A New Class of Explanations for Classifiers with Non-binary Features

Chunxi Ji(✉) and Adnan Darwiche

University of California, Los Angeles, Los Angeles, CA 90095, USA
{jich,darwiche}@cs.ucla.edu

Abstract. Two types of explanations have been receiving increased attention in the literature when analyzing the decisions made by classifiers. The first type explains why a decision was made and is known as a sufficient reason for the decision, also an abductive explanation or a PI-explanation. The second type explains why some other decision was not made and is known as a necessary reason for the decision, also a contrastive or counterfactual explanation. These explanations were defined for classifiers with binary, discrete and, in some cases, continuous features. We show that these explanations can be significantly improved in the presence of non-binary features, leading to a new class of explanations that relay more information about decisions and the underlying classifiers. Necessary and sufficient reasons were also shown to be the prime implicates and implicants of the complete reason for a decision, which can be obtained using a quantification operator. We show that our improved notions of necessary and sufficient reasons are also prime implicates and implicants but for an improved notion of complete reason obtained by a new quantification operator that we also define and study.

Keywords: Explainable AI · Decision Graphs · Prime Implicants/Implicates

1 Introduction

Explaining the decisions of classifiers has been receiving significant attention in the AI literature recently. Some explanation methods operate directly on classifiers, e.g., [43,44], while some other methods operate on symbolic encodings of their input-output behavior, e.g., [8,25,37,40], which may be compiled into tractable circuits [5,11,21,45–47]. When explaining the decisions of classifiers, two particular notions have been receiving increased attention in the literature: The sufficient and necessary reasons for a decision on an instance.

A *sufficient reason* for a decision [17] is a minimal subset of the instance which is guaranteed to trigger the decision. It was first introduced under the name *PI-explanation* in [46] and later called an *abductive explanation* [25].[1] Consider the classifier in Fig. 1a and a patient, Susan, with the following characteristics: Age $\geq$55, Btype=A and Weight=over. Susan is judged as susceptible to disease

[1] We will use sufficient reasons and PI/abductive explanations interchangeably.

S. Gaggl et al. (Eds.): JELIA 2023, LNAI 14281, pp. 106–122, 2023.
https://doi.org/10.1007/978-3-031-43619-2_8

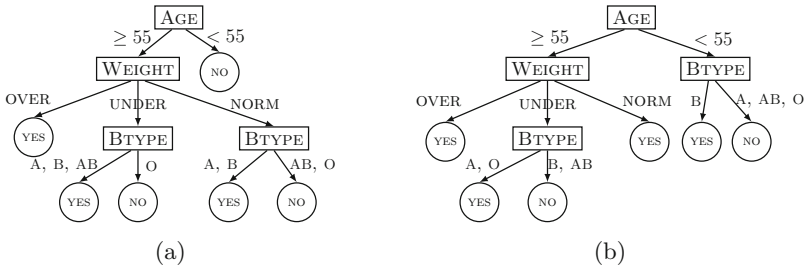


Fig. 1. Two classifiers of patients susceptible to a certain disease. The classifier in (b) will be discussed later in the paper.

by this classifier, and a sufficient reason for this decision is {AGE ≥55, BTYPE=A}. Hence, the classifier will judge Susan as susceptible to disease as long as she has these two characteristics, regardless of how the feature WEIGHT is set.[2]

A *necessary reason* for a decision [18] is a minimal subset of the instance that will flip the decision if changed appropriately. It was formalized earlier in [24] under the name *contrastive explanation* which is discussed initially in [33,39].[3] Consider again the patient Susan and the classifier in Fig. 1a. A necessary reason for the decision on Susan is {AGE ≥55}, which means that she would not be judged as susceptible to disease if she were younger than 55. The other necessary reason is {WEIGHT=OVER, BTYPE=A} so the decision on Susan can be flipped by changing these two characteristics (and this cannot be achieved by changing only one of them). Indeed, if Susan had WEIGHT=NORM and BTYPE=AB, she will not be judged as susceptible. However, since WEIGHT and BTYPE are discrete variables, there are multiple ways for changing them and some changes may not flip the decision (e.g., WEIGHT=UNDER and BTYPE=B).

The notion of a *complete reason* behind a decision was introduced in [17] and its prime implicants were shown to be the sufficient reasons for the decision. Intuitively, the complete reason is a particular condition on the instance that is both necessary and sufficient for the decision on that instance; see [16]. A declarative semantics for complete reasons was given in [19] which showed how to compute them using *universal literal quantification*. Furthermore, the prime implicates of a complete reason where shown to be the necessary reasons for the decision in [18]. Given these results, one would first use universal literal quantification to obtain the complete reason for a decision and then compute its prime implicates and implicants to obtain necessary and sufficient explanations.

[2] See, e.g., [13,44,49] for some approaches that can be viewed as approximating sufficient reasons and [26] for a study of the quality of some of these approximations.

[3] We will use necessary reasons and contrastive explanations interchangeably in this paper. Counterfactual explanations are related but have alternate definitions in the literature. For example, as defined in [5], they correspond to length-minimal necessary reasons; see [18]. But according to some other definitions, they include contrastive explanations (necessary reasons) as a special case; see Sect. 5.2 in [34]. See also [1] for counterfactual explanations that are directed towards Bayesian network classifiers and [2] for a relevant recent study and survey.

Necessary and sufficient reasons are *subsets* of the instance being explained so each reason corresponds to a set of variable settings (Feature=Value), like Weight=under and Btype=B, which we shall call *simple literals.* Since necessary and sufficient reasons correspond to sets of simple literals, we will refer to them as *simple* or *classical* explanations. We will show next that these simple explanations can be significantly improved if the classifier has non-binary features, leading to more general notions of necessary, sufficient and complete reasons that provide more informative explanations of decisions.

Consider again the decision on Susan discussed above which had the sufficient reason $\{\text{Age} \geq 55, \text{Btype}=\text{A}\}$. Such an explanation can be viewed as a *property* of the instance which guarantees the decision. The property has a specific form: a conjunction of feature settings (i.e., instance characteristics) which leaves out characteristics of the instance that are irrelevant to the decision (Weight=over). However, the following is a weaker property of the instance which will also trigger the decision: $\{\text{Age} \geq 55, \text{Btype} \in \{\text{A}, \text{B}\}\}$. This property tells us that not only is Weight=over irrelevant to the decision, but also that Btype=A is not particularly relevant since Btype could have been B and the decision would have still been triggered. In other words, what is really relevant is that $\text{Btype} \in \{\text{A}, \text{B}\}$ or, alternatively, $\text{Btype} \notin \{\text{AB}, \text{O}\}$. Clearly, this kind of explanation reveals more information about why the classifier made its decision. We will later formalize and study a new class of explanations for this purpose, called *general sufficient reasons,* which arise only when the classifier has non-binary features.

A necessary reason for a decision can also be understood as a property of the instance, but one that will flip the decision if violated in a *certain* manner [18]. As mentioned earlier, $\{\text{Weight}=\text{over}, \text{Btype}=\text{A}\}$ is a necessary reason for the decision on Susan. This reason corresponds to the property (Weight=over or Btype=A). We can flip the decision by violating this property through changing the values of Weight and Btype in the instance. Since these variables are non-binary, there are multiple changes (six total) that will violate the property. Some violations will flip the decision, others will not (we are only guaranteed that at least one violation will flip the decision). For example, Weight=norm, Btype=O and Weight=under, Btype=AB will both violate the property but only the first one will flip the decision. However, the following weaker property is guaranteed to flip the decision regardless of how it is violated: (Weight=over or $\text{Btype} \in \{\text{A}, \text{B}, \text{AB}\}$). We can violate this property using two different settings of Weight and Btype, both of which will flip the decision. This property corresponds to the *general necessary reason* $\{\text{Weight}=\text{over}, \text{Btype} \in \{\text{A}, \text{B}, \text{AB}\}\}$, a new notion that we introduce and study later. Similar to general sufficient reasons, general necessary reasons provide more information about the behavior of a classifier and arise only when the classifier has non-binary features.

We stress here that using simple explanations in the presence of non-binary features is quite prevalent in the literature; see, e.g., [4,6,8,18,23,28,36]. Two notable exceptions are [12,27] which we discuss in more detail later.[4]

[4] Interestingly, the axiomatic study of explanations in [3] allows non-binary features, yet Axiom 4 (*feasibility*) implies that explanations must be simple.

Our study of general necessary and sufficient reasons follows a similar structure to recent developments on classical necessary and sufficient reasons. In particular, we define a new quantification operator like the one defined in [19] and show how it can be used to compute the *general reason* of a decision, and that its prime implicates and implicants contain the general necessary and sufficient reasons. Complete reasons are known to be monotone formulas. We show that general reasons are *fixated formulas* which include monotone ones. We introduce the fixation property and discuss some of its (computational) implications.

This paper is structured as follows. We start in Sect. 2 by discussing the syntax and semantics of formulas with discrete variables which are needed to capture the input-output behavior of classifiers with non-binary features. We then introduce the new quantification operator in Sect. 3 where we study its properties and show how it can be used to formulate the new notion of general reason. The study of general necessary and sufficient reasons is conducted in Sect. 4 where we also relate them to their classical counterparts and argue further for their utility. Section 5 provides closed-form general reasons for a broad class of classifiers and Sect. 6 discusses the computation of general necessary and sufficient reasons based on general reasons. We finally close with some remarks in Sect. 7. Proofs of all results are in Appendix A of [30].

2 Representing Classifiers Using Class Formulas

We now discuss the syntax and semantics of *discrete formulas*, which we use to represent the input-output behavior of classifiers. Such symbolic formulas can be automatically compiled from certain classifiers, like Bayesian networks, random forests and some types of neural networks; see [16] for a summary.

We assume a finite set of variables Σ which represent classifier features. Each variable $X \in \Sigma$ has a finite number of *states* $x_1, \ldots, x_n$, $n > 1$. A *literal* ℓ for variable X, called X-literal, is a set of states such that $\emptyset \subset \ell \subset \{x_1, \ldots, x_n\}$. We will often denote a literal such as $\{x_1, x_3, x_4\}$ by x_{134} which reads: the state of variable X is either x_1 or x_3 or x_4. A literal is *simple* iff it contains a single state. Hence, x_3 is a simple literal but x_{134} is not. Since a simple literal corresponds to a state, these two notions are interchangeable.

A *formula* is either a constant $\top$, $\bot$, literal ℓ, negation $\overline{\alpha}$, conjunction $\alpha \cdot \beta$ or disjunction $\alpha + \beta$ where α, β are formulas. The set of variables appearing in a formula Δ are denoted by $vars(\Delta)$. A *term* is a conjunction of literals for distinct variables. A *clause* is a disjunction of literals for distinct variables. A *DNF* is a disjunction of terms. A *CNF* is a conjunction of clauses. An *NNF* is a formula without negations. These definitions imply that terms cannot be inconsistent, clauses cannot be valid, and negations are not allowed in DNFs, CNFs, or NNFs. Finally, we say a term/clause is *simple* iff it contains only simple literals.

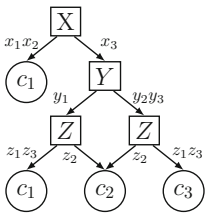
A *world* maps each variable in Σ to one of its states and is typically denoted by ω. A world ω is called a *model* of formula α, written $\omega \models \alpha$, iff α is satisfied by ω (that is, α is true at ω). The constant $\top$ denotes a valid formula (satisfied by every world) and the constant $\bot$ denotes an unsatisfiable formula (has no

models). Formula α implies formula β, written $\alpha \models \beta$, iff every model of α is also a model of β. A term τ_1 subsumes another term τ_2 iff $\tau_2 \models \tau_1$. A clause σ_1 subsumes another clause σ_2 iff $\sigma_1 \models \sigma_2$. Formula α is weaker than formula β iff $\beta \models \alpha$ (hence β is stronger than α).

The *conditioning* of formula Δ on simple term τ is denoted $\Delta|\tau$ and obtained as follows. For each state x of variable X that appears in term τ, replace each X-literal ℓ in Δ with $\top$ if $x \in \ell$ and with $\bot$ otherwise. Note that $\Delta|\tau$ does not mention any variable that appears in term τ. A *prime implicant* for a formula Δ is a term α such that $\alpha \models \Delta$, and there does not exist a distinct term β such that $\alpha \models \beta \models \Delta$. A *prime implicate* for a formula Δ is a clause α such that $\Delta \models \alpha$, and there does not exist a distinct clause β such that $\Delta \models \beta \models \alpha$.

An *instance* of a classifier will be represented by a simple term which contains exactly one literal for each variable in Σ. A classifier with n classes will be represented by a set of mutually exclusive and exhaustive formulas $\Delta^1, \ldots, \Delta^n$, where the models of formula Δ^i capture the instances in the i^{th} class. That is, instance $\mathcal{I}$ is in the i^{th} class iff $\mathcal{I} \models \Delta^i$. We refer to each Δ^i as a *class formula,* or simply a *class,* and say that instance $\mathcal{I}$ is in class Δ^i when $\mathcal{I} \models \Delta^i$.

Consider the decision diagram on the right which represents a classifier with three ternary features (X, Y, Z) and three classes c_1, c_2, and c_3. This classifier can be represented by the class formulas $\Delta^1 = x_{12} + x_3 \cdot y_1 \cdot z_{13}$, $\Delta^2 = x_3 \cdot z_2$ and $\Delta^3 = x_3 \cdot y_{23} \cdot z_{13}$. This classifier has 27 instances, partitioned as follows: 20 instances in class c_1, 3 in class c_2 and 4 in class c_3. For example, instance $\mathcal{I} = x_3 \cdot y_2 \cdot z_2$ belongs to class c_2 since $\mathcal{I} \models \Delta^2$.



3 The General Reason for a Decision

An operator $\forall x$ which eliminates the state x of a Boolean variable X from a formula was introduced and studied in [19]. This operator, called universal literal quantification, was also generalized in [19] to the states of discrete variables but without further study. Later, [18] studied this discrete generalization, given next.

Definition 1. *For variable X with states $x_1, \ldots, x_n$, the universal literal quantification of state x_i from formula Δ is defined as* $\forall x_i \cdot \Delta = \Delta|x_i \cdot \prod_{j \neq i}(x_i + \Delta|x_j)$.

The operator $\forall$ is commutative so we can equivalently write $\forall x \cdot (\forall y \cdot \Delta)$, $\forall y \cdot (\forall x \cdot \Delta)$, $\forall x, y \cdot \Delta$ or $\forall\{x, y\} \cdot \Delta$. It is meaningful then to quantify an instance $\mathcal{I}$ from its class formula Δ since $\mathcal{I}$ is a set of states. As shown in [19], the quantified formula $\forall \mathcal{I} \cdot \Delta$ corresponds to the complete reason for the decision on instance $\mathcal{I}$. Hence, the prime implicants of $\forall \mathcal{I} \cdot \Delta$ are the sufficient reasons for the decision [17] and its prime implicates are the necessary reasons [18].

We next define a new operator $\overline{\forall}$ that we call a *selection operator* for reasons that will become apparent later. This operator will lead to the notion of a general reason for a decision which subsumes the decision's complete reason, and provides the basis for defining general necessary and sufficient reasons.

Definition 2. *For variable X with states $x_1, \ldots, x_n$ and formula Δ, we define $\overline{\forall} x_i \cdot \Delta$ to be $\Delta|x_i \cdot \Delta$.*

The selection operator $\overline{\forall}$ is also commutative, like $\forall$.

Proposition 1. *$\overline{\forall} x \cdot (\overline{\forall} y \cdot \Delta) = \overline{\forall} y \cdot (\overline{\forall} x \cdot \Delta)$ for states x, y.*

Since a term τ corresponds to a set of states, the expression $\overline{\forall} \tau \cdot \Delta$ is well-defined just like $\forall \tau \cdot \Delta$. We can now define our first major notion.

Definition 3. *Let $\mathcal{I}$ be an instance in class Δ. The general reason for the decision on instance $\mathcal{I}$ is defined as $\overline{\forall} \mathcal{I} \cdot \Delta$.*

The complete reason $\forall \mathcal{I} \cdot \Delta$ can be thought of as a property/abstraction of instance $\mathcal{I}$ that justifies (i.e., can trigger) the decision. In fact, it is equivalent to the weakest NNF Γ whose literals appear in the instance and that satisfies $\mathcal{I} \models \Gamma \models \Delta$ [18,19]. The next result shows that the general reason is a weaker property and, hence, a further abstraction that triggers the decision.

Proposition 2. *For instance $\mathcal{I}$ and formula Δ where $\mathcal{I} \models \Delta$, we have $\mathcal{I} \models \forall \mathcal{I} \cdot \Delta \models \overline{\forall} \mathcal{I} \cdot \Delta \models \Delta$. ($\mathcal{I} \not\models \Delta$ only if $\forall \mathcal{I} \cdot \Delta = \overline{\forall} \mathcal{I} \cdot \Delta = \bot$)*

The next result provides further semantics for the general reason and highlights the key difference with the complete reason.

Proposition 3. *The general reason $\overline{\forall} \mathcal{I} \cdot \Delta$ is equivalent to the weakest NNF Γ whose literals are implied by instance $\mathcal{I}$ and that satisfies $\mathcal{I} \models \Gamma \models \Delta$.*

The complete and general reasons are abstractions of the instance that explain why it belongs to its class. The former can only reference simple literals in the instance but the latter can reference any literal that is implied by the instance. The complete reason can be recovered from the general reason and the underlying instance. Moreover, the two types of reasons are equivalent when all variables are binary since $\forall x \cdot \Delta = \overline{\forall} x \cdot \Delta$ when x is the state of a binary variable.

We next provide a number of results that further our understanding of general reasons, particularly their semantics and how to compute them. We start with the following alternative definition of the operator $\overline{\forall} x_i$.

Proposition 4. *For formula Δ and variable X with states $x_1, \ldots, x_n$, $\overline{\forall} x_i \cdot \Delta$ is equivalent to $(\Delta|x_i) \cdot \prod_{j \neq i} (\ell_j + (\Delta|x_j))$, where ℓ_j is the literal $\{x_1, \ldots, x_n\} \setminus \{x_j\}$.*

According to this definition, we can always express $\overline{\forall} x_i \cdot \Delta$ as an NNF in which every X-literal includes state x_i (recall that $\Delta|x_i$ and $\Delta|x_j$ do not mention variable X). This property is used in the proofs and has a number of implications.[5]

[5] For example, we can use it to provide *forgetting* semantics for the dual operator $\overline{\exists} x_i \cdot \Delta = \overline{\overline{\forall} x_i \cdot \overline{\Delta}}$. Using Definition 2, we get $\overline{\exists} x_i \cdot \Delta = \Delta + \Delta|x_i$. Using Proposition 4, we get $\overline{\exists} x_i \cdot \Delta = \Delta|x_i + \sum_{j \neq i} (x_j \cdot \Delta|x_j)$. We can now easily show that (1) $\Delta \models \overline{\exists} x_i \cdot \Delta$ and (2) $\overline{\exists} x_i \cdot \Delta$ is equivalent to an NNF whose X-literals do not mention state x_i. That is, $\overline{\exists} x_i$ can be understood as forgetting the information about state x_i from Δ. This is similar to the dual operator $\exists x_i \cdot \Delta = \overline{\forall x_i \cdot \overline{\Delta}}$ studied in [19,32] except that $\overline{\exists} x_i$ erases less information from Δ since one can show that $\Delta \models \overline{\exists} x_i \cdot \Delta \models \exists x_i \cdot \Delta$.

When Δ is a class formula, [19] showed that the application of $\forall x$ to Δ can be understood as *selecting* a specific set of instances from the corresponding class. This was shown for states x of Boolean variables. We next generalize this to discrete variables and provide a selection semantics for the new operator $\overline{\forall}$.

Proposition 5. *Let τ be a simple term, Δ be a formula and ω be a world. Then $\omega \models \forall \tau \cdot \Delta$ iff $\omega \models \Delta$ and $\omega' \models \Delta$ for any world ω' obtained from ω by changing the states of some variables that are set differently in τ. Moreover, $\omega \models \overline{\forall} \tau \cdot \Delta$ iff $\omega \models \Delta$ and $\omega' \models \Delta$ for any world ω' obtained from ω by setting some variables in ω to their states in τ.*

That is, $\forall \tau \cdot \Delta$ selects all instances in class Δ whose membership in the class does not depend on characteristics that are inconsistent with τ. These instances are also selected by $\overline{\forall} \tau \cdot \Delta$ which further selects instances that remain in class Δ when any of their characteristics are changed to agree with τ.

The complete reason is monotone which has key computational implications as shown in [17–19]. The general reason satisfies a weaker property called *fixation* which has also key computational implications as we show in Sect. 6.

