob to increase his annual income to more than $\$80k$. Notably, this minimal contrastive explanation applies to Bob, but also to any other applicant like Charlie (represented at start by $\boldsymbol{x}' = (A_1 = 70, A_2 = 25, A_3 = master)$ and having the same desiderata as Bob concerning the way his characteristics may change. Indeed, we have $\boldsymbol{x}'^F = \boldsymbol{x}^F$.

The contribution of this paper mainly consists of two constraint encodings, **I** and **C**. Slightly abusing words, **I** and **C** are also referred to as "algorithms", even if the algorithm in both cases is the one of the contraint solver used, with **I** or **C** as input. Given F and $\boldsymbol{x}$, **I** can be leveraged to *generate target intervals* I_t depending on the subset t of the characteristics of $\boldsymbol{x}^F$ the user would like to preserve (i.e., not to change). **I** can be used to help the user envision an interval $I_{\boldsymbol{x}}$ that is as small as possible and containing all the regression values that are attainable given the characteristics (or combinations of characteristics) the user is (more or less) ready to change and those he/she is reluctant to modify. **C** can be leveraged to *generate minimal contrastive explanations* for $\boldsymbol{x}$ given F and $I_{\boldsymbol{x}}$. For the sake of generality, minimal contrastive explanations from $\boldsymbol{X}^F$ are looked for. A dissimilarity mapping is used by **C** to characterize the notion of minimal change at work. This mapping is user-dependent. This is fundamental since another client of the bank characterized by the same instance $\boldsymbol{x}$ as Bob could be less prone than Bob to change some of his/her characteristics. We show how sophisticated preferences about possible changes of the characteristics (or combinations of characteristics) of $\boldsymbol{x}$ can be represented and taken into account by **C**. Experiments have been made showing that **C** is efficient enough for computing minimal contrastive explanations in many cases. The code used and the datasets considered in the experiments are furnished as a supplementary material, available online [6].

2 Preliminaries

We consider a set of instances $\boldsymbol{X}$ described using a finite set $\mathcal{A} = \{A_1, \ldots, A_n\}$ of *attributes* (aka *features*) where each attribute A_i ($i \in [n]$) takes its value in a domain D_i. Each attribute A_i can be numerical, categorical, or Boolean. Thus, $\mathcal{A}$ is the union of three pairwise disjoint subsets $\mathcal{A}_N$, $\mathcal{A}_C$, $\mathcal{A}_B$ containing respectively the numerical, categorical, and Boolean attributes. An *instance* $\boldsymbol{x} \in \boldsymbol{X}$ is a vector $(v_1, \ldots, v_n)$ where each v_i ($i \in [n]$) is an element of D_i. Each pair $A_i = v_i$ is called a *characteristic* of the instance $\boldsymbol{x}$. $\boldsymbol{x}[A_i]$ denotes the value v_i of the coordinate corresponding to A_i in $\boldsymbol{x}$.

A *regression tree* over $\mathcal{A}$ is a binary tree T, each of its internal nodes being labeled with a Boolean condition on an attribute from $\mathcal{A}$, and leaves are labeled by real numbers. The conditions are of the form $A_i > v_j^i$ with v_j^i a number when A_i is a numerical attribute, $A_i = v_j^i$ when A_i is a categorical attribute, and A_i (or equivalently $A_i = 1$) when A_i is a Boolean attribute. The *value* $T(\boldsymbol{x}) \in \mathbb{R}$ of T for an input instance $\boldsymbol{x} \in \boldsymbol{X}$ is given by the real number labelling the leaf node reached from the root as follows: at each internal node, go to the left when the condition labelling the node is not satisfied by $\boldsymbol{x}$, and go to the right otherwise.

A *boosted regression tree* over $\mathcal{A}$ is an ensemble of trees (alias a forest) $F = \{T_1, \cdots, T_m\}$, where each T_i ($i \in [m]$) is a regression tree over $\mathcal{A}$, and such that the value $F(\boldsymbol{x}) \in \mathbb{R}$ of F for an input instance $\boldsymbol{x} \in \boldsymbol{X}$ is given by $F(\boldsymbol{x}) = \oplus_{i=1}^m T_i(\boldsymbol{x})$. In the following, $\oplus$ is the *sum* operator but other operators strictly monotonic in each argument (e.g., the mean) could be considered instead.

The *running domain* RD_i of $A_i \in \mathcal{A}$ in F is the subset of D_i containing all the values v_i^j that are encountered in the labels of the internal nodes of the trees of F that are about A_i. When A_i is a categorical

[1] In the same direction (i.e., promoting the generality of explanations), see also the recently introduced concept of inflated explanation, that is not specific to tree ensembles [15].

attribute with running domain $RD_i = \{v_i^1, \ldots, v_i^{d(i)}\}$, A_i is associated with the set of Boolean variables $B(A_i) = \{X_i^1, \ldots, X_i^{d(i)}\}$ where for each $j \in [d(i)]$, X_i^j stands for the condition $(A_i = v_i^j)$. When A_i is a numerical attribute with running domain $RD_i = \{v_i^1, \ldots, v_i^{d(i)}\}$ where $v_i^1 > \ldots > v_i^{d(i)}$, A_i is associated with the set of Boolean variables $B(A_i) = \{X_i^1, \ldots, X_i^{d(i)}\}$ where for each $j \in [d(i)]$, X_i^j stands for the condition $(A_i > v_i^j)$. When A_i is a Boolean attribute A_i is associated with the set of Boolean variables $B(A_i) = \{X_i\}$. $\mathcal{B}$ is defined by $\bigcup_{A_i \in \mathcal{A}} B(A_i)$. $\mathcal{B}$ is supposed to be totally ordered in an arbitrary (yet fixed) way, so that every element $X_j \in \mathcal{B}$ has an index j varying from 1 to p, where p denotes the cardinality of $\mathcal{B}$. Every $\boldsymbol{x} \in \boldsymbol{X}$ can be associated with an instance $\boldsymbol{x}^F = (v_1, \ldots, v_p)$ from $\boldsymbol{X}^F$ such that $v_j = 1$ ($j \in [p]$) if and only if the Boolean condition X_j is satisfied by $\boldsymbol{x}$.

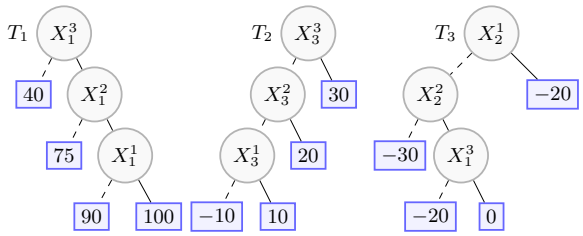

Figure 1: A boosted regression tree based on three trees T_1, T_2, T_3.

Example 1 [cont'ed] Let us consider the loan application scenario presented in the introduction. We suppose that the boosted regression tree $F = \{T_1, T_2, T_3\}$ depicted in Figure 1 has been learned and that f is the corresponding regression function. We have $RD_1 = \{80, 65, 30\}$, $RD_2 = \{50, 30\}$, $RD_3 = \{bachelor, master, Ph.D.\}$. The set $\mathcal{B}$ gathering the Boolean conditions used in F contains 8 elements: $X_1^1 = (A_1 > 80)$, $X_1^2 = (A_1 > 65)$, $X_1^3 = (A_1 > 30)$, $X_2^1 = (A_2 > 50)$, $X_2^2 = (A_2 > 30)$, $X_3^1 = (A_3 = bachelor)$, $X_3^2 = (A_3 = master)$, $X_3^3 = (A_3 = Ph.D.)$. The instance $\boldsymbol{x} = (A_1 = 70, A_2 = 29, A_3 = master)$ corresponding to Bob is associated with the instance $\boldsymbol{x}^F = (0, 1, 1, 0, 0, 0, 1, 0)$ over the Boolean conditions occurring in F. The regression value of $\boldsymbol{x}$ (or equivalently of $\boldsymbol{x}^F$) is equal to $90 + 20 - 30 = 80$.

A term t over $\mathcal{B}$ is a conjunctively-interpreted set of literals over $\mathcal{B}$. $\top$ denotes the term associated with the empty set of literals. t is *canonical* when it contains one literal per Boolean variable in $\mathcal{B}$. Such a term thus corresponds to a truth assignment over $\mathcal{B}$. Every instance $\boldsymbol{x}^F \in \boldsymbol{X}^F$ can be associated with a canonical term $t_{\boldsymbol{x}^F}$ over $\mathcal{B}$ such that for every $X \in \mathcal{B}$, X (resp. $\overline{X}$) belongs to $t_{\boldsymbol{x}^F}$ if and only if $\boldsymbol{x}^F$ satisfies (resp. does not satisfy) the Boolean condition X. A term t over $\mathcal{B}$ *covers* an instance $\boldsymbol{x}^F \in \boldsymbol{X}^F$ whenever $t \subseteq t_{\boldsymbol{x}}$. Given an attribute $A_i \in \mathcal{A}$ and an instance $\boldsymbol{x}^F \in \boldsymbol{X}^F$, $\boldsymbol{x}^F[A_i]$ denotes the vector of dimension $d(i)$ (the cardinality of the running domain $RD_i = \{v_i^1, \ldots, v_i^{d(i)}\}$ of A_i) containing precisely all the Boolean values associated with the attributes from $\mathcal{B}$ that represent the conditions of the form $(A_i > v_i^j)$ (when A_i is numerical) or $(A_i = v_i^j)$ (when A_i is categorical or Boolean), where $v_i^j \in RD_i$.

3 Computing contrastive explanations

Our two encodings **I** and **C** rely on a set $\mathcal{M}$ of Mixed-Integer Linear Programming (MILP) constraints that specifies how the values of the Boolean attributes $\mathcal{B}$ occurring in the forest F representing the regression function f are connected to the regression values. In the following, we first present this set $\mathcal{M}$ of hard constraints. Then we successively show how to take advantage of it to generate regression intervals (this is algorithm **I**) and to compute contrastive explanations for an input instance given F and a targeted regression interval (this is algorithm **C**) .

Encoding a boosted regression tree using MILP constraints All the hard contraints in $\mathcal{M}$ must be satisfied by any eligible solution. They encode the boosted tree F under consideration and the domain theory Σ that indicates how the conditions from $\mathcal{B}$ are logically connected. The variables used in $\mathcal{M}$ are either 0/1 variables - representing, among others, the conditions X_i^j from $\mathcal{B}$ labelling the internal nodes in the trees of F - or continuous variables - representing the real values at the leaves and the regression values. Notably, $\mathcal{M}$ encodes F and Σ exactly, i.e., $\mathcal{M}$ is not a surrogate model. As a consequence, the results provided by our algorithms **I** and **C** are *guaranteed to be faithful to* F. This contrasts with approaches where proxys are used instead of the ML model under consideration.

Each tree T_i of $F = \{T_1, \ldots, T_m\}$ is represented by a set of terms $\{t_1^i, \ldots, t_{p(i)}^i\}$, where each term gathers the literals representing the conditions over $\mathcal{B}$ that are true in a root-to-leaf path of T_i. Each t_j^i ($j \in [p(i)]$) characterizes a unique path of T_i and w_j^i is the real number labelling the leaf of the j^{th} path of T_i. With each tree T_i ($i \in [m]$) one associates a set of 0/1 variables $\mathcal{L}_i = \{L_{t_1^i}^i, \ldots, L_{t_{p_i}^i}^i\}$. Variable $L_{t_j^i}^i$ ($j \in [p(i)]$) is set to 1 when the conditions given by t_j^i are met. For each $i \in [m]$ and each $t_j^i \in T_i$, the following MILP constraint is put into $\mathcal{M}$:

$$\sum_{X \in t_j^i} X + \sum_{\overline{X} \in t_j^i} (1 - X) - L_{t_j^i}^i \leq |t_j^i| - 1$$

When T_i is a decision tree, the terms in $\{t_1^i, \ldots, t_{p(i)}^i\}$ are pairwise inconsistent. This implies that exactly one $L_{t_j^i}^i$ must be set to 1. This is ensured by adding to $\mathcal{M}$ for each tree T_i ($i \in [m]$) the following MILP constraint:

$$\sum_{t_j^i \in T_i} L_{t_j^i}^i = 1$$

One also needs to consider a set of continuous variables $\mathcal{W} = \{W_1, \ldots, W_m\}$ and constraints associating with each tree T_i ($i \in [m]$) the real number w_j^i that corresponds to the value of T_i for the selected root-to-leaf path. Each variable W_i ($i \in [m]$) can be easily defined by the following MILP constraint:

$$\sum_{j \in [p_i]} w_j^i \times L_{t_j^i}^i = W_i$$

The next step consists in defining another continuous variable FW representing the value of the regression tree for any truth assignment over $\mathcal{B}$:

$$\sum_{W_i \in \mathcal{W}} W_i = FW$$

We finally consider constraints representing the domain theory Σ associated with the attributes A_i (and the values in their their running domains $\{v_i^1, \ldots, v_i^{d(i)}\}$), that are used in F. A conjunction of constraints is added to $\mathcal{M}$ for each attribute A_i used in F whenever A_i is numerical or categorical. Thus, for every numerical attribute $A_i \in \mathcal{A}$ such that the conditions $A_i > v_i^1$, ..., $A_i > v_i^{d(i)}$ occur

in F where $\forall j \in [d(i)-1]$, we have $v_i^j > v_i^{j+1}$, the constraint to be considered is $\bigwedge_{j=1}^{d(i)-1} X_i^j - X_i^{j+1} \geq 0$. For every categorical attribute $A_i \in \mathcal{A}$ such that the (pairwise distinct) conditions $A_i = v_i^1$, ..., $A_i = v_i^{d(i)}$ occur in F, the constraint to be considered is $\bigwedge_{j=1}^{d(i)} \bigwedge_{k=j+1}^{d(i)} X_i^j + X_i^k \leq 1$ if the running domain RD_i of A_i is supposed to be open (i.e., we do not assume that $RD_i = D_i$), and the constraint is $\sum_{j=1}^{d(i)} x_i^j = 1$ if the running domain RD_i of A_i is considered as closed (i.e., RD_i is supposed to be equal to D_i).

