Constraint-Driven Explanations for Black Box ML Models

Aditya A. Shrotri[*1†], **Nina Narodytska**[*2],
Alexey Ignatiev[3], **Kuldeep S. Meel**[4], **Joao Marques-Silva**[5], **Moshe Y. Vardi**[1]

[1]Rice University, Houston, USA [2]VMware Research Inc., Palo Alto, USA [3]Monash University, Melbourne, Australia
[4]National University of Singapore, Singapore [5]IRIT, CNRS, Toulouse, France
{as128,vardi}@rice.edu, nnarodytska@vmware.com, alexey.ignatiev@monash.edu, meel@comp.nus.edu.sg,
joao.marques-silva@irit.fr

Abstract

The need to understand the inner workings of opaque Machine Learning models has prompted researchers to devise various types of post-hoc explanations. A large class of such explainers proceed in two phases: first perturb an input instance whose explanation is sought, and then generate an interpretable artifact to explain the prediction of the opaque model on that instance. Recently, Deutch and Frost proposed to use an additional input from the user: a set of constraints over the input space to guide the perturbation phase. While this approach affords the user the ability to tailor the explanation to their needs, striking a balance between flexibility, theoretical rigor and computational cost has remained an open challenge.

We propose a novel constraint-driven explanation generation approach which simultaneously addresses these issues in a modular fashion. Our framework supports the use of expressive Boolean constraints giving the user more flexibility to specify the subspace to generate perturbations from. Leveraging advances in Formal Methods, we can theoretically guarantee strict adherence of the samples to the desired distribution. This also allows us to compute fidelity in a rigorous way, while scaling much better in practice. Our empirical study demonstrates concrete uses of our tool CLIME in obtaining more meaningful explanations with high fidelity.

1 Introduction

The field of eXplainable AI (XAI) has emerged out of the need for humans to understand the complex and opaque decision processes governing modern Machine Learning models. Researchers seek to develop both naturally interpretable models (Hu, Rudin, and Seltzer 2019; Angelino et al. 2018; Rudin 2019; Avellaneda 2020) as well as post-hoc explanations for opaque models like Deep Neural Networks and ensembles (Ribeiro, Singh, and Guestrin 2016; Lundberg and Lee 2017). State-of-the-art learning approaches in most domains, however, are uninterpretable and necessitate the latter approach.

A number of different approaches have been proposed in literature for generating post-hoc explanations (c.f. Adadi and Berrada (2018a)). A broad class of techniques explain individual predictions by capturing the behavior of the opaque model in a small neighborhood of the input instance in two phases (Ribeiro, Singh, and Guestrin 2016; Shrikumar, Greenside, and Kundaje 2017; Simonyan, Vedaldi, and Zisserman 2013). In the first phase, the input instance is perturbed according to some criteria and in the second phase the behavior of the model on the perturbed instances is captured using various interpretable artifacts such as linear classifiers (Ribeiro, Singh, and Guestrin 2016; Lundberg and Lee 2017), gradients (Zeiler and Fergus 2014; Sundararajan, Taly, and Yan 2017), counterfactuals (Wachter, Mittelstadt, and Russell 2017), subgraphs of GNNs (Ying et al. 2019) etc.

Different choices for each phase yield different tradeoffs between flexibility, computational cost and theoretical rigor. For example, one of the earliest and most popular post-hoc explainers called LIME (Ribeiro, Singh, and Guestrin 2016) employs a fixed heuristic perturbation procedure, and uses a simple linear classifier trained on the perturbed instances as the interpretable artifact. The advantages are that LIME is model-agnostic in that it can explain predictions of any black-box model, and is reasonably fast in practice. However, it suffers from drawbacks like lack of a strong theoretical foundation and susceptibility to adversarial attacks (Slack et al. 2020) among others. The explainer SHAP (Lundberg and Lee 2017) rectified some of these issues by using Shapley values from game theory for axiomatically deriving the coefficients of the linear model. While Shapley values are provably 'ideal' under some mild assumptions, they are expensive to compute (Van den Broeck et al. 2021; Arenas et al. 2021), and in practice we have to resort to approximations or accept the loss of model-agnosticity. Recently, Deutch and Frost (2019) proposed to employ user-defined constraints to generate explanations. In particular, they allow the user to supply domain knowledge through constraints in the form of linear inequalities over the input space. These constraints guide the perturbation procedure for generating counterfactual explanations. While constraints provide flexibility in tailoring explanations, the modeling language (linear inequalities) is restrictive and the proposed algorithm is not model-agnostic. Thus, striking a balance between flexibility, rigor, and com-

[*]These authors contributed equally.

[†]This work was mostly done during internship at VMware Research. Code, results and full version of the text is available at https://gitlab.com/Shrotri/clime

putational efficiency remains a major challenge.

In this work, we take a step towards addressing this challenge via design of an efficient constraint-driven explanation framework that provides robust theoretical guarantees. Firstly, along with the classifier to be explained, our framework takes constraints in the form of Boolean formulas as input. Boolean formulas are known to be expressive enough to succinctly encode *all* types of constraints on discrete spaces (Cadoli and Schaerf 2005). This allows the user to precisely define the distribution of perturbed samples, and gives them the flexibility to drill down into the structure of input space. Secondly, by leveraging advances in formal methods, we can generate perturbed samples with strong theoretical guarantees on adherence to the desired distribution, while scaling to large formulas in practice (Soos, Gocht, and Meel 2020; Gupta et al. 2019). This allows us to rigorously measure the fidelity of the generated interpretable artifact to the input model i.e., how closely the artifact actually explains the model. Finally, our perturbation framework is decoupled from the artifact generation phase, and the generated samples can be used to train any surrogate classifier that is appropriate for the task. Following LIME, we build a linear model over the generated samples as the interpretable artifact. We experimentally demonstrate how the resulting tool, called CLIME, can be used for generating high quality explanations.

In summary, our contributions are as follows:

1. Framework for precisely crafting explanations for specific subspaces of the input domain through logical constraints
2. A theoretical framework and an efficient algorithm for estimating the 'true' fidelity up to any desired accuracy
3. Empirical study showing the efficacy of constraints in
   - Efficient fidelity computation with strong guarantees
   - Zooming in and refining explanations guided by fidelity
   - Detecting and foiling adversarial attacks

2 Preliminaries

We follow notations from (Ribeiro, Singh, and Guestrin 2016). Let $D = (X, y) = \{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\}$ denote the input dataset from some distribution $\mathcal{D}$ where $x^i \in \mathbb{R}^d$ is a vector that captures the feature values of the ith sample, and $y^i \in \{\mathcal{C}_0, \mathcal{C}_1\}$ is the corresponding class label[1]. We use subscripts, i.e. x_j, to denote the j^{th} feature of the vector x. We denote by $f : \mathcal{R}^d \to [0, 1]$ the opaque classifier that takes a data point x^i as input and returns the probability of x^i belonging to $\mathcal{C}_1$. We assume that an instance x is assigned label $l_f(x) = \mathcal{C}_1$ if $f(x) \geq 0.5$ and $l_f(x) = \mathcal{C}_0$ otherwise.

Surrogate linear models. The exact problem formulation varies depending on the choice of interpretable artifact to be generated. Following LIME, SHAP and a host of other popular methods, we choose a simple linear model, as our explanation artifact. Specifically, given a classifier f, the task is to learn a linear model g such that g mimics the behavior of f in the neighborhood of some given point x. The function g is built on an 'interpretable domain' of inputs rather than the original domain. To do so, the original features (that can be continuous or categorical) are mapped to Boolean features. While $x \in \mathbb{R}^d$ represents an instance in the original domain, we use prime-notation, i.e. $x' \in \{0, 1\}^{d'}$ to represent an instance in the interpretable domain. Using Boolean features is a natural choice for 'interpretable domain', as we can understand explanations in terms of a presence/absence of a feature's value. Thus g operates in the interpretable domain $\{0, 1\}^{d'}$. Existing explainers differ in the way the x is perturbed and g is trained. For instance, LIME perturbs the interpretable instance x' by randomly changing 1s to 0s in the binary representation. The generated samples ${z'}^1, {z'}^2, \ldots \in \mathcal{Z}'$ are mapped back to the original space as $z^1, z^2, \ldots \in \mathcal{Z}$, where $\mathcal{Z}$ and $\mathcal{Z}'$ are called the neighborhoods of x and x' in the respective spaces. We highlight that we follow the same technique for mapping between the original domain and the interpretable domain that is used by LIME. In case of tabular data with continuous features the mapping is not injective, and so one of the pre-images of each interpretable feature value is randomly selected as the corresponding original feature value. The instances ${z'}^1, {z'}^2, \ldots$ with corresponding labels $f(z^1), f(z^2) \ldots$ are used as the training set along with a heuristic loss function for building a linear model g using regression. In Sec. 3, we discuss the limitations of this approach and show how constraint-sampling can mitigate some of these issues.

Boolean (logical) constraints and uniform sampling. We use notation standard in the area of Boolean Satisfiability (SAT). A Boolean formula over n variables $\varphi : \{0, 1\}^n \to \{0, 1\}$ assigns a truth value $0/1$ or false/true to each of the 2^n assignments to it's variables and is constructed using logical operators like AND ($\wedge$), OR ($\vee$), NOT ($\neg$), XOR ($\oplus$) etc. An assignment of truth values to variables denoted $s \in \{0, 1\}^n$ is said to satisfy φ (denoted $s \models \varphi$) iff $\varphi(s) = 1$. The total number of assignments that satisfy φ is denoted as $\#\varphi = \sum_{s \in \{0,1\}^n} \varphi(s)$. An algorithm is said to be a (perfectly) uniform sampler if it takes as input an arbitrary formula φ and returns an assignment s^* such that $\forall s \models \varphi$, we have $Pr[s^* = s] = \frac{1}{\#\varphi}$. An almost-uniform sampler is an algorithm that takes, along with φ, a parameter $\varepsilon > 0$ as input and returns s^* such that $\forall s \models \varphi$, we have $\frac{1}{(1+\varepsilon)\#\varphi} \leq Pr[s^* = s] \leq \frac{1+\varepsilon}{\#\varphi}$. The tools WAPS (Gupta et al. 2019) and UniGen3 (Soos, Gocht, and Meel 2020) are state-of-the-art perfectly uniform and almost-uniform samplers respectively.

Fidelity. The notion of fidelity aims to capture how closely the explainer model 'reflects' the behavior of the opaque model, and can be seen as a measure of the quality of the explanation (Ribeiro, Singh, and Guestrin 2016, 2018). The fidelity $\widehat{\rho}$ of the explainer model g to the opaque model f is calculated as the precision of g (Ribeiro, Singh, and Guestrin 2018), i.e. the fraction of the sampled neighbors where the output class of f and g agree. Let $\mathcal{Z}$ and $\mathcal{Z}'$ be the neighborhoods of x and x' as defined above. Then,

$$\widehat{\rho} = \frac{\sum_{z' \in \mathcal{Z}'} \mathcal{I}[l_f(z) = l_g(z')]}{|\mathcal{Z}'|} \tag{1}$$

[1] We focus on binary classification; extension to multi-class classification follows by one-vs-rest approach.

where $\mathcal{I}$ is the indicator function, and z is the preimage of z'.

3 Constraint-Driven Explanations

We present our framework CLIME which belongs to a large class of post-hoc explainers that operate in two-phases: in the first phase, it perturbs an input instance x and in the second phase, it generates an interpretable model g to explain the prediction of the opaque model f on x. The new distinctive capability of CLIME is that it lets the user specify constraints on the input space to define the allowed perturbations of x in a model-agnostic way. Next, we discuss our choice for the constraint modeling language and give high-level overview of the two-phase algorithm.

Constraint modeling language. We assume that the constraints are specified in propositional logic, i.e. as a Boolean formula φ . Boolean constraints are powerful enough to represent log-linear family of distributions (Chavira and Darwiche 2008), yet allow fast sampling of solutions either uniformly or with a user-provided bias (Gupta et al. 2019), thanks to the advances in SAT technology (Marques-Silva, Lynce, and Malik 2021). Boolean constraints are also easy to use, and many toolkits for formal analysis such as model-checkers (Clarke et al. 2001) require their input to be specified using Boolean logic.

As an example, assume that φ represents the constraint that at least k features must be fixed for some user-defined k. For image data this constraint enforces the requirement that at least k superpixels must be 'on', while for text it forces at least k words from x to be present in each sample. This blocks out very sparse data ensuring that only informative instances are used for training the explainer model. Example 3.1 describes more scenarios where constraints are useful.

The first phase: sampling data points. The CLIME framework generates explanations on instances sampled (almost-) uniformly from user-defined subspaces which are defined through constraints. In this work, we employ techniques for (almost) uniformly sampling solutions of constraints for generating explanations, but we note that the extension to biased (weighted) sampling is straightforward (Chakraborty et al. 2015).

The pseudo-code of the constrained explanation framework is presented in Alg. 1. Along with the input instance x CLIME also takes as input a Boolean formula φ. The variables of φ are exactly the Boolean features of the interpretable domain $\mathcal{Z}'$, and the solutions φ is the user-defined subspace $\mathcal{U}^{\mathcal{Z}'}$ i.e. $\mathcal{U}^{\mathcal{Z}'} = \{s \in \{0,1\}^n \mid s \models \varphi\}$.

The samples generated from φ determine the neighborhood $\mathcal{Z}'$ (line 1 of Alg. 1). Note that $\mathcal{U}^{\mathcal{Z}'}$ is the universe of all possible assignments from which $\mathcal{Z}'$ is sampled. We assume access to a procedure $getSamples$ that returns N independent samples satisfying φ. The algorithm takes as input a parameter ε, which represents the tolerance to deviation from perfectly-uniform sampling. If $\varepsilon = 0$, then the call to $getSamples$ in line 1 must be to a perfectly uniform sampler like WAPS (Gupta et al. 2019), otherwise an almost-uniform sampler like Unigen3 (Soos, Gocht, and Meel 2020) suffices. We highlight that CLIME is the first framework with a capability to incorporate constraints to generate explanations in model-agnostic settings, to the best of our knowledge.

The second phase: learning an explainer. We adopt LIME's method for training a linear explainer model for the second (artifact generation) phase. The samples z'^i are mapped back to the original domain, and the output of f on each z^i and the distance of each z^i to x are used for training g in line 6, where at most K coefficients are allowed to be non-zero to ensure interpretability. More formally, let the complexity of an explanation g be denoted as $\Omega(g)$ (complexity of a linear model can be the number of non-zero weights), and let $\pi_x(z)$ denote the proximity measure between inputs x and $z \in \mathcal{Z}$, where $\pi_x(z)$ can be defined using cosine or L_2 distance. The objective function for training g is crafted to ensure that g (1) approximates the behavior of f accurately in the vicinity of x where the proximity measure is high, and (2) achieves low complexity and is thereby interpretable. The explanation is obtained as $g^* = argmin_{g \in G}\ L(\pi_x, g, f) + \Omega(g)$ where G is the set of all linear classifiers and the loss function L is defined as: $L(f, g, \pi_x) = \sum_{z \in \mathcal{Z}}[f(z) - g(z')]^2 \pi_x(z)$. Intuitively, the loss function captures how unfaithful g is to f in the neighborhood of x.

Example 3.1. *We consider the bank dataset (Moro, Cortez, and Rita 2014) that was also used in Deutch and Frost (2019). The bank dataset contains bank client data that describes client characteristics as well as their communications with a bank. The model predicts whether a client will subscribe for the term deposit. Four* integrity *constraints were proposed for this dataset by Deutch and Frost (2019). Integrity constraints enforce data consistency and accuracy.*

- ***Previous contacts with a client.*** *Two constraints were proposed. A client has not been contacted before iff the time since previous contact is undefined. A client has not been not contacted before iff the previous outcome is unknown. In terms of the features, these constraints are expressed as ('previous' = 0) ⇔ ('pdays' = undefined) ⇔ ('poutcome' = unknown).*
- ***Features interdependencies.*** *Two constraints were proposed. If a client is a student then they are not married and younger than 35 years old. If a client has an 'admin' job then their education is secondary.*

We can find explanations for a random sample in two scenarios. First, we do not supply constraints to the explainer. In this case, we get an explanation: I = ('duration', 'housing', 'previous', 'poutcome','pdays'), which consists of the 5 most important features (by weight) that contributed to the prediction of the opaque classifier on the input instance. If we add integrity constraints then we get a different explanation (i.e. different set of top 5 features): J = ('duration', 'pdays', 'housing', 'campaign', 'loan').

At this point, it is hard for the user to judge the quality of these explanations I and J. In the next section, we present a technique that enables users to make this judgement.

□

Algorithm 1: ExplainWithCLIME$(f, \varphi, \varepsilon, N, x, x', \pi_x, K)$

Input: f: Model to be explained φ: Boolean constraints
ε: Tolerance N: Number of samples
π_x: Similarity kernel K: Length of explanation
Output: g: Interpretable linear classifier

$\mathcal{Z}' \leftarrow$ getSamples$(\varphi, \varepsilon, N)$;
$Z \leftarrow \{\}$
for $z' \in \mathcal{Z}'$ **do**
    $Z \leftarrow Z \cup \{z', f(z), \pi_x(z)\}$
end for
$g \leftarrow$ K-LASSO(Z, K)

Algorithm 2: computeFidelity$(f, g, \varepsilon, \delta, \gamma)$

Input: f: Model to be explained g: Explainer Model
ε: Tolerance δ: Confidence γ: Threshold
Output: $\widehat{\rho}$: Estimate of ρ (see Thm. 1)

if checkThreshold$(f, g, \varepsilon, \delta, \gamma)$ == True **then**
    return $\perp$ ▷ $\widehat{\rho} \leq \gamma - \varepsilon$; report failure
end if
/*Threshold check passed; compute 2-sided bound*/
$\widehat{\rho} \leftarrow AA'(0.4 * \varepsilon, 0.4 * \varepsilon, \delta)$ ▷ See Appendix A
return $\widehat{\rho}$

Algorithm 3: checkThreshold$(f, g, \varepsilon, \delta, \gamma)$

Input: f: Model to be explained g: Explainer Model
ε: Tolerance δ: Confidence γ: Threshold
Output: True with high probability if $\rho \leq \gamma - \varepsilon$

$\nu \leftarrow \min(\varepsilon + \varepsilon^2/2 - \gamma\varepsilon/2, (\varepsilon - \gamma\varepsilon/2)/(1 + \varepsilon/2))$
/* compute the number of samples N based on $\varepsilon, \delta, \gamma$*/
$N \leftarrow \frac{1}{2\nu^2} \log(\frac{1}{\delta})$
$\mathcal{Z}' \leftarrow$ getSamples$(\varphi, \varepsilon/2, N)$
$C \leftarrow 0$
/* compute sample fidelity */
for $z' \in \mathcal{Z}'$ **do**
    /*z is the preimage of z'*/
    $c \leftarrow \mathcal{I}[l_f(z) = l_g(z')]$
    $C \leftarrow C + c/N$
end for
if $C \leq \gamma$ **then**
    /*Value below threshold; terminate early*/
    return True
else
    return False
end if

4 Certifying Explanation Quality

For increasing user trust, it is necessary to provide a measure of the quality of explanations generated. A fundamental requirement from a high-quality explainer model is that it should closely mimic the behavior of the opaque model in the specified neighborhood. This is especially important for explanations of user defined sub-spaces, as it may be possible that no simple explanation exists for a large subspace, and further refinements to the constraints may be required to get a high-quality explanation.

The fidelity metric, as defined in Eqn. 1, aims to quantify this property, in terms of the fraction of samples in the neighborhood $\mathcal{Z}'$, on which the prediction made by the explainer model matches the prediction of the opaque model. Two parameters influence the accuracy of the fidelity score: the number of samples in $\mathcal{Z}'$ and the quality of these samples, i.e. their uniformity in the universe $\mathcal{U}^{\mathcal{Z}'}$ of all such possible samples. Both of these parameters were chosen heuristically in prior works (Ribeiro, Singh, and Guestrin 2016, 2018), which raises the question, *is the fidelity score, as measured by Eqn. 1 trustworthy?* Intuitively, a score measured on 10 samples will not be as accurate as one measured on 10000 due to randomness inherent in any sampling procedure. Such uncertainties can be unacceptable in, for instance, safety-critical applications of XAI such as healthcare (Amann et al. 2020).

We address this gap by first rigorously defining fidelity, and then presenting an efficient algorithm for computing it. We observe that the true fidelity score is the one that is calculated on *all* possible instances belonging to a user-defined subspace of inputs $\mathcal{U}^{\mathcal{Z}'}$, i.e.

$$\rho = \frac{\sum_{z' \in \mathcal{U}^{\mathcal{Z}'}} \mathcal{I}[l_f(z) = l_g(z')]}{|\mathcal{U}^{\mathcal{Z}'}|} \tag{2}$$

The user-defined subspace $\mathcal{U}^{\mathcal{Z}'}$ consists of the solutions of the input formula φ. In practice, φ can have hundreds of variables and exponentially many solutions which makes enumerating all elements of $\mathcal{U}^{\mathcal{Z}'}$ in the numerator of Eqn. 2 infeasible. Thus, computing ρ exactly is usually intractable. Approximating ρ can be faster, but requires formal guarantees to be meaningful. We observe that the score $\widehat{\rho}$, as measured by Eqn. 1, is the 'sample mean' of the true 'population mean' ρ, as defined by Eqn. 2. This observation allows us to compute the estimate $\widehat{\rho}$ in theoretically grounded way, so as to statistically guarantee its closeness to ρ.

We use a PAC-style notion of approximation (Valiant 1984), which provides strong probabilistic guarantees on the accuracy of the output. The goal is to find an approximation ρ that is within user-defined tolerance of the true value with high confidence. Specifically, we wish to compute $\widehat{\rho}$ such that

$$\Pr[(1 - \varepsilon)\rho \leq \widehat{\rho} \leq (1 + \varepsilon)\rho] \geq (1 - \delta) \tag{3}$$

where $\varepsilon > 0$, $\delta > 0$ are user-defined tolerance and confidence.

To the best of our knowledge, no existing approach is directly applicable to finding a good approximation of ρ, in a model-agnostic way. The technique presented by (Narodytska et al. 2019), requires the opaque model to be encoded as a Boolean formula, severely limiting both its scalability as well as the types of models that can be explained. On the other

hand, algorithms based on Monte Carlo sampling such as the AA algorithm by (Dagum et al. 2000), are known to be fast when ρ is high, but require far too many samples when ρ is low (Meel, Shrotri, and Vardi 2019). They also require perfectly uniform samples, while it may only be feasible to generate almost-uniform samples from the universe $\mathcal{U}^{\mathcal{Z}'}$.

In this section, we propose an efficient and model-agnostic estimation algorithm based on (Dagum et al. 2000), that is able to work with almost-uniform samples and also terminates quickly if the quantity being approximated is small. Two key insights inform the design of our approach: we first observe that ε-almost uniform sampling can change the value of ρ at most by a factor of $(1+\varepsilon)$. Secondly, in typical scenarios, users are interested in two-sided bounds on fidelity (as given by Eqn. 3) only if it is high enough. If the fidelity is lower than some threshold, say 0.1, then it doesn't matter if it is 0.05 or 0.01, since the explanation will be unacceptable in either case. In other words, below a certain threshold, one-sided bounds suffice.

Procedure computeFidelity (Alg. 2) is used for computing $\widehat{\rho}$, given an opaque model f, an explainer model g and three parameters ε, δ and γ that control the precision of $\widehat{\rho}$. computeFidelity invokes checkThreshold (Alg. 3) on line 1. checkThreshold first computes the number of samples N required for the probabilistic guarantees, and then invokes a sampler through $getSamples$ as in Alg. 1. If $\widehat{\rho} \leq \gamma - \varepsilon$, then checkThreshold returns True with probability at least $1-\delta$ and computeFidelity reports failure on line 2. This check ensures that the sample complexity remains low even if the fidelity is very small, which is a common pitfall for Monte Carlo algorithms. If checkThreshold returns False, then computeFidelity makes a call to procedure AA' (line 4), which is an adaptation of the algorithm by (Dagum et al. 2000) that provides the guarantees of Eqn. 3 with almost-uniform samples (see Appendix A in the full version). Theorem 1 captures the guarantees and the behavior of the framework.

Theorem 1. *If $\rho \leq \gamma - \varepsilon$, then* computeFidelity *returns $\perp$ with high probability (i.e. at least $1-\delta$). If $\rho \geq \gamma + \varepsilon$, w.h.p., it returns an estimate $\widehat{\rho}$ such that $\Pr[(1-\varepsilon)\rho \leq \widehat{\rho} \leq (1+\varepsilon)\rho] \geq (1-\delta)$.*

We highlight that our certification framework is more general than just fidelity computation. In Appendix A in the full version, we show how Algs. 2 an 3 can be used for accurately estimating the true mean of any 0/1 random variable with almost-uniform samples and early termination.

Example 4.1. *We continue with Example 3.1. Now, the user can use the fidelity metric to compare quality of explanations. We compute the fidelity score for both explanations I and J that we obtained without and with constraints, respectively. We get that fidelity(I)= 0.91 and fidelity(J)= 0.90. First, the user notices that the fidelity scores are the same for these explanations, so I and J can be seen as explanations of the same quality. Second, these fidelity scores can be considered low, hinting the user to refine the input space. In the next section we show, through an extensive evaluation, how the user can perform such a refinement to obtain high quality explanations.* □

The overall schematic of CLIME is shown in Fig. 1. Blue boxes highlight CLIME's building blocks that are different from LIME. Namely, CLIME allows a user to specify constraints, performs constrained sampling and computes the fidelity metric of a explainer.

5 Experiments

We seek to answer the following research questions through our empirical study:

1. How scalable is the certification framework presented in Sec. 4?
2. What benefits do constraints provide for analysing ML models?
3. How susceptible are constrained explanations to adversarial attacks?

Efficiency of certification

A salient benefit of leveraging the AA-algorithm of (Dagum et al. 2000) for the fidelity computation approach of Sec. 4, is that its sample complexity is guaranteed to be close-to-optimal. Nevertheless, the practical performance of our approach is unknown apriori, given the added cost of generating (almost-) uniform samples from constraints. Therefore, in this experiment, we evaluate the scalability of our framework vs. that of the technique of (Narodytska et al. 2019).

We implemented and ran Algs. 2, 3 on 150 benchmarks used in (Narodytska et al. 2019). The benchmarks are CNF formulas that encode the Anchor (Ribeiro, Singh, and Guestrin 2018) explanations of Binarized Neural Networks trained on Adult, Recidivism and Lending datasets. The true fidelity of an explanation can be computed from the count of the number of solutions of the corresponding formula. We compared the running-time of our tool to the time taken by the approach of (Narodytska et al. 2019), which utilizes the state-of-the-art approximate model-counting tool called ApproxMC (Soos, Gocht, and Meel 2020). Note that both ApproxMC and our tool provide the same probabilistic guarantees on the returned estimate (Soos and Meel 2019). The results are shown as a scatter-plot in Fig. 2. The x-coordinate of a point in blue represents the time taken by ApproxMC on a benchmark, while the y-coordinate represents the time taken by our approach. As all the points are far below the diagonal dotted red line, we can infer that ApproxMC takes significantly longer than our tool to compute the same estimate. In fact, on average (geometric mean), our tool is $7.5\times$ faster than ApproxMC. It is clear from Fig. 2 that our algorithm scales far better than the alternative, despite being more general and *model-agnostic*. We also experimentally compared the scalability of our tool to ApproxMC for different values of input tolerance ε and confidence δ. We found that our tool scales significantly better than ApproxMC for tighter tolerance and confidence values. Thus, our experiments demonstrate that our tool significantly outperforms the state-of-the-art. We provide more details and results in Appendix A in the full version.

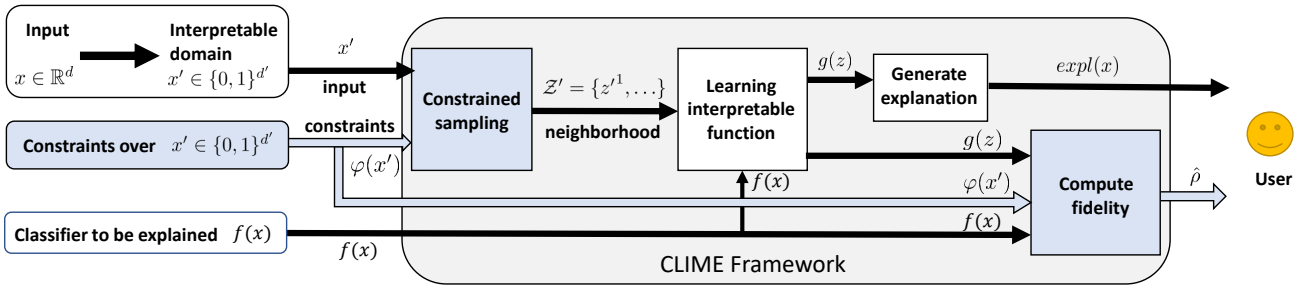


Figure 1: CLIME Schema

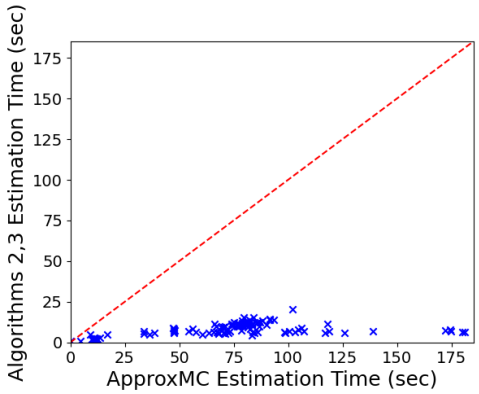


Figure 2: Scalability of Algs. 2,3 vs. ApproxMC

Model analysis

First, we consider the bank dataset that was proposed in Deutch and Frost (2019) that we describe in Example 3.1. We train 10 random forest models with different random seeds and the same hyper-parameters as in Deutch and Frost (2019). The average accuracy of these models is 90%. In all experiments, we compute the average fidelity scores over 100 input explanations and 10 RF models.

Let us consider a scenario where the user needs to explain why a client who has not been contacted in the past made a decision to not subscribe for a term deposit. We find explanations for 100 samples per model in two scenarios: (a) without constraints and (b) with integrity constraints, as in Example 4.1, and average the result. We get fidelity scores 0.90 and 0.89 for scenarios (a) and (b), respectively. These result confirm scores that we obtain for a single instance in Example 4.1. However, the obtained fidelity scores might not be acceptable for the user. A low fidelity score can indicate either that the constrained space needs to be refined or that the interpretable artifact needs to be changed (ex: using decision trees instead of linear classifiers to explain non-linear decision boundaries). We highlight that unlike Deutch and Frost (2019), under our modular framework, it is easy to learn a different artifact. In this work, we focus on constraint refinement and assume that the user intends to continue drilling down by specifying *user-defined* constraints to better communicate their focus space to an explainer.

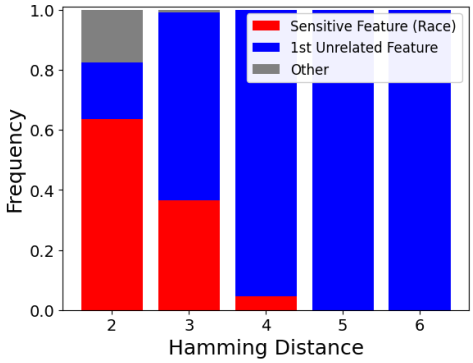
To achieve their goal, the user can specify an additional constraint: 'consider only clients that have not been previously contacted'. So, the user adds this constraint on top of integrity constraints, creating a new setup: (c) CLIME is supplied with integrity and the user constraint. We again compute the average fidelity score that is 0.98 for the scenario (c). Clearly, adding the user-defined constraint allowed to refine the input space to obtain high quality explanations.

Note that the user-defined constraint 'a client has not been contacted before' triggers integrity constraints (see Example 3.1 for the definition of integrity constraints) forcing that 'the time since previous contact should be undefined' and 'the previous outcome should be unknown'. Hence, these features, i.e. 'previous', 'pdays', 'poutcome' in the dataset, are *fixed* by the user's constraints. Therefore, these fixed features should not appear in the explanations. Table 1 shows the top 5 features that were used in explanations for the scenarios (a) and (c) in 100 instances. Note that three fixed features are often chosen by CLIME *without* constraints (scenario (a)), making these explanations less useful for the user. In contrast, CLIME with constraints (scenario (c)) never chooses these features in its explanations demonstrating the correct behaviour.

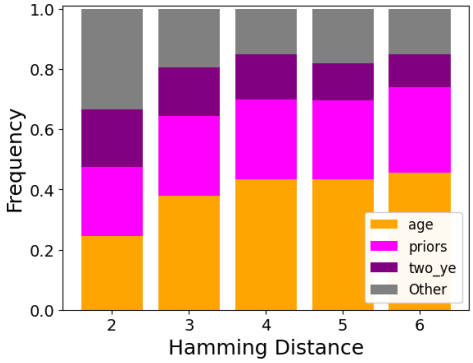
Second, we consider the adult dataset (Kohavi 1996), originally taken from the Census bureau. It is used for predicting whether or not a given adult person earns more than $50K a year depending on various attributes, e.g. race, sex, education, hours of work, etc. We pre-processed columns with continuous features, e.g. the pre-processor discretizes the capital gain and capital loss features into categorical features, e.g. 'Unknown', 'Low' and 'High'(Ribeiro, Singh, and Guestrin 2018). We train 10 Random Forest models with different random seeds and 20 trees and max depth is 7. The accuracy is 83% on average. We compute the average fidelity score of 100 inputs explanations and RF models. Consider a scenario when the user wants to find an explanation for male individuals without a college degree. So, the user adds a constraint (EDUCATION $\in$ [DROPOUT, HIGH-SCHOOL, SOME-COLLEGE]) AND (SEX = MALE). Next, they find that the average fidelity score is 0.68 for explanations in this constrained space. This indicates a need for refining the in-

Scenario	Top five features (left to right) in explanations				
(a)	'previous'	'pdays'	'duration'	'poutcome'	'housing'
(c)	'age'	'contact'	'duration'	'personal loan'	'month'

Table 1: Features of Bank dataset appearing in explanations (a) without constraints, and (c) with integrity and user constraints



(a) Adversarial Classifier



(b) Non-Adversarial Classifier

Figure 3: Recidivism: Top CLIME explanation distribution vs. Hamming Distance

put space. For example, they can add a constraint that there is no information available about individuals' Capital, i.e.: (CAPITAL GAIN = 'UNKNOWN') AND (CAPITAL LOSS = 'UNKNOWN'). In this more constrained space, the average fidelity score of explanations increases to 0.98. Therefore the user can be confident that explanations are reliable.

Detecting Adversarial Attacks

(Slack et al. 2020) presented a way to craft an adversarial attack that seeks to hide the biased predictions made by a biased classifier (eg: one that decides credit card applications solely on a sensitive feature like race) from detection by post-hoc explainers like LIME, by exploiting the ad-hoc perturbation procedure that can generate Out-of-Distribution (OOD) samples. In this experiment we evaluate the suitability of constraints in foiling such attacks.

Adversarial classifier setup. Given a biased classifier, (Slack et al. 2020) construct an adversarial classifier that 'wraps' around the biased classifier aiming to hide its biases. It relies on the ability to successfully train a classifier to predict whether an input sample is in- or out-of-distribution (OOD). This classifier, called the 'OOD detector', is used as a submodule in the adversarial classifier to predict whether an input instance is OOD or not. If the input instance is OOD, then the adversarial classifier outputs a *prediction based on an innocuous feature*. If the instance is not OOD, then it outputs the biased classifier's prediction. When LIME is used for explaining a prediction made by the adversarial classifier, almost all of LIME's perturbed samples are detected to be OOD, resulting in the innocuous feature appearing as the top explanation. In this way, the sensitive feature (eg. race), is successfully hidden, when in reality it was the ground-truth explanation. The attack crucially relies on the ability to train an OOD detector to tell apart LIME's perturbed samples and instances from the original distribution with high accuracy. Note that accurately measuring the fidelity of the explainer model alone is not sufficient to detect such an attack as the problem lies in the way the opaque classifier is constructed and not in the resulting explanation.

A blind spot of the OOD detector. We hypothesized that for discrete data, the OOD detector should have trouble differentiating between instances from the original dataset and instances with low Hamming distances to points in the original dataset. We emphasize here, that LIME almost always generates samples at a high Hamming distance from the input instance, as it perturbs each feature independently. Consequently, the penalization of high Hamming distance, either through the loss function or through explicit filtering by rejection sampling, is ultimately ineffective, as the number of samples is ad-hoc and fixed apriori. This may partly explain LIME's susceptibility to such attacks.

Our idea is to generate constrained samples based on their Hamming distance h from the original input. For adversarial classifiers, for small values of h, we hit the blind spot of the OOD detector and our sampled instances will be marked as in-distribution. So, sensitive features will be revealed by CLIME. As we move away from the instance, the innocuous features should dominate. Towards this end, we constructed a formula φ^h for a given Hamming distance h and a set of instances S, such that the solutions of φ^h are exactly those instances that are at most at a distance h from each instance in S. We took S to be the set of 3 nearest neighbors of the instance being explained. If our hypothesis is correct, the adversarial classifier should have trouble hiding the sensitive features in the subspace $\mathcal{Z}_{\varphi^h}$ for low values of h. Thus, sensitive features should appear in the top CLIME explanations when h is small, and innocuous features when h is large.

To test our hypothesis, we generated 200 CLIME explanations using discretized versions of the three datasets used by

(Slack et al. 2020). We computed the relative frequency of a feature appearing as the top explanation for different values of h. The results for the Recidivism dataset with 1 innocuous feature are depicted in Figure 3a (we defer results on other datasets and an extended discussion to Appendix B in the full version for lack of space). We plot the hamming distance h on the X-axis and the frequency of seeing a feature as the top explanation on the Y-axis. We see the sensitive feature 'race' appears as the top explanation for 60% of the instances explained when $h = 2$. In contrast, the innocuous feature (1st Unrelated Feature), appears almost exclusively as the top explanation when $h \geq 5$. For comparison, we depict results on CLIME explanations for a non-adversarial classifier in Fig. 3b. We see that the relative frequency with which a feature appears as the top explanation changes very little for different hamming distances. This confirms our hypothesis. In contrast, for LIME explanations (not shown), the innocuous feature appears as the top explanation for *all* 200 instances. We thus conclude that CLIME can not only avoid being fooled, but also potentially detect the adversarial natural of an opaque classifier by observing the change in the top features with and without Hamming distance constraints. While it may be possible to craft even more sophisticated attacks, these results clearly demonstrate CLIME's ability to detect adversarial attacks that exploit OOD sampling.

6 Related Work

Model explainability is one of the most important problems in machine learning. Therefore, there are a large number of recent surveys on the topic, e.g. (Hoffman and Klein 2017; Hoffman, Mueller, and Klein 2017; Lipton 2018; Adadi and Berrada 2018b; Guidotti et al. 2019; Rudin 2019). To overview, we partition related work into categories such as those that incorporate constraints, those that do or do not provide theoretical guarantees on the quality of the generated explanations as well as works that analyze the quality of explanations.

Explaining with Constraints. To the best of our knowledge, the work of (Deutch and Frost 2019) is the first and only approach to incorporate user-defined constraints into the explanation process explicitly. Other approaches like (Fong and Vedaldi 2017) allow some restrictions on the types of allowed perturbations (such as rotations and deletions for images), but are much more limited compared to the expressive power of a full-fledged formally defined constraint language. Shih, Choi, and Darwiche (2019), on the other hand, compile the Boolean function underlying the opaque classifier into a tractable datastructure that supports fast querying. This is inherently different from our approach in that it generates global explanations, and does not allow the user to craft and refine explanations according to their needs.

The approach of Deutch and Frost (Deutch and Frost 2019) is the closest to CLIME, yet differs in several ways. Firstly, they generate counterfactual explanations as the interpretable artifact, which are defined as the smallest changes to the input instance that make the opaque predictor label it differently. In contrast, CLIME is a feature attribution method that directly explains the relative importance of features contributing to a prediction. Further, Deutch and Frost (2019) employ a two-step 'perturb and project' method for incorporating constraints into the counterfactual generation phase, wherein the input instance is first perturbed in the direction of the target label without constraints, and is then projected on to the constrained space. This method is not guaranteed to converge and also imposes restrictions on the type of models it can explain. In contrast, CLIME directly samples perturbations satisfying the constraints, and is completely agnostic to the model being explained.

Explanations without theoretical guarantees. There were a number of approaches proposed to compute (model-agnostic) local explanations. We have overviewed LIME (Ribeiro, Singh, and Guestrin 2016) in Section 2. Anchor is a successor of LIME (Ribeiro, Singh, and Guestrin 2018). The main contribution of Anchor is to produce explanations that hold globally, for the entire distribution of inputs. SHAP (Lundberg and Lee 2017) is another popular model-agnostic explainer to produce local explanations. Similar to other explainers, SHAP does not provide any theoretical justification for the sampling procedure. However, SHAP employs game theoretic principles to produce an explainable model. Our work focuses on model-agnostic, local explanations, however, we produce explanations with provable guarantees. CXPlain proposes to train an additional 'explanation model' to provide explanations for a given ML model (Schwab and Karlen 2019). Learning of the explanation model involves an estimation of feature importance using a causal objective. The causal objective captures how input features cause a marginal improvement in the predictive performance of the ML model. In our work, we do not consider each feature individually and reason about the space of features as a whole. Moreover, our framework allows us to work with constrained spaces. Finally, works such as (Lakkaraju et al. 2019) provide limited capabilities for customizing global explanations by letting the user supply a set of features that the they deem important. Similar to (Björklund et al. 2019), they avoid sampling a neighbourhood around a given point by using original data points to construct an explainer. While avoiding sampling helps scalability, it also undermines applicability. For instance, dealing with user-defined constraints, as well as unbalanced or skewed input datasets can be problematic. In both cases, the input data may be too sparse to yield meaningful explanations. Recently, (Lakkaraju, Arsov, and Bastani 2020) demonstrated that these explanations are less robust compared to LIME, for example.

Another line of work in on gradient-based explainers, for example, saliency maps (Zeiler and Fergus 2014), Integrated Gradient (Sundararajan, Taly, and Yan 2017), DeepLIFT (Shrikumar, Greenside, and Kundaje 2017). Gradient-based methods assume full knowledge about the ML model and, also, require these models to be differentiable. While these methods are very efficient, they do not provide theoretical guarantees on the produced explanations. On top of that these approaches are not model-agnostic.

Explanations with theoretical guarantees. Recently, a formal approach to compute explanations of ML models was proposed. If an ML model allows a formal representation

in restricted fragment of the first order logic, then one can (a) define a formal notion of an explanation and (b) design an algorithm to produce these explanations (Shih, Choi, and Darwiche 2018, 2019; Ignatiev, Narodytska, and Marques-Silva 2019a; Darwiche and Hirth 2020; Darwiche 2020). One of the formal approaches is built on powerful knowledge compilation techniques, e.g. (Shih, Choi, and Darwiche 2018, 2019). The other approach employs very efficient formal reasoning engines, like SAT, SMT or ILP solvers, as a part of explanation generation algorithms (Ignatiev, Narodytska, and Marques-Silva 2019a; Narodytska et al. 2019). If the process of ML model compilation into a tractable structure is feasible then the first approach is very effective and allows the user to analyse the ML model efficiently. However, the compilation can be computationally expensive and resource demanding, so the second approach is more efficient in some applications. There are some limitations of these approaches. First, similar to gradient-based methods, they require full knowledge of the original ML model. Second, although for a number of ML models these approaches are shown to be practically effective (Ignatiev, Narodytska, and Marques-Silva 2019b; Izza, Ignatiev, and Marques-Silva 2020; Marques-Silva et al. 2020, 2021; Izza and Marques-Silva 2021; Ignatiev and Marques-Silva 2021; Huang et al. 2021; Ignatiev et al. 2022; Huang et al. 2022; Marques-Silva and Ignatiev 2022), formal approaches to XAI still face scalability issues in case of some other ML models (Ignatiev, Narodytska, and Marques-Silva 2019a) as formal reasoning about ML models is in general computationally expensive.

Quality of the explanations. Finally, we consider a recent line of work on analysis of the quality of explanations. (Ribeiro, Singh, and Guestrin 2018) proposed several heuristic measures to evaluate quality of explanations including fidelity and coverage, but do not provide a way to estimate the true value of these metrics. In (Ghorbani, Abid, and Zou 2019; Alvarez-Melis and Jaakkola 2018), it was shown using perturbation-based methods that explanations are susceptible to adversarial attacks and lack robustness property. For example, (Zhang et al. 2019) investigated several sources of uncertainty in LIME, like sampling variance in explaining a sample. The authors experimentally demonstrated that LIME often fails to capture the most important features locally. However, the paper does not propose a solution to remedy identified issues. Moreover, (Slack et al. 2020) showed that it is easy to fool an explainer, like LIME and SHAP, as we discussed in detail in Section 5. (Narodytska et al. 2019) presented a technique for evaluation quality of explanations based on model counting, but their approach suffers from scalability issues (as shown in Sec. 4) and is only applicable to BNNs. (Lakkaraju, Arsov, and Bastani 2020) proposed to use adversarial training (Madry et al. 2018) to improve robustness of the explanations. While the proposed approach improves robustness to adversarial attacks it cannot be easily extended to work in constraint environments and does not provide theoretical guarantees on the fidelity of the explanations. A related line of work on probabilistic verification of ML models has seen a surge in interest. (Albarghouthi et al. 2017) encoded the underlying model and fairness properties as formulas in SMT over real arithmentic, and relied on symbolic integration techniques. However, this approach is known not to scale, eg. it can only handle neural networks with a single hidden layer containing just three hidden units. (Bastani, Zhang, and Solar-Lezama 2019) present an alternative approach that uses Monte Carlo sampling and adaptive concentration inequalities. However, unlike Alg. 2, their method only returns a yes/no answer and does not provide a quantitative estimate. Further, their algorithm may fail to terminate on some inputs, and the sample complexity is not proven to be close-to-optimal.

7 Conclusions and Future Work

We presented a modular model-agnostic explanation framework CLIME that is able to operate on constrained subspaces of inputs. We introduced a new estimation algorithm that enables computation of an explanation's quality up to any desired accuracy. XAI is inherently human-centric, and in this light, our framework empowers the user to iteratively refine the explanation according to their needs. We demonstrated concrete scenarios where the user can zoom in to the input space guided by the fidelity metric.

The need for making XAI more rigorous and evidence-based has been highlighted in the past (Doshi-Velez and Kim 2017), and we believe our framework takes a concrete step in this direction. CLIME can also be readily extended in numerous ways. Helping the user with defining relevant subspaces by mining constraints from data is an interesting direction. Richer constraint languages like SMT (Barrett and Tinelli 2018) can provide even more flexibility, once sampling technology matures. Construction of CLIME's explainer model can also potentially be extended to incorporate Shapley values as in (Lundberg and Lee 2017).

Acknowledgements

This work was supported in part by NSF grants IIS-1527668, CCF-1704883, IIS-1830549, CNS-2016656, DoD MURI grant N00014-20-1-2787, an award from the Maryland Procurement Office, as well as the AI Interdisciplinary Institute ANITI, funded by the French program "Investing for the Future – PIA3" under Grant agreement no. ANR-19-PI3A-0004, and by the H2020-ICT38 project COALA "Cognitive Assisted agile manufacturing for a Labor force supported by trustworthy Artificial intelligence", and by National Research Foundation Singapore under its NRF Fellowship Programme [NRF-NRFFAI1-2019-0004].

References

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

Adadi, A.; and Berrada, M. 2018b. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6: 52138–52160.

Albarghouthi, A.; D'Antoni, L.; Drews, S.; and Nori, A. V. 2017. FairSquare: probabilistic verification of program fairness. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA): 1–30.

Alvarez-Melis, D.; and Jaakkola, T. S. 2018. On the Robustness of Interpretability Methods. *CoRR*, abs/1806.08049.

Amann, J.; Blasimme, A.; Vayena, E.; Frey, D.; and Madai, V. I. 2020. Explainability for artificial intelligence in healthcare: a multidisciplinary perspective. *BMC Medical Informatics and Decision Making*, 20(1): 1–9.

Angelino, E.; Larus-Stone, N.; Alabi, D.; Seltzer, M.; and Rudin, C. 2018. Learning Certifiably Optimal Rule Lists for Categorical Data. *Journal of Machine Learning Research*, 18: 234:1–234:78.

Arenas, M.; Barceló, P.; Bertossi, L.; and Monet, M. 2021. The Tractability of SHAP-Score-Based Explanations for Classification over Deterministic and Decomposable Boolean Circuits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 6670–6678.

Avellaneda, F. 2020. Efficient Inference of Optimal Decision Trees. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, 3195–3202. AAAI Press.

Barrett, C.; and Tinelli, C. 2018. Satisfiability modulo theories. In *Handbook of Model Checking*, 305–343. Springer.

Bastani, O.; Zhang, X.; and Solar-Lezama, A. 2019. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA): 1–27.

Björklund, A.; Henelius, A.; Oikarinen, E.; Kallonen, K.; and Puolamäki, K. 2019. Sparse robust regression for explaining classifiers. In *International Conference on Discovery Science*, 351–366. Springer.

Cadoli, M.; and Schaerf, A. 2005. Compiling problem specifications into SAT. *Artificial Intelligence*, 162(1-2): 89–120.

Chakraborty, S.; Fried, D.; Meel, K. S.; and Vardi, M. Y. 2015. From weighted to unweighted model counting. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Chavira, M.; and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7): 772–799.

Clarke, E.; Biere, A.; Raimi, R.; and Zhu, