Designing an XAI Interface for Tree-Based ML Models

Gilles Audemard[a], **Sylvie Coste-Marquis**[a], **Pierre Marquis** [a,b,*], **Mehdi Sabiri**[a] **and Nicolas Szczepanski**[a]

[a]Univ. Artois, CNRS, CRIL, Lens, France
[b]Institut Universitaire de France, France
name@cril.fr

Abstract. We present and evaluate empirically an XAI protocol for ruling interactions between a tree-based ML model (the AI system) and its user U, in the context of a prediction task. The pieces of knowledge held by U concerning the prediction task are supposed to be representable by a set of classification rules that is reliable and consistent, but (typically) incomplete. The proposed protocol aims to help U decide what to do with each prediction made by AI (accept it, reject it). It also aims to improve the quality of further predictions made by AI thanks to the expertise of U, and, reciprocally, to complete the pieces of knowledge held by U by leveraging the predictions made by AI. Experiments show that the approach can prove valuable in practice.

1 Introduction

The field of "eXplainable AI (XAI)" was born a couple of years ago [15] as a response to the opacity of Machine Learning (ML) models. The objective of XAI is to make ML models trustworthy enough. This goes through the generation of explanations for the predictions made, but is not limited to it; especially, the ability to correct wrong predictions is also part of the picture. Since its very beginning, XAI has given rise to a large amount of approaches (see e.g., [14, 22, 7, 1, 33, 32, 29, 13, 26, 23] for recent overviews).

In this paper, an XAI protocol for governing interactions between a user U and an AI system is described. The AI system has the form of an ML model used to make predictions about scenarios that U is not able to handle alone, since he/she is supposed to have only limited knowledge about the domain of the application under consideration. Such scenarios are represented by instances $\boldsymbol{x}$ from a set $\boldsymbol{X}$, and AI thus corresponds to a mapping associating any $\boldsymbol{x}$ with a class, taken from a finite set C. The protocol is implemented by an interface that encapsulates AI and is used by U for interacting with AI.

In our setting, a couple of assumptions are made about U and AI. On the one hand, U is supposed to hold reliable pieces of knowledge about the application domain that is targeted by AI. That mentioned, U is not requested to be an ML specialist. He/she is just supposed to hold pieces of knowledge that can be leveraged to associate predictions with instances (maybe only few of them when U is more a layperson than an expert), but it is not expected that U is able to properly associate a prediction with every possible instance (otherwise, U would probably not need the help of AI). In the classification case, we say that $U(\boldsymbol{x})$ is defined whenever U is able to classify $\boldsymbol{x}$, and that $U(\boldsymbol{x})$ is undefined otherwise. When $U(\boldsymbol{x})$ is undefined for at least one $\boldsymbol{x}$, U can thus be viewed as an "incomplete classifier". Whatever the case, we assume that U is confident with the pieces of knowledge he/she holds, so that if U knows how to classify a given instance but AI has a different opinion about the right class of $\boldsymbol{x}$, U is not ready to change his/her mind and considers that AI gets wrong on $\boldsymbol{x}$. Furthermore, U is supposed to classify instances in a consistent way. This means that the reasons used by U to classify instances (which can be modeled abstractly as classification rules) do not conflict: there cannot be two rules with distinct conclusions (the classes that are reached) and compatible conditions. Indeed, if this were the case, then U would need to classify in several classes at the same time instances matching such compatible conditions, which does not make sense (in classification problems, classes are supposed to be mutually exclusive). On the other hand, in order to be implemented, our protocol requires some XAI abilities: the generation of post-hoc, local, and faithful explanations and the correction of erroneous predictions must be feasible. Interestingly, this is the case when AI is a tree-based ML model (a decision tree, a random forest, or a boosted tree). Especially, the explanation and correction abilities that are expected are offered by the open source PyXAI library (https://github.com/crillab/pyxai).

The goal of our protocol is to assist U, helping him/her to decide what to do with each prediction made by AI (accept it, reject it), but more than that, to try and improve the quality of the further predictions made by AI, by leveraging the expertise of U, and, reciprocally, to complete the pieces of knowledge held by U by taking advantage of the predictions made by AI. The contribution of the paper consists of the definition of an XAI protocol, and an evaluation of this protocol on binary classification problems. Users U are simulated by generating incomplete (yet consistent) sets of classification rules from random forests. The chosen AI systems take the form of decision trees. We make them interact via our XAI protocol, and we measure how the accuracy of AI as well as the coverage of the set of rules used by U to classify instances evolve along with interactions. Of course, we do not claim that the interaction of an AI system with a real user should necessarily comply with the proposed protocol. Our objective is only to figure out whether synergetic effects (in terms of accuracy of AI and coverage of U) could result from an interaction guided by the protocol. Interestingly, the empirical results obtained show that clear benefits can be got.

The rest of the paper is organized as follows. Formal preliminaries are presented in Section 2 (we assume the reader acquainted with basic notions of propositional logic). Then our protocol is pointed out in Section 3. Experimental results are provided in Section 4. Finally, Section 5 concludes the paper. The code used, additional empirical results, and the datasets considered in the experiments are furnished

* Corresponding Author

as a supplementary material, available online [3].

2 Preliminaries

Classification and explanations We suppose that the instances $\boldsymbol{x}$ under consideration are described using attribute / value pairs. $A = \{A_1, \ldots, A_k\}$ is the set of *attributes* used. Each attribute of A is either Boolean, categorical, or numerical, and it takes its values in a *domain* D_i. An *instance* $\boldsymbol{x}$ over A is a tuple from $D_1 \times \ldots \times D_k$. Every $\boldsymbol{x} = (v_1, \ldots, v_k)$ is also viewed logically as the conjunctively-interpreted set $t_{\boldsymbol{x}}$ of *characteristics* $\{(A_i = v_i) : i \in [k]\}$.

The attributes of A are not necessarily independent, and a *domain theory* Σ (represented by a propositional circuit or a propositional formula) that makes precise how the attributes (and their values) are logically connected may be available [12].

In the single-label classification case, one considers a single set C of labels, denoting classes. Then, a *classifier* f over A is a mapping from $\boldsymbol{X}$ to C. It is a *binary classifier* when $C = \{0, 1\}$. For a binary classifier f, an instance $\boldsymbol{x} \in \boldsymbol{X}$ is *positive* when $f(\boldsymbol{x}) = 1$ and it is *negative* when $f(\boldsymbol{x}) = 0$.

When the classifier f is a tree-based ML model (a decision tree [6, 25], a random forest [5] or a boosted tree [11]), f can also be viewed as a Boolean function over the set of Boolean conditions used in f. Stated otherwise, we can assume that f is a classifier over a set of Boolean attributes, corresponding to the Boolean conditions used in f. In that case, the Boolean attributes are usually non-independent. Indeed, they can come from the same numerical or categorical attributes used at start for learning the classifier (for example, we can consider a Boolean attribute $x_1 = (age > 21)$ related to a numerical attribute age but also a a Boolean attribute $x_2 = (age > 18)$ which is connected to age and logically linked to x_1 since x_1 cannot be true while x_2 would be false. The corresponding characteristics are *literals* (here, x_1, x_2 and the complementary literals $\overline{x_1}, \overline{x_2}$). The domain theory indicating how the characteristics are connected could be the formula $\Sigma = \overline{x_1} \vee x_2$ in that case.