Definition 4. *An NNF is locally fixated on instance $\mathcal{I}$ iff its literals are consistent with $\mathcal{I}$. A formula is fixated on instance $\mathcal{I}$ iff it is equivalent to an NNF that is locally fixated on $\mathcal{I}$.*

We also say in this case that the formula is $\mathcal{I}$-fixated. For example, if $\mathcal{I} = x_1 \cdot y_1 \cdot z_2$ then the formula $x_{12} \cdot y_1 + z_2$ is (locally) $\mathcal{I}$-fixated but $x_{12} \cdot z_1$ is not. By the selection semantic we discussed earlier, a formula Δ is $\mathcal{I}$-fixated only if for every model ω of Δ, changing the states of some variables in ω to their states in $\mathcal{I}$ guarantees that the result remains a model of Δ. Moreover, if Δ is $\mathcal{I}$-fixated, then $\mathcal{I} \models \Delta$ but the opposite does not hold (e.g., $\Delta = x_1 + y_1$ and $\mathcal{I} = x_1 \cdot y_2$). We now have the following corollary of Proposition 3.

Corollary 1. *The general reason $\overline{\forall} \mathcal{I} \cdot \Delta$ is $\mathcal{I}$-fixated.*

The next propositions show that the new operator $\overline{\forall}$ has similar computational properties to $\forall$ which we use in Sect. 5 to compute general reasons.

Proposition 6. *For state x and literal ℓ of variable X, $\overline{\forall} x \cdot \ell = \ell$ if $x \in \ell$ ($x \models \ell$); else $\overline{\forall} x \cdot \ell = \bot$. Moreover, $\overline{\forall} x \cdot \Delta = \Delta$ if X does not appear in Δ.*

Proposition 7. *For formulas α, β and state x_i of variable X, we have $\overline{\forall} x_i \cdot (\alpha \cdot \beta) = (\overline{\forall} x_i \cdot \alpha) \cdot (\overline{\forall} x_i \cdot \beta)$. Moreover, if variable X does not occur in both α and β, then $\overline{\forall} x_i \cdot (\alpha + \beta) = (\overline{\forall} x_i \cdot \alpha) + (\overline{\forall} x_i \cdot \beta)$.*

An NNF is $\vee$-decomposable if its disjuncts do not share variables. According to these propositions, we can apply $\overline{\forall} \mathcal{I}$ to an $\vee$-decomposable NNF in linear time, by simply applying $\overline{\forall} \mathcal{I}$ to each literal in the NNF (the result is $\vee$-decomposable).

4 General Necessary and Sufficient Reasons

We next introduce generalizations of necessary and sufficient reasons and show that they are prime implicates and implicants of the general reason for a decision. These new notions have more explanatory power and subsume their classical counterparts, particularly when explaining the behavior of a classifier beyond a specific instance/decision. For example, when considering the classifier in Fig. 1b, which is a variant of the one in Fig. 1a, we will see that the two classifiers will make identical decisions on some instances, leading to identical simple necessary and sufficient reasons for these decisions but distinct general necessary and sufficient reasons. Moreover, we will see that general necessary and sufficient reasons are particularly critical when explaining the behavior of classifiers with (discretized) numeric features.

4.1 General Sufficient Reasons (GSRs)

We start by defining the classical notion of a (simple) sufficient reason but using a different formulation than [46] which was the first to introduce this notion under the name of a PI-explanation. Our formulation is meant to highlight a symmetry with the proposed generalization.

Definition 5 (SR). *A sufficient reason for the decision on instance $\mathcal{I}$ in class Δ is a weakest simple term τ s.t. $\mathcal{I} \models \tau \models \Delta$.*

This definition implies that each literal in τ is a variable setting (i.e., characteristic) that appears in instance $\mathcal{I}$. That is, the (simple) literals of sufficient reason τ are a subset of the literals in instance $\mathcal{I}$. We now define our generalization.

Definition 6 (GSR). *A general sufficient reason for the decision on instance $\mathcal{I}$ in class Δ is a term τ which satisfies (1) τ is a weakest term s.t. $\mathcal{I} \models \tau \models \Delta$ and (2) no term τ' satisfies the previous condition if $vars(\tau') \subset vars(\tau)$.*

This definition does not require the GSR τ to be a simple term, but it requires that it has a minimal set of variables. Without this minimality condition, a GSR will be redundant in the sense of the upcoming Proposition 8. For a term τ and instance $\mathcal{I}$ s.t. $\mathcal{I} \models \tau$, we will use $\mathcal{I} \dot{\cap} \tau$ to denote the smallest subterm in $\mathcal{I}$ that implies τ. For example, if $\mathcal{I} = x_2 \cdot y_1 \cdot z_3$ and $\tau = x_{12} \cdot y_{13}$, then $\mathcal{I} \dot{\cap} \tau = x_2 \cdot y_1$.

Proposition 8. *Let $\mathcal{I}$ be an instance in class Δ and τ be a weakest term s.t. $\mathcal{I} \models \tau \models \Delta$. If τ' is a weakest term s.t. $\mathcal{I} \models \tau' \models \Delta$ and $vars(\tau') \subset vars(\tau)$, then $\mathcal{I} \dot{\cap} \tau \models \mathcal{I} \dot{\cap} \tau' \models \Delta$. Also, $\mathcal{I} \dot{\cap} \tau$ is a SR iff such a term τ' does not exist.*

According to this proposition, the term τ is redundant as an explanation in that the subset of instance $\mathcal{I}$ which it identifies as being a culprit for the decision ($\mathcal{I} \dot{\cap} \tau$) is dominated by a smaller subset that is identified by the term τ' ($\mathcal{I} \dot{\cap} \tau'$).

Consider the classifiers in Figs. 1a and 1b and the patient Susan: Age $\geq$55, Btype=A and Weight=over. Both classifiers will make the same decision yes on

Susan with the same SRs: (Age $\geq$55 $\cdot$ Btype=A) and (Age $\geq$55 $\cdot$ Weight=over). The GSRs are different for these two (equal) decisions. For the first classifier, they are (Age $\geq$55 $\cdot$ Btype $\in\{$A, B$\}$) and (Age $\geq$55 $\cdot$ Weight=over). For the second, they are (Age $\geq$55 $\cdot$ Btype $\in\{$A, O$\}$) and (Age $\geq$55 $\cdot$ Weight $\in\{$over, norm$\}$). GSRs encode all SRs and contain more information.[6]

Proposition 9. *Let τ be a simple term. Then τ is a SR for the decision on instance $\mathcal{I}$ iff $\tau = \mathcal{I} \dot{\cap} \tau'$ for some GSR τ'.*

Consider the instance Susan again, $\mathcal{I}$ = (Age $\geq$55)$\cdot$(Btype=A)$\cdot$(Weight=over) and the classifier in Fig. 1b. As mentioned, the GSRs for the decision on Susan are τ_1' = (Age $\geq$55 $\cdot$ Btype $\in\{$A, O$\}$) and τ_2' = (Age $\geq$55 $\cdot$ Weight $\in\{$over, norm$\}$) so $\tau_1 = \mathcal{I} \dot{\cap} \tau_1'$ = (Age $\geq$55$\cdot$Btype=A) and $\tau_2 = \mathcal{I} \dot{\cap} \tau_2'$ = (Age $\geq$55$\cdot$Weight=over), which are the two SRs for the decision on Susan.