Generating regression intervals Given a boosted regression tree F over $\mathcal{A}$ representing a regression function f and a term t over $\mathcal{B}$, let us define the smallest regression interval I_t containing all the regression values $f(\boldsymbol{x})$ that are attainable by the instances $\boldsymbol{x}$ covered by t:

Definition 1. *Let F be a boosted regression tree over $\mathcal{A}$ representing a regression function f and t a term over $\mathcal{B}$. The* regression interval $I_t = [m_t, M_t]$ *induced by t is defined by $m_t = min(\{f(\boldsymbol{x}) : \boldsymbol{x} \in \boldsymbol{X}, t \text{ covers } \boldsymbol{x}\})$ and $M_t = max(\{f(\boldsymbol{x}) : \boldsymbol{x} \in \boldsymbol{X}, t \text{ covers } \boldsymbol{x}\})$.*

Example 1 [cont'ed] Considering our running example again, the interval $I_\top$ includes all the amounts of money the bank is ready to grant a client who wants to buy a boat, independently of his/her characteristics (i.e., when no characteristics of the client are required to be kept : $t = \top$). Knowing $I_\top$ may prove enough for Bob to make an informed decision: if $I_\top \cap [90k, +\infty) = \varnothing$, Bob knows that he will never get enough money from the bank to reach his goal of buying the boat. Given the regression function represented by the boosted regression tree presented in Example 1, $I_\top = [0, 130k]$. The smallest value $m_\top = 0$ is associated with any applicant having income $A_1 \leq \$30k$, an age $A_2 \leq 30$, and a level of education A_3 less than a bachelor. The largest value $M_\top = 130k$ is associated with any applicant having income $A_1 > \$80k$, an age A_2 between 31 and 50, and holding a Ph.D.

Depending on the characteristics Bob cannot change (e.g., diminishing his age) and those he would not like to change (e.g., waiting for too long, or changing his level of education), Bob can take advantage of successive invocations of **I** (with distinct t reflecting a graduation in Bob's preferences) to figure out what is possible and make up his mind about a target interval $I_{\boldsymbol{x}}$ that would be convenient for him. For instance, consider the term $t = \overline{X_2^3} \wedge X_3^2$ specifying that Bob is not ready to wait (so, to get older), and that he is not ready to change his level of education. We have $I_t = [30k, 90k]$. If in addition, Bob indicates that he does not plan to lower his income, then the term to be considered is $t' = X_1^2 \wedge \overline{X_2^3} \wedge X_3^2$ and $I_{t'} = [80k, 90k]$. Finally, if Bob only indicates that he is not ready to change his level of education, then the term to be considered is $t'' = X_3^2$ and $I_{t''} = [30k, 120k]$.

The algorithm **I** to be used for generating the regression interval induced by a given term t is any MILP solver used twice on the program obtained by adding to $\mathcal{M}$ the following hard constraints encoding the term t used to make precise the characteristics of $t_{\boldsymbol{x}^F}$ that are required not to change:

$$\forall X \in t, X = 1$$

$$\forall \overline{X} \in t, X = 0$$

m_t is obtained by minimizing the objective function that reduces to FW (or equivalently, maximizing the objective function $-FW$), while M_t is obtained by maximizing the objective function FW.

Notably, we can ensure that taking advantage of a MILP solver to compute I_t given F and t is not using a sledgehammer to crack a nut. To make it more formal, it is easy to show that there is no polynomial-time algorithm for computing I_t given F and t unless P = NP:

Proposition 1. *Computing I_t given F and t is* NP*-hard.*

Proof. We show that computing $I_\top = [m_\top, M_\top]$ is already NP-hard. First, it is easy to show that every CNF formula $\alpha = \bigwedge_{i=1}^{m} \delta_i$ over variables $X_1, \ldots, X_n$ can be turned in linear time into a boosted regression tree $F = \{T_1, \ldots, T_m\}$ over $X_1, \ldots, X_n$. Indeed, each clause δ_i ($i \in [m]$) (supposed wlog to be non-tautologous) can be represented by an equivalent comb-shaped (decision) tree T_i, i.e., the leaves of T_i are labelled by 1, except for the leaf ending the unique path of T_i corresponding to $\neg\delta_i$, which is labelled by 0. More formally, T_i can be defined by induction on the length of δ_i. The base case is when δ_i is the empty clause. In this case, T_i reduces to a single leaf node labelled by 0. For the inductive case, suppose that $\delta_i = \ell \vee \delta_i'$ where X is the variable of literal ℓ. Then the root of T_i is a decision node labelled by X, and there are two cases: if $\ell = X$, the left child of T_i is a decision tree corresponding to T_i' and the right child of T_i is a leaf node labelled by 1; if $\ell = \neg X$, the left child of T_i is a leaf node labelled by 1 and the right child of T_i is a decision tree corresponding to T_i'. By construction, for every truth assignment $\boldsymbol{x}$ over $X_1, \ldots, X_n$, $F(\boldsymbol{x})$ is the number of clauses of α that are satisfied by $\boldsymbol{x}$. Hence, α is satisfiable if and only if there exists an instance $\boldsymbol{x}$ over $X_1, \ldots, X_n$ such that $F(\boldsymbol{x}) = m$, which is the case precisely when $M_\top = m$. □

Generating contrastive explanations When the user represented by $\boldsymbol{x}$ has made up his/her mind about the target regression interval $I_{\boldsymbol{x}} = [min, Max]$ he/she may expect (depending on what he/she is ready to give up), he/she may look for contrastive explanations for $\boldsymbol{x}$ given f and $I_{\boldsymbol{x}}$, and specifically to minimal contrastive explanations for $\boldsymbol{x}$ given f and $I_{\boldsymbol{x}}$, i.e., those contrastive explanations requiring a minimal change.

The concept of minimal change is user-dependent, and when dealing with numerical attributes A_i, it can easily be the case that the thresholds that matter for the user under consideration are not those appearing about A_i in the boosted tree F representing f. A set $U_i \subseteq D_i$ of $u(i)$ values $u_i^{u(i)}, \ldots, u_i^1$ for A_i that may not belong to RD_i must thus be taken into account. Among them $u_i^{u(i)}$ is the smallest value that can be envisioned by the user for A_i (it can be equal to 0, to $-\infty$, or to any value lower than every element of RD_i).

Each value u_i^j must be added to RD_i when it does not belong to it, and for each of them a new Boolean variable $X_i^{s(j)}$representing $(A_i > u_i^j)$ must be added to $\mathcal{B}$ and connected to the other Boolean variables about A_i in the domain theory. $s(j)$ is the index shift to be applied to j in order to reflect that fact that the values in $RD_i \cup U_i$ are still ordered in a decreasing way.

Example 1 [cont'ed] Suppose that Bob is ready to wait for up to five years, but no more, before being granted the loan. Thus, from the boat acquisition perspective, Bob views his current age 29 and 34 as indifferent, but since he does not want to wait more than 5 years, he would consider a change from 29 to 35 (or more) as strictly less preferred than a change from 29 to any value between 29 and 34. Since Bob is currently 29 years old, the value $29 + 5 = 34$ must be taken into account. The set $U_2 = \{34, 0\}$ is considered, $u_2^2 = 0$ being the least possible value for an age (one could consider instead the age of majority if required to take out a loan). These two values

are added to the running domain RD_2 of A_2, that is extended from $\{50, 30\}$ to $\{50, 34, 30, 0\}$. Two Boolean variables X_2^2 representing $(A_2 > 34)$ and X_2^4 representing $(A_2 > 0)$ are considered. Once this extension has been done, $\mathcal{B}$ consists of X_2^1 representing $(A_2 > 50)$, X_2^2 representing $(A_2 > 34)$, X_2^3 representing $(A_2 > 30)$, and X_2^4 representing $(A_2 > 0)$. The corresponding domain theory Σ must be updated accordingly, so as to reflect that $(X_2^1 \Rightarrow X_2^2) \wedge (X_2^2 \Rightarrow X_2^3) \wedge (X_2^3 \Rightarrow X_2^4)$ holds.