Given a classifier f over A and an instance $\boldsymbol{x} \in \boldsymbol{X}$, an *abductive explanation* t for $\boldsymbol{x}$ given f [17] is a conjunctively-interpreted set $t \subseteq t_{\boldsymbol{x}}$ of characteristics of $\boldsymbol{x}$, such that every instance $\boldsymbol{x}' \in \boldsymbol{X}$ covered by t (i.e., satisfying $t \subseteq t_{\boldsymbol{x}'}$) is such that $f(\boldsymbol{x}') = f(\boldsymbol{x})$. Every instance $\boldsymbol{x}$ has an abductive explanation given f, since $t = t_{\boldsymbol{x}}$ satisfies the conditions that are requested. Of course, such a trivial explanation is useless in general, when they exist, abductive explanations that do not coincide with $t_{\boldsymbol{x}}$ are preferred. Often, subset-minimal abductive explanations [19] (aka sufficient reasons [10] or PI-explanations [30]) and minimum-sized abductive explanations [4, 2] are targeted.

Classification rules We now present a couple of definitions pertaining to classification rules. A *classification rule* r is a pair $r = t \rightarrow c$ where t is a conjunction of characteristics over A and c is an element of C. t is the *condition* of r, and c is the *conclusion* of r. A classification rule $r = t \rightarrow c$ *classifies* an instance $\boldsymbol{x}$ over A as c if and only if $\boldsymbol{x}$ satisfies t. In that case, we note $r(\boldsymbol{x}) = c$.

Given a domain theory Σ and two classification rules $r_1 = t_1 \rightarrow c_1$ and $r_2 = t_2 \rightarrow c_2$, we say that r_1 *specializes* r_2 (or, equivalently, that r_2 *generalizes* r_1) if and only if $c_1 = c_2$ and t_2 is a logical consequence of $t_1 \wedge \Sigma$. A rule $r_1 = t_1 \rightarrow c_1$ is *in conflict* with a set R of classification rules given a domain theory Σ if and only if there exists a rule $r_2 = t_2 \rightarrow c_2$ of R such that $c_1 \neq c_2$ and $t_1 \wedge t_2 \wedge \Sigma$ is consistent. A set R of classification rules is said to be:

- *consistent* if and only if for every pair r_1, r_2 of rules of R such that $r_1 = t_1 \rightarrow c_1$ and $r_2 = t_2 \rightarrow c_2$, if $c_1 \neq c_2$ then $t_1 \wedge t_2 \wedge \Sigma$ is inconsistent.
- *complete* if and only if for every instance $\boldsymbol{x}$ over A, there exists at least one rule $r = t \rightarrow c$ such that $\boldsymbol{x}$ satisfies t. Stated otherwise, any instance $\boldsymbol{x}$ over A is classified by a rule of R.
- *simplified* if and only if for every pair r_1, r_2 of distinct rules of R, r_1 does not specialize r_2 given Σ.

For an instance $\boldsymbol{x}$, when all the rules of R that classify $\boldsymbol{x}$ have the same conclusion, there is no ambiguity about the way R classifies $\boldsymbol{x}$ provided that at least one rule r of R classifies $\boldsymbol{x}$. In this case, we say that $R(\boldsymbol{x})$ is defined and we note $R(\boldsymbol{x}) = r(\boldsymbol{x})$. In the remaining case, we say that $R(\boldsymbol{x})$ is undefined, noted $R(\boldsymbol{x}) = \bot$. When R is consistent, $\boldsymbol{x}$ is classified by R precisely when there exists a rule r of R that classifies $\boldsymbol{x}$. When R is consistent and complete, R classifies every instance $\boldsymbol{x}$ over A. Stated otherwise, R is a classifier. When R is consistent, complete, and simplified, every instance $\boldsymbol{x}$ over A is classified by a unique rule r of R.

It is easy to show that the consistency of a set R of classification rules can be decided in quadratic time in $|R| + |\Sigma|$ whenever Σ is tractable for clausal entailment. Contrastingly, deciding the completeness of a set R of classification rules is a coNP-complete problem, even when Σ is an empty set of clauses. Every set of rules can be simplified, i.e., turned into an equivalent set of rules, in quadratic time in $|R|+|\Sigma|$ whenever Σ is tractable for clausal entailment. Here, two sets of rules R_1 and R_2 are said to be equivalent when the set of instances $\boldsymbol{x}$ for which $R_1(\boldsymbol{x})$ is defined is the same as the set of instances $\boldsymbol{x}$ for which $R_2(\boldsymbol{x})$ is defined and for each $\boldsymbol{x}$ in this set, we have $R_1(\boldsymbol{x}) = R_2(\boldsymbol{x})$.

Let us illustrate the notions presented in this section, using a simple example. Let $A = \{a, b, c\}$ be a set of Boolean attributes and $C = \{1, 0\}$, i.e., we consider a binary classifier. For the sake of simplicity, we suppose that Σ is valid (i.e., there is no domain theory). The characteristics of a Boolean attribute x_i of A are denoted x_i (when the attribute takes the value 1) and $\overline{x_i}$ (when the attribute takes the value 0). Let R be the following set of classification rules:

$$R = \{(a \wedge b) \rightarrow 1, b \rightarrow 1, (\bar{b} \wedge c) \rightarrow 0, (a \wedge c) \rightarrow 0\}.$$

R is not consistent, not complete, and not simplified. Indeed, $b \rightarrow 1$ is in conflict with $(a \wedge c) \rightarrow 0$ since the former rule classifies $\boldsymbol{x} = (1, 1, 1)$ as 1 while the latter rule classifies $\boldsymbol{x}$ as 0. Therefore, R is not consistent. R is not complete because there is no rule in R classifying the instance $(0, 0, 0)$. Thus, $R((0, 0, 0))$ is undefined. Finally, R is not simplified since the rule $(a \wedge b) \rightarrow 1$ of R specializes the rule $b \rightarrow 1$ belonging to R as well.

3 An XAI protocol for tree-based models

Ideally, benefits should be gained from interactions between U and AI, from both sides. The goal should not consist only in accepting or rejecting the predictions made by AI about instances. Indeed, from the point of view of AI, the interaction with U should lead to a more accurate predictor: AI should be able to take advantage of the expertise of U for making less classification mistakes. From the perspective of U, the interaction with AI should lead U to hold more complete knowledge about classification issues.

In the following, we suppose that U is represented by a set of rules R_U that is consistent and simplified, but not complete (otherwise, there would be no need of AI!). The elements of R_U can be viewed as pieces of knowledge the user U is quite confident in. The classifier AI (whatever it is) can always be associated with a set R_{AI} of classification rules that is consistent and complete. Especially, $\{t \to AI(\boldsymbol{x}) \mid t$ is an abductive explanation for $\boldsymbol{x}$ given AI and $\boldsymbol{x} \in \boldsymbol{X}\}$ is such a set of classification rules. This set is of size exponential in the number of attributes used to describe the instances, but we do not need to compute it entirely (the only important point to keep in mind is that any classifier corresponds to a set of classification rules that is consistent and complete). Instead, some classification rules of R_{AI} are going to be derived on demand, i.e., whenever a prediction about an instance $\boldsymbol{x}$ is computed. The rules derived concern $\boldsymbol{x}$ and are used to decide whether the prediction $AI(\boldsymbol{x})$ must be accepted or rejected. Finally, the classification rules of R_U are supposed to be more reliable than the classification rules of R_{AI}.

On this basis, we now show how to design an XAI protocol ensuring beneficial interactions between U and AI. Whenever an instance $\boldsymbol{x}$ is considered, the approach consists in computing classification rules that hold for AI and are about $\boldsymbol{x}$. Those rules are derived from explanations for the instances $\boldsymbol{x}$ for which predictions by AI are requested. Faithfulness (aka correctness, soundness, or fidelity) ensures that the explanations that are generated accurately reflect the decision process followed by the model. Then a policy is defined, based on the conflicts of those rules with the classification rules of U. Depending on the case, the policy indicates whether $\boldsymbol{x}$ should be accepted or rejected, whether (and how) AI should be corrected, and whether the classification rules of U should be completed.

Deciding what to do with the predictions and doing more A first, albeit important observation is that classification rules can be easily generated in linear time from faithful explanations. Thus, if t is an abductive explanation for $\boldsymbol{x}$ given a classifier AI, then $r = t \to AI(\boldsymbol{x})$ is a classification rule that can be deduced from AI: for every instance $\boldsymbol{x}'$ satisfying t, it is guaranteed that $AI(\boldsymbol{x}') = AI(\boldsymbol{x})$.

Suppose that a rule $r = t \to c$ from R_{AI} has been generated from an abductive explanation for the instance $\boldsymbol{x}$ under consideration. This rule r classifies $\boldsymbol{x}$ as c. Four distinct cases (1) to (4) are then worth being considered (see Table 1).

Case (1). Suppose first that $\boldsymbol{x}$ is also classified by R_U (so $R_U(\boldsymbol{x})$ is defined), in such a way that $R_U(\boldsymbol{x}) \neq AI(\boldsymbol{x})$. In this case, there exists at least one rule $r_U = t_1 \to c_1$ in R_U such that t_1 covers $\boldsymbol{x}$ and $c_1 \neq c$. By definition, since t covers $\boldsymbol{x}$, r is in conflict with r_U, thus with R_U. Accordingly, in case (1), *the prediction $AI(\boldsymbol{x})$ must be rejected, and AI must be corrected by r_U*. Of course, it may be the case that several rules r_U of R_U classifies $\boldsymbol{x}$ (in the same way, i.e., as c_1, otherwise $R_U(\boldsymbol{x})$ would not be defined). Then, *AI must be corrected by every such r_U*. Indeed, r is necessarily in conflict with each of them.

For example, suppose that $R_U = \{(a \wedge b) \to 1, (b \wedge c) \to 1, \overline{b} \to 0\}$. One can easily check that R_U is consistent and simplified, but not complete (e.g., $(0, 1, 0)$ is not classified by R_U). Suppose that AI is equivalent to the following set of classification rules: $\{\overline{a} \to 1, \overline{b} \to 1, (a \wedge b) \to 0\}$, which is consistent, complete, and simplified. Consider the instance $(1, 1, 1)$. It is classified by R_U as a positive instance, using the rule $r_U = (a \wedge b) \to 1$, but also using the rule $(b \wedge c) \to 1$. $(1, 1, 1)$ is classified by AI as a negative instance using the rule $r = (a \wedge b) \to 0$. Hence, the prediction made by AI must be rejected, and AI must be corrected by $r_U = (a \wedge b) \to 1$ and also by $(b \wedge c) \to 1$. We can check that $r = (a \wedge b) \to 0$ is in conflict with $r_U = (a \wedge b) \to 1$, but also with $(b \wedge c) \to 1$.

Case (2). Suppose now that the instance $\boldsymbol{x}$ is classified by R_U (so $R_U(\boldsymbol{x})$ is defined), in such a way that $R_U(\boldsymbol{x}) = AI(\boldsymbol{x})$. In this case, r does not conflict with the rules of R_U that classify $\boldsymbol{x}$ since all those rules necessarily have the same conclusion c. *The prediction $AI(\boldsymbol{x})$ can be accepted since it complies with the prediction achieved by U using more reliable pieces of knowledge about the prediction task.* However, suppose that r is in conflict with some rules r_U of R_U. Then AI must be corrected by every such r_U. This allows one to anticipate additional discrepancies (i.e., case (1)) as to the predictions achieved by R_U and AI on other instances.

For example, suppose that $R_U = \{(a \wedge b) \to 1, (b \wedge c) \to 1, \overline{b} \to 0\}$, as in the previous example. Assume this time that AI is equivalent to the following set of classification rules: $\{c \to 1, \overline{c} \to 0\}$, which is consistent, complete, and simplified. Consider again the instance $(1, 1, 1)$. It is classified by R_U as a positive instance and by AI as a positive instance. Thus, the prediction made can be accepted. However, it turns out that the classification rule $r = c \to 1$ used by AI to classify $(1, 1, 1)$ is in conflict with the classification rule $r_U = \overline{b} \to 0$ of R_U. Thus, AI should be corrected by $r_U = \overline{b} \to 0$. Once this correction is made, the instance $(1, 0, 1)$ is classified in the same way (as a negative instance) by R_U and AI, while the classifications of this instance $(1, 0, 1)$ by R_U and AI were different before the correction. Indeed, before the correction, $(1, 0, 1)$ was classified as a negative instance by R_U and as a positive instance by AI. Since U is considered as more reliable than AI, $(1, 0, 1)$ must be classified as a negative instance. Correcting AI by r_U as soon as the instance $(1, 1, 1)$ is treated prevents AI from making a wrong prediction that would be achieved if the instance $(1, 0, 1)$ was considered afterwards.

Case (3). Suppose now that the instance $\boldsymbol{x}$ is not classified by R_U (i.e., $R_U(\boldsymbol{x})$ is undefined). In this situation, as in case (2), it may happen nevertheless that r is in conflict with some rules r_U of R_U. Suppose that this is the case. Then AI must be corrected by every such r_U. Again, this correction is useful to prevent the occurrence of discrepancies (i.e., case (1)) as to the predictions achieved by R_U and AI on instances that eventually will be considered later.

For example, suppose that $R_U = \{(a \wedge b) \to 1, (b \wedge c) \to 1, \overline{b} \to 0\}$, which is consistent, simplified, but not complete. Suppose that AI is equivalent to the following set of classification rules: $\{c \to 1, \overline{c} \to 0\}$, which is consistent, complete, and simplified. Consider now the instance $(0, 1, 0)$. This instance is not classified by R_U and it is classified by AI as a negative instance. However, it turns out that the classification rule $r = \overline{c} \to 0$ used by AI to classify $(0, 1, 0)$ is in conflict with the classification rule $r_U = (a \wedge b) \to 1$ of R_U. Thus, AI should be corrected by $r_U = (a \wedge b) \to 1$. If this correction is made, the instance $(1, 1, 0)$ will be classified in the same way (as a positive instance) by R_U and AI, while the classifications of $(1, 1, 0)$ by R_U and AI would differ otherwise.