The use of general terms to explain the decision on an instance $\mathcal{I}$ in class Δ was first suggested in [12]. This work proposed the notion of a general PI-explanation as a prime implicant of Δ that is consistent with instance $\mathcal{I}$. This definition is equivalent to Condition (1) in our Definition 6 which has a second condition relating to variable minimality. Hence, the definition proposed by [12] does not satisfy the desirable properties stated in Propositions 8 and 9 which require this minimality condition. The merits of using general terms were also discussed when explaining decision trees in [27], which introduced the notion of an *abductive path explanation (APXp).* In a nutshell, each path in a decision tree corresponds to a general term τ that implies the formula Δ of the path's class. Such a term is usually used to explain the decisions made on instances that follow that path. As observed in [27], such a term can often be shortened, leading to an APXp that still implies the class formula Δ and hence provides a better explanation. An APXp is an implicant of the class formula Δ but not necessarily a prime implicant (or a variable-minimal prime implicant). Moreover, an APXp is a property of the specific decision tree (syntax) instead of its underlying classifier (semantics). See Appendix B in [30] for further discussion of these limitations.[7]

4.2 General Necessary Reasons (GNRs)

We now turn to simple necessary reasons and their generalizations. A necessary reason is a property of the instance that will flip the decision if violated in a certain way (by changing the instance). As mentioned earlier, the difference between the classical necessary reason and the generalized one is that the latter comes with stronger guarantees. Again, we start with a definition of classical necessary reasons using a different phrasing than [24] which formalized them under the name of contrastive explanations [33]. Our phrasing, based on [18], highlights a symmetry with the generalization and requires the following notation.

[6] Unlike SRs, two GSRs may mention the same set of variables. Consider the class formula $\Delta = (x_1 \cdot y_{12}) + (x_{12} \cdot y_1)$ and instance $\mathcal{I} = x_1 \cdot y_1$. There are two GSRs for the decision on $\mathcal{I}$, $x_1 \cdot y_{12}$ and $x_{12} \cdot y_1$, and both mention the same variables X, Y.

[7] A dual notion, contrastive path explanation (CPXp), was also proposed in [27].

For a clause σ and instance $\mathcal{I}$ s.t. $\mathcal{I} \models \sigma$, we will use $\mathcal{I} \backslash\backslash \sigma$ to denote the largest subterm of $\mathcal{I}$ that does not imply σ. For example, if $\mathcal{I} = x_2 \cdot y_1 \cdot z_3$ and $\sigma = x_{12} + y_{13}$ then $\mathcal{I} \backslash\backslash \sigma = z_3$. We will also write $\mathcal{I} \dot{\models} \sigma$ to mean that instance $\mathcal{I}$ implies every literal in clause σ. For instance $\mathcal{I} = x_2 \cdot y_1 \cdot z_3$, we have $\mathcal{I} \dot{\models} x_{12} + y_{13}$ but $\mathcal{I} \not\dot{\models} x_{12} + y_{23}$ even though $\mathcal{I} \models x_{12} + y_{23}$.

Definition 7 (NR). *A necessary reason for the decision on instance $\mathcal{I}$ in class Δ is a strongest simple clause σ s.t. $\mathcal{I} \dot{\models} \sigma$ and $(\mathcal{I} \backslash\backslash \sigma) \cdot \overline{\sigma} \not\models \Delta$ (if we minimally change the instance to violate σ, it is no longer guaranteed to stay in class Δ).*

A necessary reason guarantees that *some* minimal change to the instance which violates the reason will flip the decision. But it does not guarantee that *all* such changes will. A general necessary reason comes with a stronger guarantee.

Definition 8 (GNR). *A general necessary reason for the decision on instance $\mathcal{I}$ in class Δ is a strongest clause σ s.t. $\mathcal{I} \dot{\models} \sigma$, $(\mathcal{I} \backslash\backslash \sigma) \cdot \overline{\sigma} \models \overline{\Delta}$, and no clause σ' satisfies the previous conditions if $vars(\sigma') \subset vars(\sigma)$.*

The key difference between Definitions 7 and 8 are the conditions $(\mathcal{I} \backslash\backslash \sigma) \cdot \overline{\sigma} \not\models \Delta$ and $(\mathcal{I} \backslash\backslash \sigma) \cdot \overline{\sigma} \models \overline{\Delta}$. The first condition guarantees that *some* violation of a NR will flip the decision (by placing the modified instance outside class Δ) while the second condition guarantees that *all* violations of a GNR will flip the decision.

The next proposition explains why we require GNRs to be variable-minimal. Without this condition, the changes identified by a GNR to flip the decision may not be minimal (we can flip the decision by changing a strict subset of variables).

For instance $\mathcal{I}$ and clause σ s.t. $\mathcal{I} \models \sigma$, we will use $\mathcal{I} \dot{\cap} \sigma$ to denote the disjunction of states that appear in both $\mathcal{I}$ and σ (hence, $\mathcal{I} \dot{\cap} \sigma \models \sigma$). For example, if $\mathcal{I} = x_1 \cdot y_1 \cdot z_1$ and $\sigma = x_{12} + y_{23} + z_1$, then $\mathcal{I} \dot{\cap} \sigma = x_1 + z_1$.

Proposition 10. *Let $\mathcal{I}$ be an instance in class Δ and let σ be a strongest clause s.t. $\mathcal{I} \dot{\models} \sigma$ and $(\mathcal{I} \backslash\backslash \sigma) \cdot \overline{\sigma} \models \overline{\Delta}$. If σ' is another strongest clause satisfying these conditions and $vars(\sigma') \subset vars(\sigma)$, then $\mathcal{I} \backslash\backslash \sigma' \models \mathcal{I} \backslash\backslash \sigma$. Moreover, $\mathcal{I} \dot{\cap} \sigma$ is a NR iff such a clause σ' does not exist.*

That is, if violating σ requires changing some characteristics C of instance $\mathcal{I}$, then σ' can be violated by changing a strict subset of these characteristics C.

Consider the classifiers in Figs. 1a and 1b which make the same decision, YES, on Susan (AGE $\geq$55, BTYPE=A, WEIGHT=OVER). The NRs for these equal decisions are the same: (AGE $\geq$55) and (WEIGHT=OVER + BTYPE=A). The GNRs for the classifier in Fig. 1a are (AGE $\geq$55), (BTYPE $\in$ {A, B, AB} + WEIGHT=OVER) and (BTYPE $\in$ {A, B} + WEIGHT $\in$ {UNDER, OVER}). If the instance is changed to violate any of them, the decision will change. For example, if we set BTYPE to AB and WEIGHT to NORM, the third GNR will be violated and the decision on Susan becomes NO. For the classifier in Fig. 1b, the GNRs for the decision are different: (AGE $\geq$55) and (BTYPE $\in$ {A, O} + WEIGHT $\in$ {NORM, OVER}). However, both sets of GNRs contain more information than the NRs since the minimal changes they identify to flip the decision include those identified by the NRs.