Of course, $\boldsymbol{X}^F$ must also be modified in order to take into account the new Boolean variables that have been introduced into $\mathcal{B}$.

When f is represented by a boosted tree F, the following definition can be considered:

Definition 2. *Let F be a boosted tree over a set $\mathcal{A}$ of attributes. Let Σ be the domain theory that connects all the Boolean conditions from $\mathcal{B}$, i.e., those used in F and the Boolean conditions $(A_i > u_i^j)$ corresponding to the additional values $v_i^j \in U_i$ that have been introduced by the user for the numerical attribute A_i. Let $\boldsymbol{x}$ be an instance from $\boldsymbol{X}$. Let $I_{\boldsymbol{x}}$ be an interval of $\mathbb{R}$.*

- *A* contrastive explanation *for $\boldsymbol{x}$ is an instance $\boldsymbol{x}_c^F$ from $\boldsymbol{X}^F$ such that $F(\boldsymbol{x}_c^F) \in I_{\boldsymbol{x}}$.*
- *Given an strict ordering $<_{\boldsymbol{x}}$ over $\boldsymbol{X}^F$, a* minimal contrastive explanation *for $\boldsymbol{x}^F$ is a contrastive explanation for $\boldsymbol{x}$ that is minimal w.r.t. $<_{\boldsymbol{x}}$ (i.e., it is as close as possible to $\boldsymbol{x}$ w.r.t. $<_{\boldsymbol{x}}$).*

The algorithm **C** to be used for generating a minimal contrastive explanation for $\boldsymbol{x}$ given F and $I_{\boldsymbol{x}}$ is any MILP solver used on the program obtained by adding first to $\mathcal{M}$ the following hard constraints indicating the expected range of regression values:

$$min \leq FW$$

$$FW \leq Max$$

In order to be used to compute minimal contrastive explanations, $\mathcal{M}$ must be completed to account for the user preferences that characterize the underlying notion of minimality. In our work, $<_{\boldsymbol{x}}$ is represented by a dissimilarity mapping $dis : \boldsymbol{X}^F \times \boldsymbol{X}^F \rightarrow \mathbb{R}^+ \cup \{+\infty\}$ so that $\boldsymbol{x}_c^F <_{\boldsymbol{x}} \boldsymbol{x}'^F$ holds if and only if $dis(\boldsymbol{x}^F, \boldsymbol{x}_c^F) < dis(\boldsymbol{x}^F, \boldsymbol{x}'^F)$. We suppose that the user preferences can be expressed in an additive way by the weighted sum of "local" dissimilarity mappings bearing on parts of $\boldsymbol{x}^F$ which are about attributes or, more generally, about combinations of attributes. The value returned by any "local" dissimilarity mapping is an element of $\mathbb{R}^+ \cup \{+\infty\}$ indicating the cost of the effort to be supported by the user to change the value of the part of $\boldsymbol{x}^F$ it is about, depending on the value it reaches.

Before considering combinations of attributes, let us start with the base case consisting of attributes alone. For each attribute A_i, the user may express his/her own preferences over any change of the current value of A_i in $\boldsymbol{x}$. Some preferences can be compensated (in that case, the dissimilarity score is a real number), others cannot be compensated (then the dissimilarity score is $+\infty$). Indeed, because some value changes are not feasible (e.g., getting younger), it is important to have a way to guarantee that some value updates are impossible and the dissimilarity score $+\infty$ is used to this end.

When A_i is a categorical attribute with an open running domain RD_i, one considers an additional variable $X_i^{d(i)+1}$ encoding the Boolean condition $(A_i = v_i^{d(i)+1})$, where $v_i^{d(i)+1} = none$ stands for any value of A_i that does not belong to RD_i. One replaces the constraint $\bigwedge_{j=1}^{d(i)} \bigwedge_{k=j+1}^{d(i)} X_i^j + X_i^k \leq 1$ of $\mathcal{M}$ by the constraint $\sum_{j=1}^{d(i)+1} X_i^j = 1$. In some sense, this is a way to close the running domain RD_i. We define RD_i^* as $RD_i \cup \{v_i^{d(i)+1}\}$ if RD_i is open, and as RD_i otherwise. The user is then asked to state the costs $w_{v_i^j}$ (an element of $\mathbb{R}^+ \cup \{+\infty\}$) of the transitions from the current value $\boldsymbol{x}[A_i]$ of $\boldsymbol{x}$ on attribute A_i to the values v_i^j in RD_i^*. We assume that $w_{v_i^j} = 0$ when $\boldsymbol{x}[A_i] = v_i^j$ (i.e., there is no change). Whenever $w_{v_i^j} = +\infty$, the constraint $X_i^j = 0$ is added to $\mathcal{M}$.

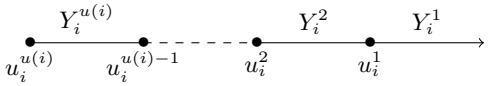

Figure 2: Splitting the domain of a numerical attribute A_i using intervals defined from U_i

Example 1 [cont'ed] Let us focus on the categorical attribute A_3 used to describe the level of education of the applicants in the loan granting scenario. The running domain of this attribute is $\{bachelor, master, Ph.D.\}$. It must be considered open since it can be the case that the level of education of an applicant does not correspond to any of those three degrees. Considering again the instance $\boldsymbol{x} = (A_1 = 70, A_2 = 29, A_3 = master)$ associated with Bob, it would make sense to have a transition cost from $A_3 = master$ to $A_3 = none$ or to $A_3 = bachelor$ set to $+\infty$ (since one cannot downgrade one's level of education), a transition cost from $A_3 = master$ to $A_3 = master$ set to 0 (since no change occurs), and finally a transition cost w_3^3 from $A_3 = master$ to $A_3 = Ph.D.$ set to a positive number, let us say $w_3^3 = 7$ (this value reflects the effort Bob has to furnish to ensure such a transition). By the way, one can observe that such a transition cost mapping is not required to be symmetric in its arguments (e.g., for another applicant than Bob, a transition cost from $A_3 = bachelor$ to $A_3 = master$ that is not $+\infty$ would be acceptable).

When A_i is a numerical attribute A_i, $\boldsymbol{x}[A_i]$ belongs to a specific interval and the user-dependent cost of shifting the value of $\boldsymbol{x}$ on A_i from this interval to another interval must be taken into account. This cost has to be specified for each of the $u(i)$ disjoint intervals induced by the $u(i)$ values in U_i reported by the user about A_i. Those values can be ordered so that $u_i^1 > \ldots > u_i^{u(i)}$, where $u_i^{u(i)}$ is the smallest value envisioned by the user for A_i. For every j from 2 to $u(i)$, we introduce a 0/1 variable Y_i^j defined by the constraint $(X_i^{s(j+1)} \geq Y_i^j) \wedge (X_i^{s(j)} + Y_i^j \leq 1) \wedge (X_i^{s(j+1)} \leq X_i^{s(j)} + Y_i^j)$ which is added to $\mathcal{M}$ (see Figure 2). Thanks to this constraint, Y_i^j takes the value 1 precisely when the current value of A_i moves to $[u_i^{j+1}, u_i^j)$. For the sake of uniformity, we also introduce a 0/1 variable Y_i^1 defined by the constraint $Y_i^1 = X_i^{s(1)}$ which is added to $\mathcal{M}$ as well. The user is then asked to indicate the cost $w_{u_i^j}$ (an element of $\mathbb{R}^+ \cup \{+\infty\}$) of the transition from the current interval containing the value $\boldsymbol{x}[A_i]$ of $\boldsymbol{x}$ on attribute A_i to the other intervals generated using the thresholds from U_i (each of them being characterized by its lower bound u_i^j in U_i). We assume that $w_{u_i^j} = 0$ when $j > 1$ and $\boldsymbol{x}[A_i] \in [u_i^j, u_i^{j-1})$, or when $j = 1$ and $\boldsymbol{x}[A_i] \in [u_i^1, +\infty)$ (i.e., when there is no interval shift). Whenever $w_{u_i^j} = +\infty$, the constraint $Y_i^j = 0$ is added to $\mathcal{M}$. This is a way to encode an actionability constraint. [2]

[2] Similarly, we could add to the hard constraints of $\mathcal{M}$ monotonicity constraints, that can be used, for instance, to represent the fact that Bob is not ready to reduce his income; to make it, a new 0/1 variable representing

Example 1 [cont'ed] Taking $u_2^2 = 0$, we get $U_2 = \{34, 0\}$, leading to split the domain D_2 of A_2 (the age of the applicant) into 2 disjoint intervals: $[0, 34)$, $[34, +\infty)$ (here, the least value u_2^2 is considered to be 0 but it could also be set to the age of majority if required to take out a loan). Two variables Y_2^2 and Y_2^1 are introduced, representing respectively the intervals $[0, 34)$ and $[34, +\infty)$, thus equivalent (respectively) to $X_2^4 \wedge \overline{X_2^2}$ and X_2^2. Bob is 29 years old, thus $A_2 = 29$ satisfies $\overline{X_2^1} \wedge \overline{X_2^2} \wedge \overline{X_2^3} \wedge X_2^4$, i.e., $(A_2 > 50) \wedge (A_2 > 34) \wedge (A_2 > 30) \wedge (A_2 > 0)$: 29 belongs to $[0, 30)$. Thus, $A_2 = 29$ satisfies also $Y_2^2 \wedge \overline{Y_2^1}$. If Bob waits for 3 years, so that his age becomes equal to $A_2 = 32$, the condition $\overline{X_2^1} \wedge \overline{X_2^2} \wedge X_2^3 \wedge X_2^4$ becomes true, but $A_2 = 32$ still satisfies $Y_2^2 \wedge \overline{Y_2^1}$. There is no interval shift, so no cost to be considered. Contrariwise, if Bob waits for 6 years, so that his age becomes equal to $A_2 = 35$, the condition $\overline{X_2^1} \wedge X_2^2 \wedge X_2^3 \wedge X_2^4$ becomes true, $Y_2^2 \wedge \overline{Y_2^1}$ no longer is satisfied since $A_2 = 35$ satisfies $\overline{Y_2^2} \wedge Y_2^1$. A transition cost $w_{u_i^1}$ associated with the interval shift (Y_2^1 becoming true) can be taken into account. For instance, $w_{u_i^1} = 10$.

Consider now the more general case when the user has preferences about expected shifts concerning *combinations of attributes* that can be represented as Boolean functions C_i ($i \in [p]$) over variables from $\mathcal{B}$. Then with each C_i ($i \in [p]$) one can associate a 0/1 variable Z_i that takes the value 1 precisely for the assignments over $\mathcal{B}$ that makes C_i true. Such en equivalence between Z_i and C_i can be represented as a conjonction of hard constraints (linear inequations or equations over 0/1 variables) that can be generated in linear time in the size of C_i using a standard encoding trick (that boils down to Tseitin transformation [24]). Those hard constraints can then be added to $\mathcal{M}$. With every C_i (thus, with every Z_i, $i \in [p]$) one can associate a weight w_{Z_i} from $\mathbb{R}^+ \cup \{+\infty\}$ representing the user-dependent cost of modifying $\boldsymbol{x}^F$ in order to make C_i true. Whenever $w_{Z_i} = +\infty$, the constraint $Z_i = 0$ is added to $\mathcal{M}$.

The last thing to be specified in the constraint-based model is its objective function. The goal that is pursued is to minimize the dissimilarity to the current instance $\boldsymbol{x}^F$, thus the objective function to be minimized is

$$\sum_{A_i \in \mathcal{A}_C \cup \mathcal{A}_B} w_{A_i} \cdot \Big(\sum_{v_i^j \in RD_i^* | w_{v_i^j} \neq +\infty} w_{v_i^j} \cdot X_i^j \Big) +$$

$$\sum_{A_i \in \mathcal{A}_N} w_{A_i} \cdot \Big(\sum_{v_i^j \in RD_i | w_{v_i^j} \neq +\infty} w_{v_i^j} \cdot Y_i^j \Big) + \sum_{Z_i \in [p] | w_{Z_i} \neq +\infty} w_{Z_i} \cdot Z_i.$$

Notably, since no restrictions about the Boolean variables occurring in C_i are made in the definition, one can take advantage of such combinations to represent "local" preferences bearing only on a unique attribute from $\mathcal{A}$, as presented in the previous paragraphs. Of course, much more complex preferences can be stated using combinations which could not be represented using a weighted sum of "local" dissimilarity mappings that would be fully decomposed, i.e., each bearing on a single attribute.

Example 1 [cont'ed] It can be the case that, in order to get the loan granted, Bob is ready to wait for more than five years (for a cost of 10), and to pass a Ph.D. (for a cost of 7), but if the two conditions are actually needed, an extra cost of 20 has to be added. To ensure it, a 0/1 variable Z with cost $w_Z = 20$ such that Z is equivalent to $Y_2^1 \wedge X_3^3$ can be introduced.

$A_1 > 70$ must be added to $\mathcal{M}$ and set to 1; the domain theory must be updated accordingly to account for the threshold 70.

From an optimal solution of $\mathcal{M}$ one can extract in linear time a minimal contrastive instance from $\boldsymbol{X}^F$. This instance indicates how to minimally change the current instance in order to get a regression value belonging to the preset interval.

Example 1 [cont'ed] Let us wrap up Bob's case. Suppose that the three attributes A_1, A_2, A_3 as considered equally important by Bob (represented by the instance $\boldsymbol{x} = (A_1 = 70, A_2 = 29, A_3 = master)$) so that $w_{A_1} = w_{A_2} = w_{A_3}$. Suppose that the values for which a shift matters for Bob regarding attributes A_1 and A_2 are those already in RD_1 and RD_2 respectively, and also shifting the value of any attribute has the same (strictly positive, but not infinite) cost. For making things simpler, suppose finally that no combination of attributes has been specified by Bob. Then $\boldsymbol{x}$ has three minimal contrastive explanations given F and $I_{\boldsymbol{x}} = [90k, +\infty)$ that can be computed using **C**:

- $(1, 1, 1, 0, 0, 0, 1, 0)$, with regression value $100 + 20 - 30 = 90$: Bob changes only his income to more than $\$80k$,
- $(0, 1, 1, 0, 1, 0, 1, 0)$, with regression value $90 + 20 + 0 = 110$: Bob changes only his age (he waits for one year),
- $(0, 1, 1, 0, 0, 0, 0, 1)$, with regression value $90 + 30 - 30 = 90$: Bob changes only his level of education to get a Ph.D.

From the computational side, as we did it for **I**, we can ensure that considering a MILP solver for driving **C** in order to compute a minimal contrastive explanation is not a bad idea. Indeed, since $\boldsymbol{X}^F$ is finite, whatever the user preferences that are considered, a minimal contrastive explanation for an instance $\boldsymbol{x}$ given a boosted tree F and a target interval $I_{\boldsymbol{x}}$ exists if and only if a contrastive explanation for $\boldsymbol{x}$ given F and $I_{\boldsymbol{x}}$ exists. Furthermore, deciding whether a contrastive explanation for $\boldsymbol{x}$ given F and $I_{\boldsymbol{x}}$ exists is computationally demanding:

Proposition 2. *Deciding whether a contrastive explanation for $\boldsymbol{x}$ given F and $I_{\boldsymbol{x}}$ exists is an* NP*-hard problem.*

Proof. Consider any CNF formula $\alpha = \bigwedge_{i=1}^m \delta_i$ over variables $X_1, \ldots, X_n$ and, as in the proof of Proposition 1, turn α in linear time into a boosted regression tree $F = \{T_1, \ldots, T_m\}$ over $X_1, \ldots, X_n$. Set $\boldsymbol{x}$ to any truth assignment over $X_1, \ldots, X_n$ and $I_{\boldsymbol{x}}$ to $[m, m]$. α is satisfiable if and only if there exists an instance $\boldsymbol{x}_c$ over $X_1, \ldots, X_n$ such that $F(\boldsymbol{x}_c) = m$, which is equivalent to state that a contrastive explanation for $\boldsymbol{x}$ given F and $I_{\boldsymbol{x}}$ exists. □

4 Empirical evaluation

The two algorithms **I** and **C** have been evaluated on several datasets. For space reasons, we present only results about **C**.

Experimental setup The empirical protocol we considered was as follows. We focused on 8 datasets for regression, which are benchmarks found on standard repositories (kaggle, UCI, openML). These datasets are described in Table 1.

For each dataset, the algorithm `XGBoost` [9] was used to learn boosted regression trees. Numerical attributes were binarized on-the-fly by the boosted tree learning algorithm. Categorical attributes were one-hot encoded. All the hyper-parameters of `XGBoost` were set to their default values (100 trees per forest, with a depth at most 6). Thus, no specific tuning was performed. Indeed, our purpose was to evaluate the performance of **C** whatever the quality of the boosted regression trees we started with.

We performed a 10-fold cross validation: for each dataset, 10 boosted regression trees F were learned from a training set containing 80% of the dataset, and the accuracy of each boosted tree F was measured as its mean $R2$ score [16] over the remaining 20% of the dataset (the test set). The mean number $\#\mathcal{B}$ of distinct Boolean conditions used in F and the mean $R2$ score per dataset are reported in Table 1 as well. The mean $R2$ score obtained is quite good for most datasets, except for winequality-red and winequality-white, but we considered the corresponding boosted trees for those two datasets nevertheless (our goal is to be able to derive explanations for boosted trees as they have been learned, and not as one would like them to be). Finally, for each F, 10 instances were picked uniformly at random in the test set, leading to 100 instances per dataset.

In order to evaluate **C**, we needed to define target regression intervals. To make experiments that are representative of the intended use of the algorithm, intervals that are more or less distant to the regression value $F(\boldsymbol{x})$ of the instance $\boldsymbol{x}$ to be explained and that are more or less large had to be considered. To generate such target intervals, for each forest F, we computed first the minimal min and maximal Max regression values obtained by applying F to each of the 10 instances $\boldsymbol{x}$ associated with F. The interval given by min and Max was then partitioned into 100 intervals, each of them of size

$$size = \frac{Max - min}{100}.$$

Then, for each $\boldsymbol{x}$, the targeted regression value for $\boldsymbol{x}$ was set to

$$F(\boldsymbol{x}) + (shift \times size),$$

where $shift$ varied in $\{-50, -20, -10, -5, 5, 10, 20, 50\}$, reflecting the fact that the regression value of $\boldsymbol{x}$ can be expected to increase or to decrease to a more or less significant extent. Each target interval was centered on this value and its two bounds were obtained by substracting from it (respectively, adding to it) a value equal to $\frac{radius}{2} \times size$, where $radius$ varied in $\{1, 2, 3\}$. This led to target intervals having length equal to $size$, $2 \times size$, and $3 \times size$. Values of the two bounds were rounded to 2 decimals, and when rounding produced equal bounds, the resulting interval was not taken into account.

We also needed to represent user preferences. To this aim, we defined a "local" dissimilarity mapping per attribute $A_i \in \mathcal{A}$ (no combinations of attributes have been considered for the sake of simplicity) and we supposed all the attributes to be equally important (thus, the weight w_{A_i} of each attribute A_i has been set to the same value 1). For each attribute, the current value of A_i in $\boldsymbol{x}$ was protected with probability $\frac{1}{5}$ (leading to a hard constraint in the model in that case, since this value is required not to change when the attribute is protected). When A_i was not protected, the cost of changing its current value was set to 1 for categorical attributes and for Boolean attributes. For numerical attributes A_i, the cost of a change was set to 0 (respectively, 1, 5) when the target value of A_i laid within an interval of no more than 10% (respectively, between 10% and 25%, between 25% and 50%) from its current value; if the target value of A_i exceeded its current value by more than 50% or was lower than its current value by more than 50%, the change was not allowed (a hard constraint was added to ensure it).

Dataset	#A	#N	#C	#B	#I	#$\mathcal{B}$	R2
abalone	9	8	1	0	4177	38	0.99
airfoil-self-noise	5	5	0	0	1503	372	0.92
creditcard	29	0	0	29	284807	5546	0.96
bike-sharing-day	15	13	0	2	731	1648	0.99
bike-sharing-hour	13	11	0	2	17379	1396	0.99
steel-industry-data	9	6	3	0	731	3087	0.99
winequality-red	11	11	0	0	1599	1557	0.42
winequality-white	11	11	0	0	4898	1786	0.46

Table 1: Description of the datasets used, and performance of the boosted trees that have been learned. #A is the number of attributes per instance in the considered dataset. #N, #C, and #B are respectively the number of **n**umerical, **c**ategorical and **B**oolean attributes. #I is the number of instances in the dataset. #$\mathcal{B}$ is the mean number of distinct Boolean conditions and $R2$ is the mean R2 score over the 10 boosted trees learned for each dataset.

All experiments were conducted on a cluster equipped with quad-core bi-processors Intel(R) Xeon(R) CPU E5-2637 v4 @ 3.50GHz and 128 GiB of memory, running CentOS 8 with Linux version 4.18.0-301.1.el8.x86_64 kernel. Hyperthreading was disabled, and no cache sharing between cores was permitted. A timeout of 100s per instance was considered. The MILP models were solved using *Gurobi Optimizer* version 11.0.1 build v11.0.1rc0.