About the decision to be made concerning the prediction $AI(\boldsymbol{x})$ when r is in conflict with some rules r_U of R_U, several policies can be adopted. On the one hand, from the point of view of the *brave policy*, *the prediction $AI(\boldsymbol{x})$ can be accepted* since no rule of R_U indicates that the prediction should be rejected. On the other hand, from the point of view of the *cautious policy*, *the prediction $AI(\boldsymbol{x})$ should be rejected* since the rule r of AI used to make the decision is not fully correct. Indeed, it classifies some instances in a different way than R_U (thus, r has to be specialized). Other policies could be defined easily by taking account of the number and the specificity of the rules of R_U r is in conflict with.

Case (4). The remaining case covers the situations where $R_U(\boldsymbol{x}) = AI(\boldsymbol{x})$ or $R_U(\boldsymbol{x}) = \perp$, and there is no conflict between r and R_U. In such a case, it makes sense *to accept the prediction made* by AI since there is no argument against it. No correction step is needed, r can simply be added to R_U and the resulting set can then be simplified (it can be the case that r specializes / generalizes some rules of R_U).

For example, suppose that $R_U = \{(a \wedge b) \rightarrow 1, (\overline{a} \wedge \overline{b}) \rightarrow 0\}$, which is consistent, simplified, but not complete ($(0,1,0)$ is not classified by R_U). Suppose that AI is equivalent to the following set of classification rules: $\{a \rightarrow 1, \overline{a} \rightarrow 0\}$, which is consistent, complete, and simplified. Let us first consider the instance $(1,1,1)$. This instance is classified by R_U as a positive instance and by AI as a positive instance. The classification rule $r = a \rightarrow 1$ used by AI to classify $(1,1,1)$ is not in conflict with any classification rule of R_U. The rule $r_U = (a \wedge b) \rightarrow 1$ of R_U can be replaced by the more general rule $r = a \rightarrow 1$ of R_{AI}, thus leading to a new set of rules for U, given by $\{a \rightarrow 1, (\overline{a} \wedge \overline{b}) \rightarrow 0\}$. This set is consistent and simplified. Consider now the instance $(0,1,0)$. This instance is not classified by R_U and it is classified by AI as a negative instance. The classification rule $r = \overline{a} \rightarrow 0$ used by AI to classify $(0,1,0)$ is not in conflict with any classification rule of R_U. The rule $r_U = (\overline{a} \wedge \overline{b}) \rightarrow 0$ of R_U can be replaced by the more general rule $r = \overline{a} \rightarrow 0$ of R_{AI}, thus leading to the set of rules $\{(a \wedge b) \rightarrow 1, \overline{a} \rightarrow 0\}$, that is consistent and simplified.

We can easily observe on this example that the ability to derive abductive explanations that are as general as possible (especially, subset-minimal ones) has an impact on the resulting set of rules for U. Indeed, suppose that AI is now given by the set of classification rules $\{(a \wedge b) \rightarrow 1, (a \wedge \overline{b}) \rightarrow 1, \overline{a} \rightarrow 0\}$ which is consistent, complete, and simplified, and that the instance to be classified is $(1,1,1)$. Obviously enough, this set of rules is equivalent to $\{a \rightarrow 1, \overline{a} \rightarrow 0\}$, since each of two rules $(a \wedge b) \rightarrow 1$ and $(a \wedge \overline{b}) \rightarrow 1$ could be replaced by the more general rule $a \rightarrow 1$, reflecting the fact that the corresponding abductive explanation $(a \wedge b)$ for $(1,1,1)$ given AI is not subset-minimal. Adding to R_U the classification rule $(a \wedge b) \rightarrow 1$ used by AI to classify $(1,1,1)$ would let R_U unchanged.

Table 1 synthesizes the conditions to be satisfied for each of the four cases above, and indicates for each case the interaction that takes place between U and AI whenever an instance $\boldsymbol{x}$ is considered. Remind that $r = t \rightarrow AI(\boldsymbol{x})$ is the classification rule deduced from AI that is used to classify $\boldsymbol{x}$. Case (1) captures the scenarios for which a disagreement between U and AI about the right class of $\boldsymbol{x}$ exists. Case (2) is the case when U and AI agree about the class of $\boldsymbol{x}$, but a conflict between r and R_U exists nevertheless. Case (3) corresponds to the situation when R_U does not classify $\boldsymbol{x}$, but there is a conflict between r and R_U. Finally, case (4) gathers the situations for which no conflict exists between r and R_U. Figure 1 illustrates the various interactions that can take place between U and AI (and how those interactions are triggered), when they are ruled by the XAI protocol defined above. An interaction starts whenever the user furnishes an instance $\boldsymbol{x}$ to the AI system and asks for a prediction $AI(\boldsymbol{x})$. As a key ingredient of this protocol, the rectification ability is paramount to update AI when conflicts are detected.

Case	**Conditions**	**Effects**
(1)	$R_U(\boldsymbol{x}) \neq \bot$ $R_U(\boldsymbol{x}) \neq AI(\boldsymbol{x})$ (r is in conflict with R_U)	reject $AI(\boldsymbol{x})$ correct AI by every $r_U \in R_U$ in conflict with r
(2)	$R_U(\boldsymbol{x}) = AI(\boldsymbol{x})$ r is in conflict with R_U	accept $AI(\boldsymbol{x})$ correct AI by every $r_U \in R_U$ in conflict with r
(3)	$R_U(\boldsymbol{x}) = \bot$ r is in conflict with R_U	accept or reject $AI(\boldsymbol{x})$ correct AI by every $r_U \in R_U$ in conflict with r
(4)	$R_U(\boldsymbol{x}) = AI(\boldsymbol{x})$ or $R_U(\boldsymbol{x}) = \bot$ r is not in conflict with R_U	accept $AI(\boldsymbol{x})$ add r to R_U simplify the resulting set

Table 1: Leveraging explanation and rectification facilities offered by PyXAI to design an XAI protocol.