Proposition 11. *Let σ be a simple clause. Then σ is a NR for the decision on instance $\mathcal{I}$ iff $\sigma = \mathcal{I} \hat{\cap} \sigma'$ for some GNR σ'.*

Consider the instance Susan again, $\mathcal{I} = (\text{Age} \geq 55) \cdot (\text{Btype}{=}\text{A}) \cdot (\text{Weight}{=}\text{over})$ and the classifier in Fig. 1b. As mentioned earlier, the GNRs for the decision on Susan are $\sigma'_1 = (\text{Age} \geq 55)$ and $\sigma'_2 = (\text{Btype} \in \{\text{A}, \text{O}\} + \text{Weight} \in \{\text{norm}, \text{over}\})$. Then $\sigma_1 = \mathcal{I} \hat{\cap} \sigma'_1 = (\text{Age} \geq 55)$ and $\sigma_2 = \mathcal{I} \hat{\cap} \sigma'_2 = (\text{Weight}{=}\text{over} + \text{Btype}{=}\text{A})$, which are the two NRs for the decision on Susan.

GSRs and GNRs are particularly significant when explaining the decisions of classifiers with numeric features, a topic which we discuss in Appendix C of [30].

We next present a fundamental result which allows us to compute GSRs and GNRs using the general reason for a decision (we use this result in Sect. 6).

Definition 9. *A prime implicant/implicate c of formula Δ is variable-minimal iff there is no prime implicant/implicate c' of Δ s.t. $vars(c') \subset vars(c)$.*

Proposition 12. *Let $\mathcal{I}$ by an instance in class Δ. The GSRs/GNRs for the decision on instance $\mathcal{I}$ are the variable-minimal prime implicants/implicates of the general reason $\overline{\forall} \mathcal{I} \cdot \Delta$.*

The disjunction of SRs is equivalent to the complete reason which is equivalent to the conjunction of NRs. However, the disjunction of GSRs implies the general reason but is not equivalent to it, and the conjunction of GNRs is implied by the general reason but is not equivalent to it; see Appendix D in [30]. This suggests that more information can potentially be extracted from the general reason beyond the information provided by GSRs and GNRs.

5 The General Reasons of Decision Graphs

Decision graphs are DAGs which include decision trees [7,9], OBDDs [10], and can have discrete or numeric features. They received significant attention in the work on explainable AI since they can be compiled from other types of classifiers such as Bayesian networks [47], random forests [12] and some types of neural networks [45]. Hence, the ability to explain decision graphs has a direct application to explaining the decisions of a broad class of classifiers. Moreover, the decisions undertaken by decision graphs have closed-form complete reasons as shown in [18]. We provide similar closed forms for the general reasons in this section. We first review decision graphs to formally state our results.

Each leaf node in a decision graph is labeled with some class c. An internal node T that *tests* variable X has outgoing edges $\xrightarrow{X, S_1} T_1, \ldots, \xrightarrow{X, S_n} T_n$, $n \geq 2$. The children of node T are $T_1, \ldots, T_n$ and $S_1, \ldots, S_n$ is a partition of *some* states of variable X. A decision graph will be represented by its root node. Hence, each node in the graph represents a smaller decision graph. Variables can be tested more than once on a path if they satisfy the *weak test-once property* discussed next [18,22]. Consider a path $\ldots, T \xrightarrow{X, S_j} T_j, \ldots, T' \xrightarrow{X, R_k} T_k, \ldots$ from the root to a leaf (nodes T and T' test X). If no nodes between T and T' on the path test

variable X, then $\{R_k\}_k$ must be a partition of states S_j. Moreover, if T is the first node that tests X on the path, then $\{S_j\}_j$ must be a partition of *all* states for X. Discretized numeric variables are normally tested more than once while satisfying the weak test-once property; see Appendix C in [30] for an illustration.

Proposition 13. *Let T be a decision graph, $\mathcal{I}$ be an instance in class c, and $\mathcal{I}[X]$ be the state of variable X in instance $\mathcal{I}$. Suppose $\Delta^c[T]$ is the class formula of T and class c. The general reason $\overline{\forall}\mathcal{I} \cdot \Delta^c[T]$ is given by the NNF circuit:*[8]

$$\Gamma^c[T] = \begin{cases} \top & \text{if } T \text{ is a leaf with class } c \\ \bot & \text{if } T \text{ is a leaf with class } c' \neq c \\ \prod_j (\Gamma^c[T_j] + \ell) & \text{if } T \text{ has outgoing edges } \xrightarrow{X, S_j} T_j \end{cases}$$

Here, ℓ is the X-literal $\{x_i \mid x_i \notin S_j\}$ if $\mathcal{I}[X] \notin S_j$, else $\ell = \bot$.

The following proposition identifies some properties of the above closed form, which have key computational implications that we exploit in the next section.

Proposition 14. *The NNF circuit in Proposition 13 is locally fixated on instance $\mathcal{I}$. Moreover, every disjunction in this circuit has the form $\ell + \Delta$ where ℓ is an X-literal, and for every X-literal ℓ' in Δ we have $\ell' \neq \ell$ and $\ell \models \ell'$.*

6 Computing Prime Implicants and Implicates

Computing the prime implicants/implicates of Boolean formulas was studied extensively for decades; see, e.g., [29,31,48]. The classical methods are based on *resolution* when computing the prime implicates of CNFs, and *consensus* when computing the prime implicants of DNFs; see, e.g., [15,20]. More modern approaches are based on passing encodings to SAT-solvers; see, e.g., [28,35,41]. In contrast, the computation of prime implicants/implicates of discrete formulas has received very little attention in the literature. One recent exception is [12] which showed how an algorithm for computing prime implicants of Boolean formulas can be used to compute simple prime implicants of discrete formulas given an appropriate encoding. Computing prime implicants/implicates of NNFs also received relatively little attention; see [14,18,42] for some exceptions. We next provide methods for computing variable-minimal prime implicants/implicates of some classes of discrete formulas that are relevant to GSRs and GNRs.

A set of terms S will be interpreted as a DNF $\sum_{\tau \in S} \tau$ and a set of clauses S will be interpreted as a CNF $\prod_{\sigma \in S} \sigma$. If S_1 and S_2 are two sets of terms, then $S_1 \times S_2 = \{\tau_1 \cdot \tau_2 \mid \tau_1 \in S_1, \tau_2 \in S_2\}$. For a set of terms/clauses S, $\ominus(S)$ denotes the result of removing subsumed terms/clauses from S.

[8] An NNF circuit is a DAG whose leaves are labeled with $\bot, \top$, or literals; and whose internal nodes are labelled with $\cdot$ or $+$.