Experimental results Table 2 presents the empirical results. The leftmost column gives the name of the dataset. The next columns correspond to the different values of $shift$ considered in the experiments. For each dataset, there are three lines in the table, corresponding to the different values of $radius$ (1 for the first line, 2 for the second line, and 3 for the third line). The content of each cell indicates average computation times (in seconds) over 100 instances, and the corresponding standard deviations.

The computation times that have been found are rather small, showing **C** practical enough. Especially, only few timeouts occurred (to be more precise, 3 for bike-sharing-hour, 8 for steel-industry-data, and 36 for winequality-red), showing that for most of the problems considered in our experiments ($8 \times 100 \times 8 \times 3 = 19200$ problems), a minimal contrastive explanation has been computed in less than 100s.

The experiments that have been conducted also revealed that the $shift$ retained may have a strong impact on the computation times in some cases (consider for instance winequality-red with $shift = -50$ and $shift = 50$), but this does not happen for every dataset. The length of the target interval, as reflected by $radius$, appeared as having a limited impact in the experiments made (often, considering larger intervals made the problem easier, but this was not systematic). Of course, the number #I of instances in the dataset has no impact on the computation times since in our approach, contrastive explanations are not searched into the dataset, but in the solution space of the constraints representing the ML model that is used (and this solution space is not represented explicitly).

Finally, from our experiments, it is hard to draw any conclusion about possible correlations between the prediction performance and the time needed to derive contrastive explanations. For instance, the R2 score for winequality-white is low compared to the one obtained for steel-industry-data but no timeout occurred for winequality-white, while 8 timeouts have been reached for steel-industry-data.

5 Other related work

While there has been an abundant literature on XAI for the past five years, there is only few work that tackles the very same goal as the one pursued in this paper, i.e., deriving provably correct contrastive explanations for tree-based models used for regression. Indeed, in many approaches to XAI for tree ensembles (e.g., [23]), the explanations that are generated are based on feature importance, and are

Dataset	-50	-20	-10	-5	5	10	20	50
abalone	0.03 (0.01)	0.04 (0.02)	0.04 (0.02)	0.04 (0.02)	0.04 (0.02)	0.04 (0.02)	0.04 (0.02)	0.04 (0.02)
abalone	0.03 (0.01)	0.04 (0.01)	0.04 (0.01)	0.04 (0.02)	0.04 (0.02)	0.04 (0.01)	0.04 (0.01)	0.04 (0.01)
abalone	0.03 (0.01)	0.03 (0.01)	0.04 (0.02)	0.04 (0.01)	0.04 (0.01)	0.04 (0.02)	0.03 (0.01)	0.04 (0.01)
airfoil-self-noise	6.36 (3.22)	6.7 (3.48)	6.63 (3.43)	7.08 (3.92)	6.51 (3.88)	6.58 (3.54)	6.84 (3.96)	6.54 (3.67)
airfoil-self-noise	5.8 (2.46)	5.9 (2.95)	6.42 (3.3)	6.08 (3.36)	6.85 (3.43)	6.34 (3.42)	5.99 (2.98)	5.87 (2.9)
airfoil-self-noise	5.83 (3.22)	5.76 (2.8)	6.17 (3.07)	5.81 (3.07)	5.88 (2.97)	6.05 (3.16)	6.32 (3.34)	5.71 (2.22)
creditcard	16.18 (7.26)	14.72 (7.05)	14.82 (7.17)	15.48 (6.23)	14.15 (5.77)	13.7 (5.76)	13.22 (6.01)	12.96 (4.68)
creditcard	15.99 (9.52)	13.61 (6.53)	14.92 (9.22)	14.65 (6.61)	13.79 (5.65)	13.07 (5.59)	13.01 (6.09)	11.65 (4.24)
creditcard	14.9 (7.48)	13.52 (7.02)	12.5 (5.2)	13.5 (8.17)	12.82 (5.57)	12.89 (5.52)	13.41 (7.01)	12.19 (4.65)
bike-sharing-day	7.09 (5.27)	6.49 (3.73)	6.77 (3.43)	6.93 (3.33)	7.11 (3.67)	7.53 (3.72)	7.17 (3.14)	8.93 (4.52)
bike-sharing-day	7.41 (6.47)	6.26 (3.71)	6.84 (3.45)	6.47 (3.46)	6.57 (3.4)	7.32 (3.57)	7.11 (3.53)	9.44 (5.16)
bike-sharing-day	6.78 (5.31)	6.19 (3.65)	6.59 (3.2)	6.52 (3.53)	6.92 (3.81)	6.61 (3.35)	6.97 (3.32)	7.89 (4.33)
bike-sharing-hour	9.68 (11.96)	11.83 (14.7)	8.35 (9.33)	7.08 (5.03)	7.5 (6.54)	7.31 (5.42)	7.31 (5.73)	7.22 (5.07)
bike-sharing-hour	10.07 (13.3)	*10.05 (11.81)*	6.7 (5.01)	5.86 (4.13)	7.48 (6.45)	6.38 (4.86)	6.72 (4.64)	6.48 (4.16)
bike-sharing-hour	*9.52 (12.39)*	*10.34 (12.24)*	7.02 (6.37)	5.82 (5.09)	5.99 (4.06)	6.56 (4.81)	6.44 (4.62)	6.86 (4.54)
steel-industry-data	*18.17 (12.26)*	18.61 (12.48)	18.53 (10.25)	18.87 (13.07)	17.53 (9.66)	18.22 (10.01)	*19.0 (12.12)*	18.93 (10.04)
steel-industry-data	*17.93 (11.05)*	18.43 (10.72)	18.52 (11.88)	*19.47 (13.77)*	18.38 (10.35)	19.05 (10.89)	19.74 (13.52)	18.6 (10.21)
steel-industry-data	*19.52 (13.12)*	17.86 (9.97)	*18.97 (13.69)*	*19.91 (15.81)*	18.61 (10.11)	17.79 (9.52)	*19.8 (14.17)*	18.7 (10.2)
winequality-red	7.0 (3.75)	8.32 (5.26)	10.25 (8.73)	9.17 (7.3)	10.78 (7.81)	*10.4 (7.9)*	12.42 (14.62)	*15.97 (17.75)*
winequality-red	6.59 (3.45)	7.88 (5.24)	*9.47 (7.68)*	9.55 (8.48)	10.3 (7.94)	*9.11 (5.85)*	10.84 (10.09)	*14.17 (13.9)*
winequality-red	6.34 (3.31)	7.69 (5.63)	8.13 (5.25)	9.01 (9.86)	9.25 (6.89)	8.88 (7.4)	*10.29 (9.18)*	*13.46 (13.18)*
winequality-white	12.64 (6.02)	11.72 (6.09)	10.77 (5.36)	10.6 (5.05)	13.42 (8.92)	13.27 (6.33)	13.17 (6.36)	12.26 (5.12)
winequality-white	12.17 (5.54)	12.02 (5.59)	11.15 (5.33)	10.89 (5.51)	12.68 (8.88)	13.28 (6.54)	12.12 (5.29)	12.25 (5.3)
winequality-white	11.29 (5.28)	11.47 (5.3)	10.94 (5.65)	11.28 (5.96)	12.53 (8.48)	12.68 (6.57)	12.42 (6.17)	12.42 (7.85)

Table 2: Average computation times (in seconds) needed by **C** for deriving minimal contrastive explanations. Times are given in italics when at least one timeout occurred. In this case, the statistics are computed over the instances for which **C** terminated in due time.

not ensured to be faithful. Most of the time, classification issues are targeted, not regression ones, and/or the correct explanations that are looked for are abductive ones, not contrastive ones [10, 21, 14, 4]. Finally, when contrastive explanations are generated, they are typically not based on the Boolean conditions used in the tree ensemble, but come from the initial set of instances (the one containing the instances to be explained), which limits their generality.

A notable exception is [11], that presents an approach to the derivation of contrastive explanations for tree-based models, where explanations are built up from the Boolean conditions occurring in the model. As in our work, "local" dissimilarity mappings can be defined and exploited to indicate the extent to which shifting the values of the attributes in the input instance is acceptable by the user. However, such mappings are limited to attributes (combinations of attributes, considered in our work, are not supported in [11]). Furthermore, user-specified thresholds for numerical attributes are not handled (the thresholds considered for such attributes are precisely those occurring in the tree ensemble, and they can easily be distinct from those that are actually relevant for the user). Finally, though [11] mentions an R package for random forests, the code available on git (https://github.com/numb3r33/oae/blob/master/README.md) supports only scikit-learn's implementation of random forests, while we took advantage of `XGBoost` algorithm for boosted trees in our implementation. For all these reasons, we refrained from making an empirical comparison with [11].

6 Conclusion

We have presented a constraint-based approach to the generation of contrastive explanations for instances $\boldsymbol{x}$ given boosted trees used to represent regression functions. Two constraint encodings **I** and **C** have been pointed out.

I allows to compute the smallest interval containing all the regression values that are attainable given a set of characteristics of $\boldsymbol{x}$ that are protected (i.e., not amenable to change).

Using **C** one can derive minimal contrastive explanations for $\boldsymbol{x}$ given a target interval, i.e., instances with regression values within the specified interval and that are as close as possible to $\boldsymbol{x}$. Actionability constraints can be easily dealt with. Closeness is captured via a linear combination of user-dependent mappings reflecting preferences about value change for the attributes (or combinations of attributes) considered in the representation of $\boldsymbol{x}$.

Experiments have shown the approach as practical enough, in spite of the computational intractability of the problems that are tackled (each of them has been shown NP-hard). Thus, in the empirical evaluation of **C**, only few timeouts have been reached (around 2 per 1000 instances tested), despite the number of Boolean conditions $\#\mathcal{B}$ used in the boosted trees considered in the experiments, which was often large ($\#\mathcal{B}$ was greater than 1000 for 6 datasets out of 8).

This work calls for a number of perspectives. One of them concerns the elicitation of the weights associated with changes to attributes values, which is a challenging theory-oriented issue, especially when dealing with combinations of attributes.

From the practical side, the scalability of **C** remains to be investigated more extensively. Several dimensions that may impact the performance of **C** will have to be taken into account in the forthcoming experiments, especially the number of attributes used in the dataset, the depth of the trees that have been learned, the presence of combinations of attributes.

Another perspective for further research is to consider other representations of regression functions and determine the extent to which the proposed approach can be extended to such representations.

Acknowledgements

Many thanks to the anonymous reviewers for their comments and insights. This work has benefited from the support of the AI Chair EXPEKCTATION (ANR-19-CHIA-0005-01) of the French National Research Agency (ANR). It was also partially supported by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215.

References

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

[2] A. B. Arrieta, N. D. R., J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion*, 58:82–115, 2020.

[3] G. Audemard, S. Bellart, L. Bounia, F. Koriche, J.-M. Lagniez, and P. Marquis. On the computational intelligibility of Boolean classifiers. In *Proc. of KR'21*, pages 74–86, 2021.

[4] G. Audemard, S. Bellart, J.-M. Lagniez, and P. Marquis. Computing abductive explanations for boosted regression trees. In *Proc. of IJCAI'23*, pages 3432–3441, 2023.

[5] G. Audemard, J.-M. Lagniez, P. Marquis, and N. Szczepanski. On contrastive explanations for tree-based classifiers. In *Proc. of ECAI'23*, 2023.

[6] G. Audemard, J.-M. Lagniez, and P. Marquis. Code and data for "On the Computation of Contrastive Explanations for Boosted Regression Trees", 2024. Available at www.cril.fr/expekctation/.

[7] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci. Deep neural networks and tabular data: A survey. *CoRR*, abs/2110.01889, 2021.

[8] R. Caruana, S. M. Lundberg, M. T. Ribeiro, H. Nori, and S. Jenkins. Intelligible and explainable machine learning: Best practices and practical challenges. In *Proc. of KDD'20*, pages 3511–3512. ACM, 2020.

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

[10] A. Choi, A. Shih, A. Goyanka, and A. Darwiche. On symbolically encoding the behavior of random forests. In *Proc. of FoMLAS'20, 3rd Workshop on Formal Methods for ML-Enabled Autonomous Systems, Workshop at CAV'20*, 2020.

[11] Z. Cui, W. Chen, Y. He, and Y. Chen. Optimal action extraction for random forests and boosted trees. In *Proc. of KDD'15*, pages 179–188, 2015.

[12] F. Doshi-Velez and B. Kim. A roadmap for a rigorous science of interpretability. *CoRR*, abs/1702.08608, 2017. URL http://arxiv.org/abs/1702.08608.

[13] 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.

[14] A. Ignatiev, Y. Izza, P. Stuckey, and J. Marques-Silva. Using MaxSAT for efficient explanations of tree ensembles. In *Proc. of AAAI'22*, pages 3776–3785, 2022.

[15] Y. Izza, A. Ignatiev, P. J. Stuckey, and J. Marques-Silva. Delivering inflated explanations. In *Proc. of AAAI'24*, pages 12744–12753, 2024.

[16] R. Ling and D. Kenny. Correlation and causation. *Journal of the American Statistical Association*, 77:489, 1981.

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

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

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

[20] M. Nauta, J. Trienes, S. Pathak, E. Nguyen, M. Peters, Y. Schmitt, J. Schlötterer, M. van Keulen, and C. Seifert. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. *ACM Comput. Surv.*, 55(13s), 2023.

[21] A. Parmentier and T. Vidal. Optimal counterfactual explanations in tree ensembles. In *Proc. of ICML'21*, volume 139 of *Proceedings of Machine Learning Research*, 2021.

[22] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *CoRR*, abs/2103.11251, 2021.

[23] E. Strumbelj and I. Kononenko. A general method for visualizing and explaining black-box regression models. In *Adaptive and Natural Computing Algorithms ICANNGA*, volume 6594 of *LNCS*, pages 21–30, 2011.

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

[25] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu. Explainable AI: A brief survey on history, research areas, approaches and challenges. In *Proc. of NLPCC'19*, pages 563–574, 2019.

[26] F. Yang, M. Du, and X. Hu. Evaluating explanation without ground truth in interpretable machine learning. *CoRR*, abs/1907.06831, 2019. URL http://arxiv.org/abs/1907.06831.