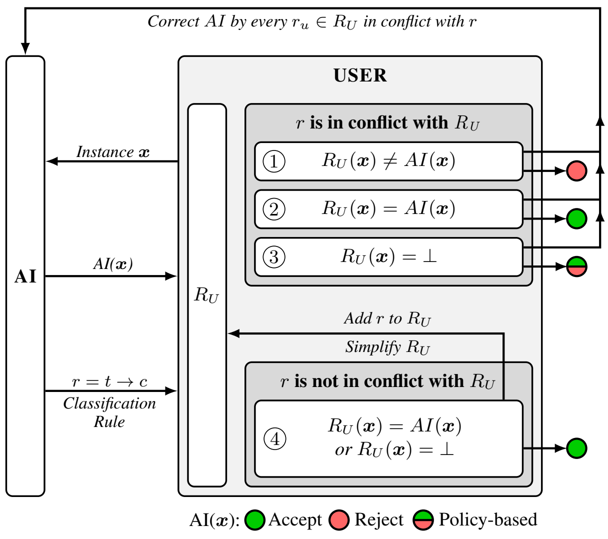
Correct AI by every $r_u \in R_U$ in conflict with r
USER
AI
r is in conflict with R_U
Instance $\boldsymbol{x}$
① $R_U(\boldsymbol{x}) \neq AI(\boldsymbol{x})$
② $R_U(\boldsymbol{x}) = AI(\boldsymbol{x})$
③ $R_U(\boldsymbol{x}) = \bot$
AI($\boldsymbol{x}$)
R_U
Add r to R_U
Simplify R_U
$r = t \rightarrow c$
Classification Rule
r is not in conflict with R_U
④ $R_U(\boldsymbol{x}) = AI(\boldsymbol{x})$ or $R_U(\boldsymbol{x}) = \bot$
AI($\boldsymbol{x}$): Accept Reject Policy-based

Figure 1: An XAI interface enabling many interactions between U and AI to take place.

Correcting tree-based classifiers using rectification Rectification is a principled approach to the update of classifiers AI [8], that can be used to implement a correction operation of AI by rules of R_U, each time the XAI protocol presented in the previous section asks for it.

By construction, in the single-label classification case, the rectification of a classifier AI by a classification rule r_U leads to a classifier that classifies every instance as AI did it, except for those instances that are classified by r_U, which are classified by the rectified classifier as r_U requests it [9].

Furthermore, when AI is a tree-based model (e.g., a decision tree [6, 25], a random forest [5], a boosted tree [11]), the resulting, rectified classifier can be computed in time polynomial in the size of the input (AI and r_U) [9]. Especially, when dealing with binary classification problems, the rectification of a decision tree T by a classification rule $r_U = t \rightarrow 1$ (resp. $r_U = t \rightarrow 0$) is equivalent to $T \vee t$ (resp. $T \wedge \neg t$). Rectifying a random forest boils down to rectifying every tree in the forest, and rectifying a boosted tree by a classification rule r_U, albeit a bit more tricky, can be done as well in polynomial time (see [9] for details).

We have also shown that when the classification rules used to rectify a classifier AI come from a set of rules that is consistent, the rectification of AI by a conjunction of such rules is equivalent to the iterated rectification of AI by each of the rules of the conjunction (and the sequence of rules used is irrelevant). Thus, in our policy, each time AI must be corrected by a (conjunctively-interpreted) set of rules $\{r_U^1, \ldots, r_U^k\}$ from R_U, a way to achieve it is first to rectify AI by r_U^1, then to rectify the resulting classifier by r_U^2, and so on.

We now illustrate the correction process by rectification by stepping back to the examples considered in the previous subsection, focusing on cases (1) to (3) since in case (4), no correction is required. For each case (1) to (3), a set of classification rules equivalent to AI

once rectified is provided.

- In the example for **Case (1)**, once AI has been rectified by r_U, AI becomes equivalent to the set of classification rules $\{\top \to 1, \bot \to 0\}$, where every instance is classified as positive. If the resulting corrected AI system is further used to classify $(0,0,0)$, which is classified as a negative instance by R_U using the rule $\overline{b} \to 0$, AI needs to be corrected once more, leading to a system equivalent to the set of classification rules $\{b \to 1, \overline{b} \to 0\}$.
- In the example for **Case (2)**, once AI has been rectified by r_U, AI becomes equivalent to the set of classification rules $\{(b \wedge c) \to 1, \overline{b} \to 0, \overline{c} \to 0\}$.
- In the example for **Case (3)**, once AI has been rectified by r_U, AI becomes equivalent to the set of classification rules $\{(a \wedge b) \to 1, c \to 1, (\overline{a} \wedge \overline{c}) \to 0, (\overline{b} \wedge \overline{c}) \to 0\}$.

4 Experiments

Having an end user U available to make experiments is demanding, especially when he/she is supposed to be acquainted with the application domain. Furthermore, making an evaluation robust enough would require to consider several application scenarios, thus to take advantage of several users (one per domain targeted), making the task even harder in practice. This is why we decided to simulate end users by artificial agents.

A key aspect of the protocol for governing XAI interactions that is presented in the paper is that it is based on trustful ingredients. On the one hand, the classification rules that are extracted from faithful abductive explanations are ensured to be correct, such rules indicate for sure how AI classifies instances. This contrasts with several popular approaches to XAI (including LIME [27], Anchors [28], and SHAP [21]) for which one can find "counterexamples" for the explanations that are generated, i.e., pairs of instances sharing an explanation but leading to distinct predictions [18, 16]. On the other hand, the rectification approach ensures that the corrections that are requested are effective.

The rationale for each step in the protocol is made precise in the previous section, and it is independent of the nature of U (artificial or human). The protocol guarantees, by design, that provided that all the classification rules in R_U are correct, the accuracy of AI will never decrease whenever a rectification-based correction takes place (cases (1) to (3)). Reciprocally, the protocol guarantees, by design, that whenever the rule r used by AI is correct, the accuracy of U will never decrease when U adopts this rule (case (4)). Finally, the coverage of U cannot decrease.

Of course, the guarantees that are listed are offered subject to conditions (i.e., the classification rules that are exchanged must be correct), and those conditions cannot be entirely evaluated in general since an oracle (i.e., a 100% correct predictor) is not available. Thus, the very purpose of our experiments is to determine how much, in practice, the accuracy of AI and the coverage of U evolve when interactions between AI and U, ruled by the XAI protocol presented in the previous section, occur.

Empirical protocol Let us now make precise the empirical protocol that has been followed in the experiments made, and the values of the various hyperparameters that have been considered.

In our experiments, we focused on a binary classification problem: $C = \{1, 0\}$. U was represented by a set R_U of classification rules, that is consistent and simplified. We considered 18 datasets,[1] reported in Table 2, which are standard datasets available online from UCR (www.timeseriesclassification.com), OpenML (www.openml.org), or UCI (archive.ics.uci.edu/ml/). Those datasets with a suffix name of the form "$_*\text{vs}_{**}$" concern primarily classification problems that are not binary, so they have been turned into binary classification problems by focusing on instances from two classes only (noted "*" and "**").

In Table 2, the first column "Dataset" gives the name of the dataset, the second column $\#F$ gives the number of features once the categorical attributes have been one-hot encoded, the third column $\#I$ indicates the number of instances in the dataset, and the last column "Repository" makes precise the source the dataset comes from. Some of these datasets are based on many features and some of them contain many instances.

Dataset	#F	#I	#B	Repository
arrowhead$_0$vs$_1$	249	146	93	UCR
arrowhead$_0$vs$_2$	249	146	86	UCR
arrowhead$_1$vs$_2$	249	130	104	UCR
australian	38	690	51	openML
balance$_1$vs$_2$	4	576	10	UCI
biodegradation	41	1055	69	openML
breastTumor	37	286	38	openML
cleveland	22	303	25	openML
compas	11	6172	13	openML
contraceptive$_0$vs$_1$	21	962	25	UCI
contraceptive$_0$vs$_2$	21	1140	30	UCI
contraceptive$_1$vs$_2$	21	844	31	UCI
divorce	54	170	36	UCI
nerve$_0$vs$_1$	1500	84	126	UCR
nerve$_0$vs$_2$	1500	107	140	UCR
nerve$_1$vs$_2$	1500	135	99	UCR
spambase	57	4601	95	UCI
wine	234	111	98	UCR

Table 2: Description of the datasets used in the experiments.

Given a dataset, we partitioned its elements into four pairwise disjoint subsets: a training set (gathering 30% of the instances from the dataset), a set of instances used to generate R_U (this set gathered $\frac{1}{4} \cdot 70\% = \frac{7}{40}$ of the instances from the dataset), a set of instances used for triggering the interactions between AI and the user U (this set gathered $\frac{1}{4} \cdot 70\% = \frac{7}{40}$ of the instances from the dataset and was upper bounded to 100 instances), and a test set used to evaluate empirically the accuracy of AI and R_U (this test set gathered $\frac{1}{2} \cdot 70\% = \frac{7}{20}$ of the instances from the dataset).

R_U was generated as follows. First, a random forest $F = \{T_1, \cdots, T_p\}$ was learned from the training set, using the algorithm furnished in the Scikit-learn library [24]. In our experiments, p was set to 100 and the maximum size of the sample used to decide to stop splitting a node of any tree T_i ($i \in [p]$) of F was set to half the number of instances in the training set. The fourth column ($\#B$) in Table 2 indicates the number of distinct Boolean conditions used in F. We considered a classification threshold $\theta \in [\frac{1}{2}, 1)$ (in our experiments $\theta = 70\%$). Then we picked up at random alternately a positive instance and a negative instance in the set of instances used to generate R_U, and for each instance $\boldsymbol{x}$ selected, we considered that the instance $\boldsymbol{x}$ was classified by F as a positive (resp. negative) instance given the threshold θ if the proportion of the trees of F classifying $\boldsymbol{x}$ as positive (resp. negative) exceeded θ. $\boldsymbol{x}$ was considered as not classified by F given the threshold θ in the remaining case. The next step was to compute a *majoritary reason* for $\boldsymbol{x}$ given F, that takes θ into account. To this end, we needed to slightly generalize Definition

[1] We removed from the datasets considered at start those leading to a initial coverage of R_U equal to 100% (adult, balance$_0$vs$_1$, balance$_0$vs$_2$, bank, german).

3 from [2], as follows:

- If $R(\boldsymbol{x}) = 1$, then t is a *majoritary reason* for $\boldsymbol{x}$ given F and θ if and only if t is a subset of $t_{\boldsymbol{x}}$, the proportion of the trees of F t is an implicant of which exceeds θ, and no proper subset of t satisfies the latter condition.
- If $R(\boldsymbol{x}) = 0$, then t is a *majoritary reason* for $\boldsymbol{x}$ given F and θ if and only if t is a subset of $t_{\boldsymbol{x}}$, the proportion of the negations of the trees[2] of F t is an implicant of which exceeds θ, and no proper subset of t satisfies the latter condition.

Such majoritary reasons are abductive explanations. In the general case, they are not subset-minimal explanations. However, each implicant test in the above definition can be achieved in time linear in the size of the input (t and T_i), so that a majoritary explanation for $\boldsymbol{x}$ given F and θ can be computed efficiently in practice using a greedy algorithm that starts with $t_{\boldsymbol{x}}$ (see [2] for details). In order to generate rules that cover sufficiently many instances (i.e., rules with a condition part that is not too specific), we looked for majoritary reasons that are sufficiently small. This has been achieved by running several times the greedy algorithm, using at each run a different elimination ordering for the features of $t_{\boldsymbol{x}}$, and keeping at the end a smallest majoritary reason obtained over the runs. In our experiments, we considered 50 runs of the greedy algorithm on each instance $\boldsymbol{x}$ from the set of instances used to generate R_U, such that $\boldsymbol{x}$ was classified by F given the threshold θ.

A classification rule $r_U = t \rightarrow 1$ (resp. $t \rightarrow 0$) has finally been generated from t whenever F classifies $\boldsymbol{x}$ as a positive (resp. negative) instance given θ. This rule r_U was added to R_U when it did not specialize a rule already in R_U, and the rules of R_U that specialize r_U were removed from R_U in order to ensure that the set R_U was simplified. The consistency of R_U is ensured because the explanations that have been produced in the process are faithful (majoritary explanations are abductive explanations for $\boldsymbol{x}$ given F and θ).

The rationale for the choices made is a follows. On the one hand, using a value for θ greater than the usual decision threshold (50%) considered for random forests was a way to generate a set of classification rules R_U with quite a good accuracy on the instances that are classified. On the other hand, using a reduced subset of instances for generating R_U was a way to limit the set of instances classified by R_U (thus, getting an incomplete set of classification rules, as expected).

The generation of AI was much more simple. AI simply is a single decision tree picked up uniformly at random from F, provided that its accuracy exceeds 50%. Doing so, as expected, the accuracy of AI before any interaction took place turned out to be lower than the initial accuracy of R_U, thus AI was at start a classifier less accurate than R_U, but a complete classifier (unlike R_U in general).

A domain theory Σ, having the form of a Krom formula and connecting the Boolean conditions used in R_U (thus, also those used in AI) has been considered whenever necessary. For example, if a numerical attribute $age \in A$ was used to describe instances and the Boolean conditions $x_1 = (age > 21)$ and $x_2 = (age > 18)$ occurred respectively in R_U and AI, Σ contained the clause $\overline{x_1} \vee x_2$.

The next step was to pick up at random instances in the set of instances triggering the interactions, and to take advantage of the XAI protocol presented in Section 3 to decide what to do with those instances (i.e., accept or reject the predictions made by AI), to modify AI and/or R_U accordingly, and to assess the performances of AI and R_U to determine how they evolve.

The performance of AI was evaluated by measuring empirically its accuracy on the test set. The performance of R_U was assessed from two perspectives: its accuracy (measured as well on the test set – of course, only the instances $\boldsymbol{x}$ of the test set for which $R_U(\boldsymbol{x})$ was defined have been considered in this evaluation) and its coverage (the proportion of instances $\boldsymbol{x}$ for which $R_U(\boldsymbol{x})$ was defined). To calculate the coverage of R_U we took advantage of the model counter D4 [20]: the number of models of Σ (the domain theory indicating how the Boolean conditions occurring in R are logically connected) is the total number of instances, the number of models of $\Sigma \wedge \bigvee_{t \rightarrow 1 \in R_U} t$ is the number of instances classified y R_U as positive, and the number of models of $\Sigma \wedge \bigvee_{t \rightarrow 0 \in R_U} t$ is the number of instances classified y R_U as negative. Thus, the coverage of R_U is given by:

$$\frac{\#(\Sigma \wedge \bigvee_{t \rightarrow 1 \in R_U} t) + \#(\Sigma \wedge \bigvee_{t \rightarrow 0 \in R_U} t)}{\#(\Sigma)}.$$

Since D4 accepts only CNF formulae as input, Tseitin's technique [31] is used to turn the DNF formulae $\bigvee_{t \rightarrow 1 \in R_U} t$ and $\bigvee_{t \rightarrow 0 \in R_U} t$ into the CNF format. This linear-time transformation is known not to change the number of models of the input. Experiments have been conducted on a computer equipped with Intel(R) XEON E5-2637 CPU @ 3.5 GHz and 128 Gib of memory.

Empirical results Interestingly, for every dataset used in the experiments, the computation times needed to achieve the interactions between AI and U were small enough. The time required per interaction step never exceeded 6.86 seconds. In average (over the triggering instances), it never exceeded 2 seconds and was greater than 0.1 second for 3 datasets only, out of 18 (namely arrowhead$_0$vs$_1$, arrowhead$_0$vs$_2$, and nerve$_0$vs$_2$). When rectifications were needed, most of the computation time was used to rectify AI.

Table 3 reports some statistics about the full interaction trace. Column $\#R$ indicates the number of rectifications of AI that have been performed. Column $\#G$ indicates the number of (strict) generalizations of rules from R_U that have been detected (remind that such generalizations may happen in Case (4)). Column "Case" indicates for each case from (1) to (4) the numbers of triggering instances falling into to the case. Finally, column "$I\#R_U$" gives the Initial number of rules in R_U ("+" indicates the number of rules concluding 1, while "-" indicates the number of rules concluding 0), and similarly for column "$F\#R_U$" about the Final number of rules in R_U.

Dataset	#R	#G	Case				I#R_U		F#R_U	
			(1)	(2)	(3)	(4)	+	-	+	-
arrowhead$_0$vs$_1$	207	0	0	11	14	0	8	9	8	9
arrowhead$_0$vs$_2$	112	7	0	4	10	11	12	4	9	4
arrowhead$_1$vs$_2$	88	0	0	7	15	0	5	10	5	10
australian	453	0	1	67	27	5	8	13	8	13
balance$_1$vs$_2$	128	0	0	47	25	28	5	4	5	4
biodegradation	144	10	0	22	14	64	3	9	3	5
breastTumor	48	38	0	0	12	38	0	6	0	5
cleveland	122	0	0	32	15	6	4	5	4	5
compas	0	100	0	0	0	100	2	1	1	1
contraceptive$_0$vs$_1$	18	24	3	0	3	94	6	0	6	0
contraceptive$_0$vs$_2$	118	12	0	8	73	19	5	4	5	4
contraceptive$_1$vs$_2$	0	0	0	0	0	100	0	6	0	6
divorce	64	0	0	22	5	2	3	4	3	4
nerve$_0$vs$_1$	2	13	0	0	1	13	0	3	0	3
nerve$_0$vs$_2$	24	12	0	0	6	12	0	6	0	5
nerve$_1$vs$_2$	14	0	0	0	14	9	1	1	1	1
spambase	236	0	0	79	20	1	48	84	48	84
wine	41	0	0	12	7	0	6	6	6	6

Table 3: Statistics about the interaction trace.

Table 4 shows how the accuracy of AI, the accuracy of R_U, and

[2] A decision tree equivalent to the negation of any $T_i \in F$ can be obtained in linear time in the size of T_i by replacing every 0-leaf of T_i by a 1-leaf and every 1-leaf of T_i by a 0-leaf. See [2] for more details.

the coverage of R_U evolved after a full sequence of interactions triggered by at most 100 instances. In Table 4, the first column "IAccAI" gives the Initial Accuracy of $\underline{AI}$, the second column "FAccAI" gives the Final Accuracy of $\underline{AI}$. Similarly, column "IAccU" gives the Initial Accuracy of $R_{\underline{U}}$, column "FAccU" gives the Final Accuracy of $R_{\underline{U}}$, column "ICU" gives the Initial Coverage of $R_{\underline{U}}$, and column "FCU" gives the Final Coverage of $R_{\underline{U}}$.

Dataset	IAccAI	FAccAI	IAccU	FAccU	ICU	FCU
arrowhead$_0$vs$_1$	0.731	0.750	0.929	0.929	1.550e-08	1.550e-08
arrowhead$_0$vs$_2$	0.596	0.923	1.000	0.953	9.308e-09	0.500
arrowhead$_1$vs$_2$	0.609	0.630	0.800	0.800	1.286e-07	1.286e-07
australian	0.702	0.723	0.909	0.909	0.122	0.122
balance$_1$vs$_2$	0.711	0.836	0.863	0.863	0.694	0.694
biodegradation	0.746	0.778	0.801	0.799	0.257	0.531
breastTumor	0.580	0.570	0.529	0.581	0.318	0.826
cleveland	0.651	0.783	0.892	0.892	0.328	0.328
compas	0.660	0.660	0.675	0.660	0.640	1.000
contraceptive$_0$vs$_1$	0.552	0.674	0.676	0.676	0.991	0.991
contraceptive$_0$vs$_2$	0.609	0.627	0.754	0.832	0.159	0.304
contraceptive$_1$vs$_2$	0.608	0.608	0.649	0.649	0.992	0.992
divorce	0.917	0.933	0.982	0.982	0.003	0.003
nerve$_0$vs$_1$	0.533	0.533	None	0.684	2.425e-16	0.500
nerve$_0$vs$_2$	0.632	0.632	None	0.719	2.852e-15	0.250
nerve$_1$vs$_2$	0.667	0.667	None	None	1.261e-13	1.261e-13
spambase	0.852	0.852	0.925	0.925	0.015	0.015
wine	0.487	0.487	0.500	0.500	2.302e-08	2.302e-08

Table 4: Evolution of the accuracy of AI, the accuracy of R_U, and the coverage of R_U. “None” means that no instance from the test set matched the condition part of a rule from R_U.

Table 3 shows that the most frequent cases encountered in the experiments deeply varies with the dataset at hand. Case (1) was the less frequent, which can be explained by the fact that the initial accuracy of AI was not low (anyway, it would not make sense to consider an AI with an accuracy not greater that 50%). The number $\#R$ of rectifications achieved (which may happen in cases (1) to (3)) greatly varied with the dataset. For 2 datasets out of 18, no rectification has been performed, but for many others the number of rectifications has been high (especially, greater than the number of triggering instances). Similarly, the number $\#G$ of (strict) generalizations made was significant for some datasets and null for other datasets.

Table 4 shows that the addition of rules made at case (4) may significantly increase the coverage of R_U. Even if, by design, the coverage of R_U may never diminish through the interactions with AI, the improvement in terms of the number of instances covered can be tremendous, as for the nerve$_0$vs$_1$ dataset. Initially, the coverage of R_U was very small for this dataset. Once all the instances used to trigger the interactions have been processed, the coverage of R_U was equal to $\frac{1}{2}$, showing that half of the instances can be classified by R_U at the end of the interactions.

Beyond the potential benefits in terms of coverage for R_U, Table 4 shows also that the rectifications made through the interactions between AI and U may lead to a valuable increase of the accuracy of AI. Overall, in our experiments, the accuracy of AI grew up for 10 datasets out of 18, did not change for 7 datasets, and diminished for a single dataset (breastTumor). A high number of rectifications may explain why the accuracy of AI has significantly increased for some datasets (see e.g., columns "IAccAI" and "FAccAI" for arrowhead$_0$vs$_2$ in Table 4), but it does not imply it (see e.g., columns "IAccAI" and "FAccAI" for australian in Table 4, where the increase in terms of accuracy was mild despite the high number of rectifications). Indeed, it can be the case that some rules of R_U used to rectify AI are actually incorrect.

Finally, comparing the values in columns "IAccU" and "FAccU" of Table 4, we can observe that the accuracy of R_U has not evolved for the majority of the datasets, that it has increased for a few datasets (breastTumor and contraceptive$_0$vs$_2$), and decreased for others (arrowhead$_0$vs$_2$, biodegradation, and compas). When it happens, the decrease of the accuracy of R_U should not be misinterpreted: it does not mean that the performance of R_U was degraded through the interactions with AI. Indeed, given the protocol used, the decrease simply results from the fact that the coverage of R_U has increased and that the accuracy of AI was initially lower than the accuracy of R_U. Especially, the instances classified at start by R_U remain classified in the same way by R_U at the end of the interaction process since R_U never is corrected (remind that one of the initial assumptions made was to consider that U is more reliable than AI on the instances that U is able to classify). The instances not classified by R_U at start and classified by R_U when all the triggering instances have been considered are classified by R_U as demanded by AI. Since the accuracy of AI is lower than the one of R_U, the risk of a classification error made by R_U for the instances for which U listens to AI increases each time such an instance is considered, and this explains why the overall accuracy of R_U may decrease.

To wrap up with the empirical results, it turns out that taking advantage of the proposed XAI protocol has led to increase the accuracy of AI for 10 datasets out of 18. For 3 datasets among the 10 (namely, arrowhead$_0$vs$_2$, biodegradation, and contraceptive$_0$vs$_2$), the coverage of R_U has also increased. For 3 of the 8 remaining datasets (namely, compas, nerve$_0$vs$_1$, and nerve$_0$vs$_2$), the accuracy of AI has remained unchanged but the coverage of R_U has increased. For breastTumor, the accuracy of AI has decreased but the coverage of R_U has increased. Finally, for 4 datasets among those considered in our experiments (namely, contraceptive$_1$vs$_2$, nerve$_1$vs$_2$, spambase, and wine), the interactions made did not lead to increase either the accuracy of AI or the coverage of R_U.

5 Conclusion

In this paper, we have presented an XAI protocol that can be leveraged for ruling interactions between a user U and a predictor AI used by U. In our setting, the pieces of knowledge about the prediction task that are owned by U is supposed to be representable by a set of classification rules, that is assumed reliable and consistent, but (in general) incomplete. AI is supposed to be any tree-based predictor (a decision tree, a random forest, or a boosted tree). The interactions made between U and AI are intended not only help U decide what to do with each prediction achieved by AI (accept it, reject it), but could also be exploited to improve the quality of the predictions made by AI by correcting those that are wrong according to U, and to augment the coverage of U, i.e., the proportion of instances U is able to handle. We took advantage of the explanation and correction abilities furnished by the PyXAI library (github.com/crillab/pyxai) to implement and evaluate the proposed protocol. Experiments have been conducted, showing the benefits that can be achieved in practice by taking advantage of this protocol.

As a next step, it would be interesting to evaluate the proposed XAI protocol in practice, with an AI system represented by a random forest or a boosted tree, and a human user U instead of an artificial agent. In this perspective, it would be useful to extend the explanation and rectification settings to deal with uncertain classification rules and with classifiers based on scorers and decision thresholds. That way, a rectification of AI could be triggered only when the classification rule used by U to classify $\boldsymbol{x}$ is more certain than the prediction about $\boldsymbol{x}$ that is achieved by AI. Taking the uncertainty of the predictions achieved by AI into account would also lead to the definitions of other policies to address case (4), depending on the accuracy / coverage trade-off that is expected for U.

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) and of the France 2030 MAIA Project (ANR-22-EXES-0009) 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. B. Arrieta, N. Díaz, 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.

[2] G. Audemard, S. Bellart, L. Bounia, F. Koriche, J.-M. Lagniez, and P. Marquis. Trading complexity for sparsity in random forest explanations. In *Proc. of AAAI'22*, pages 5461–5469, 2022.

[3] G. Audemard, S. Coste-Marquis, P. Marquis, M. Sabiri, and N. Szczepanski. Code and data for "Designing an XAI Interface for Tree-Based ML Models", 2024. Available at www.cril.fr/expekctation/.

[4] P. Barceló, M. Monet, J. Pérez, and B. Subercaseaux. Model interpretability through the lens of computational complexity. In *Proc. of NeurIPS'20*, 2020.

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

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

[7] 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.

[8] S. Coste-Marquis and P. Marquis. On belief change for multi-label classifier encodings. In *Proc. of IJCAI'21*, pages 1829–1836, 2021.

[9] S. Coste-Marquis and P. Marquis. Rectifying binary classifiers. In *Proc. of ECAI'23*, pages 485–492, 2023.

[10] A. Darwiche and A. Hirth. On the reasons behind decisions. In *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI'20)*, pages 712–720, 2020.

[11] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55 (1):119–139, 1997.

[12] N. Gorji and S. Rubin. Sufficient reasons for classifier decisions in the presence of domain constraints. In *Proc. of AAAI'22*, pages 5660–5667, 2022.

[13] R. Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, 2022. URL https://doi.org/10.1007/s10618-022-00831-6.

[14] 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.

[15] D. Gunning. DARPA's explainable artificial intelligence (XAI) program. In *Proc. of IUI'19*, 2019.

[16] A. Ignatiev. Towards trustable explainable ai. In *Proc. of IJCAI'20*, pages 5154–5158, 2020.

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

[18] A. Ignatiev, N. Narodytska, and J. Marques-Silva. On validating, repairing and refining heuristic ML explanations. *CoRR*, abs/1907.02509, 2019. URL http://arxiv.org/abs/1907.02509.

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

[20] J.-M. Lagniez and P. Marquis. An improved decision-dnnf compiler. In *Proc. of IJCAI'17*, pages 667–673, 2017.

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

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

[23] 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.

[24] 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.

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

[26] G. Ras, N. Xie, M. van Gerven, and D. Doran. Explainable deep learning: A field guide for the uninitiated. *J. Artif. Intell. Res.*, 73:329–396, 2022.

[27] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

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

[29] 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.

[30] A. Shih, A. Choi, and A. Darwiche. A symbolic approach to explaining bayesian network classifiers. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI'18)*, pages 5103–5111, 2018.

[31] 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.

[32] G. Vilone and L. Longo. Notions of explainability and evaluation approaches for explainable artificial intelligence. *Inf. Fusion*, 76:89–106, 2021.

[33] J. Zhou, A. Gandomi, F. Chen, and A. Holzinger. Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, 10:593, 2021.