Algorithm 1. GSR(Δ) — without Line 10, this is **Algorithm 2** PI(Δ)

Input: NNF circuit Δ which satisfies the properties in Proposition 14

```
1: if CACHE(Δ) ≠ NIL then return CACHE(Δ)
2: else if Δ = ⊤ then return {⊤}
3: else if Δ = ⊥ then return ∅
4: else if Δ is a literal then return {Δ}
5: else if Δ = α · β then
6:     S ← ⊖(GSR(α) × GSR(β))
7: else if Δ = α + β then
8:     S ← ⊖(GSR(α) ∪ GSR(β))
9: end if
10: S ← ⊠(S, ivars(Δ))
11: CACHE(Δ) ← S
12: return S
```

6.1 Computing General Sufficient Reasons

Our first result is Algorithm 1 which computes the variable-minimal prime implicants of an NNF circuit that satisfies the properties in Proposition 14 and, hence, is applicable to the general reasons of Proposition 13. If we remove Line 10 from Algorithm 1, it becomes Algorithm 2 which computes all prime implicants instead of only the variable-minimal ones. Algorithm 2 is the same algorithm used to convert an NNF into a DNF (i.e., no consensus is invoked), yet the resulting DNF is guaranteed to be in prime-implicant form. Algorithm 2 is justified by the following two results, where the first result generalizes Proposition 40 in [38].

In the next propositions, pi(Δ) denotes the prime implicants of formula Δ.

Proposition 15. *$pi(\alpha \cdot \beta) = \ominus(pi(\alpha) \times pi(\beta))$.*

Proposition 16. *For any disjunction $\alpha + \beta$ that satisfies the property of Proposition 14, $pi(\alpha + \beta) = \ominus(pi(\alpha) \cup pi(\beta))$.*

We will next explain Line 10 of Algorithm 1, $S \leftarrow \boxtimes(S, ivars(\Delta))$, which is responsible for pruning prime implicants that are not variable-minimal (hence, computing GSRs). Here, Δ is a node in the NNF circuit passed in the first call to Algorithm 1, and $ivars(\Delta)$ denotes variables that appear only in the sub-circuit rooted at node Δ. Moreover, $\boxtimes(S, V)$ is the set of terms obtained from terms S by removing every term $\tau \in S$ that satisfies $vars(\tau) \supset vars(\tau')$ and $V \cap (vars(\tau) \setminus vars(\tau')) \neq \emptyset$ for some other term $\tau' \in S$.[9] That is, term τ will be removed only if some variable X in $vars(\tau) \setminus vars(\tau')$ appears only in the sub-circuit rooted at node Δ (this ensures that term τ will not participate in constructing any variable-minimal prime implicant). This incremental pruning technique is enabled by the local fixation property (Definition 4).

[9] The condition $V \cap (vars(\tau) \setminus vars(\tau')) \neq \emptyset$ is trivially satisfied when Δ is the root of the NNF circuit since V will include all circuit variables in this case.

Proposition 17. *Algorithm 1, GSR(Δ), returns the variable-minimal prime implicants of NNF circuit Δ.*

6.2 Computing General Necessary Reasons

We can convert an NNF circuit into a CNF using a dual of Algorithm 2 but the result will not be in prime-implicate form, even for ciruits that satisfy the properties Proposition 14.[10] Hence, we next propose a generalization of the Boolean resolution inference rule to discrete variables, which can be used to convert a CNF into its prime-implicate form. Recall first that Boolean resolution derives the clause $\alpha + \beta$ from the clauses $x + \alpha$ and $\overline{x} + \beta$ where X is a Boolean variable.

Definition 10. *Let $\alpha = \ell_1 + \sigma_1$, $\beta = \ell_2 + \sigma_2$ be two clauses where ℓ_1 and ℓ_2 are X-literals s.t. $\ell_1 \not\models \ell_2$ and $\ell_2 \not\models \ell_1$. If $\sigma = (\ell_1 \cdot \ell_2) + \sigma_1 + \sigma_2 \neq \top$, then the X-resolvent of clauses α and β is defined as the clause equivalent to σ.*

We exclude the cases $\ell_1 \models \ell_2$ and $\ell_2 \models \ell_1$ to ensure that the resolvent is not subsumed by clauses α and β. If $\sigma = \top$, it cannot be represented by clause since a clause is a disjunction of literals over distinct variables so it cannot be trivial.

Proposition 18. *Closing a (discrete) CNF under resolution and removing subsumed clauses yields the CNF's prime implicates.*

The following proposition shows that we can incrementally prune clauses that are not variable-minimal after each resolution step. This is significant computationally and is enabled by the property of local fixation (Definition 4) which is satisfied by the general reasons in Proposition 13 and their CNFs.

Proposition 19. *Let S be a set of clauses (i.e., CNF) that is locally fixated. For any clauses σ and σ' in S, if $vars(\sigma') \subset vars(\sigma)$, then the variable-minimal prime implicates of S are the variable-minimal prime implicates of $S \setminus \{\sigma\}$.*

In summary, to compute GNRs, we first convert the general reason in Proposition 13 into a CNF, then close the CNF under resolution while removing subsumed clauses and ones that are not variable-minimal after each resolution step.

7 Conclusion

We considered the notions of sufficient, necessary and complete reasons which have been playing a fundamental role in explainable AI recently. We provided generalizations of these notions for classifiers with non-binary features (discrete or discretized). We argued that these generalized notions have more explanatory power and reveal more information about the underlying classifier. We further provided results on the properties and computation of these new notions.

Acknowledgments. This work has been partially supported by NSF grant ISS-1910317.

[10] The number of clauses in this CNF will be no more than the number of NNF nodes if the NNF is the general reason of a decision tree (i.e., the NNF has a tree structure).

References

1. Albini, E., Rago, A., Baroni, P., Toni, F.: Relation-based counterfactual explanations for Bayesian network classifiers. In: IJCAI, pp. 451–457 (2020). https://www.ijcai.org/
2. Amgoud, L.: Explaining black-box classifiers: properties and functions. Int. J. Approx. Reason. **155**, 40–65 (2023)
3. Amgoud, L., Ben-Naim, J.: Axiomatic foundations of explainability. In: IJCAI, pp. 636–642 (2022). https://www.ijcai.org/
4. Audemard, G., Bellart, S., Bounia, L., Koriche, F., Lagniez, J., Marquis, P.: On the explanatory power of Boolean decision trees. Data Knowl. Eng. **142**, 102088 (2022)
5. Audemard, G., Koriche, F., Marquis, P.: On tractable XAI queries based on compiled representations. In: KR, pp. 838–849 (2020)
6. Audemard, G., Lagniez, J., Marquis, P., Szczepanski, N.: Computing abductive explanations for boosted trees. CoRR abs/2209.07740 (2022)
7. Belson, W.A.: Matching and prediction on the principle of biological classification. J. R. Stat. Soc. Ser. C (Appl. Stat.) **8**(2), 65–75 (1959). https://www.jstor.org/stable/2985543
8. Boumazouza, R., Alili, F.C., Mazure, B., Tabia, K.: ASTERYX: a model-agnostic sat-based approach for symbolic and score-based explanations. In: CIKM, pp. 120–129. ACM (2021)
9. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth (1984)
10. Bryant, R.E.: Graph-based algorithms for Boolean function manipulation. IEEE Trans. Comput. **35**(8), 677–691 (1986)
11. Chan, H., Darwiche, A.: Reasoning about Bayesian network classifiers. In: UAI, pp. 107–115. Morgan Kaufmann (2003)
12. Choi, A., Shih, A., Goyanka, A., Darwiche, A.: On symbolically encoding the behavior of random forests. CoRR abs/2007.01493 (2020)
13. Choi, A., Xue, Y., Darwiche, A.: Same-decision probability: a confidence measure for threshold-based decisions. Int. J. Approx. Reason. **53**(9), 1415–1428 (2012)
14. de Colnet, A., Marquis, P.: On the complexity of enumerating prime implicants from decision-DNNF circuits. In: IJCAI, pp. 2583–2590 (2022). https://www.ijcai.org/
15. Crama, Y., Hammer, P.L.: Boolean functions - theory, algorithms, and applications. In: Encyclopedia of Mathematics and Its Applications (2011)
16. Darwiche, A.: Logic for explainable AI. In: 38th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS, pp. 1–11. IEEE (2023). CoRR abs/2305.05172
17. Darwiche, A., Hirth, A.: On the reasons behind decisions. In: ECAI. Frontiers in Artificial Intelligence and Applications, vol. 325, pp. 712–720. IOS Press (2020)
18. Darwiche, A., Ji, C.: On the computation of necessary and sufficient explanations. In: AAAI, pp. 5582–5591. AAAI Press (2022)
19. Darwiche, A., Marquis, P.: On quantifying literals in Boolean logic and its applications to explainable AI. J. Artif. Intell. Res. **72**, 285–328 (2021)
20. Gurvich, V., Khachiyan, L.: On generating the irredundant conjunctive and disjunctive normal forms of monotone Boolean functions. Discrete Appl. Math. **96**, 363–373 (1999)
21. Huang, X., Izza, Y., Ignatiev, A., Cooper, M.C., Asher, N., Marques-Silva, J.: Efficient explanations for knowledge compilation languages. CoRR abs/2107.01654 (2021)

22. Huang, X., Izza, Y., Ignatiev, A., Marques-Silva, J.: On efficiently explaining graph-based classifiers. In: KR, pp. 356–367 (2021)
23. Ignatiev, A., Izza, Y., Stuckey, P.J., Marques-Silva, J.: Using MaxSAT for efficient explanations of tree ensembles. In: AAAI, pp. 3776–3785. AAAI Press (2022)
24. Ignatiev, A., Narodytska, N., Asher, N., Marques-Silva, J.: From contrastive to abductive explanations and back again. In: Baldoni, M., Bandini, S. (eds.) AIxIA 2020. LNCS (LNAI), vol. 12414, pp. 335–355. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77091-4_21
25. Ignatiev, A., Narodytska, N., Marques-Silva, J.: Abduction-based explanations for machine learning models. In: Proceedings of the Thirty-Third Conference on Artificial Intelligence (AAAI), pp. 1511–1519 (2019)
26. Ignatiev, A., Narodytska, N., Marques-Silva, J.: On validating, repairing and refining heuristic ML explanations. CoRR abs/1907.02509 (2019)
27. Izza, Y., Ignatiev, A., Marques-Silva, J.: On tackling explanation redundancy in decision trees. J. Artif. Intell. Res. **75**, 261–321 (2022)
28. Izza, Y., Marques-Silva, J.: On explaining random forests with SAT. In: IJCAI, pp. 2584–2591 (2021). https://www.ijcai.org/
29. Jackson, P.: Computing prime implicates. In: Proceedings of the 1992 ACM Annual Conference on Communications, CSC 1992, pp. 65–72. Association for Computing Machinery, New York, NY, USA (1992). https://doi.org/10.1145/131214.131223
30. Ji, C., Darwiche, A.: A new class of explanations for classifiers with non-binary features. CoRR abs/2304.14760 (2023)
31. Kean, A., Tsiknis, G.: An incremental method for generating prime implicants/implicates. J. Symbolic Comput. **9**(2), 185–206 (1990)
32. Lang, J., Liberatore, P., Marquis, P.: Propositional independence: formula-variable independence and forgetting. J. Artif. Intell. Res. **18**, 391–443 (2003)
33. Lipton, P.: Contrastive explanation. Roy. Inst. Philos. Suppl. **27**, 247–266 (1990). https://doi.org/10.1017/S1358246100005130
34. Liu, X., Lorini, E.: A unified logical framework for explanations in classifier systems. J. Log. Comput. **33**(2), 485–515 (2023)
35. Luo, W., Want, H., Zhong, H., Wei, O., Fang, B., Song, X.: An efficient two-phase method for prime compilation of non-clausal Boolean formulae. In: 2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD), pp. 1–9 (2021). https://doi.org/10.1109/ICCAD51958.2021.9643520
36. Marques-Silva, J., Gerspacher, T., Cooper, M.C., Ignatiev, A., Narodytska, N.: Explanations for monotonic classifiers. In: ICML. Proceedings of Machine Learning Research, vol. 139, pp. 7469–7479. PMLR (2021)
37. Marques-Silva, J., Ignatiev, A.: Delivering trustworthy AI through formal XAI. In: AAAI, pp. 12342–12350. AAAI Press (2022)
38. Marquis, P.: Consequence finding algorithms. In: Kohlas, J., Moral, S. (eds.) Handbook of defeasible reasoning and uncertainty management systems, pp. 41–145. Springer, Cham (2000). https://doi.org/10.1007/978-94-017-1737-3_3
39. Miller, T.: Explanation in artificial intelligence: insights from the social sciences. Artif. Intell. **267**, 1–38 (2019)
40. Narodytska, N., Kasiviswanathan, S.P., Ryzhyk, L., Sagiv, M., Walsh, T.: Verifying properties of binarized deep neural networks. In: Proceedings of AAAI 2018, pp. 6615–6624 (2018)
41. Previti, A., Ignatiev, A., Morgado, A., Marques-Silva, J.: Prime compilation of non-clausal formulae. In: IJCAI, pp. 1980–1988. AAAI Press (2015)
42. Ramesh, A., Becker, G., Murray, N.V.: CNF and DNF considered harmful for computing prime implicants/implicates. J. Autom. Reason. **18**(3), 337–356 (1997)

43. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?": explaining the predictions of any classifier. In: KDD, pp. 1135–1144. ACM (2016)
44. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: high-precision model-agnostic explanations. In: AAAI, pp. 1527–1535. AAAI Press (2018)
45. Shi, W., Shih, A., Darwiche, A., Choi, A.: On tractable representations of binary neural networks. In: KR, pp. 882–892 (2020)
46. Shih, A., Choi, A., Darwiche, A.: A symbolic approach to explaining Bayesian network classifiers. In: IJCAI, pp. 5103–5111 (2018). https://www.ijcai.org/
47. Shih, A., Choi, A., Darwiche, A.: Compiling Bayesian network classifiers into decision graphs. In: AAAI, pp. 7966–7974. AAAI Press (2019)
48. Slagle, J., Chang, C.L., Lee, R.: A new algorithm for generating prime implicants. IEEE Trans. Comput. C- **19**(4), 304–310 (1970). https://doi.org/10.1109/T-C.1970.222917
49. Wang, E., Khosravi, P., den Broeck, G.V.: Probabilistic sufficient explanations. In: IJCAI, pp. 3082–3088 (2021). https://www.ijcai.org/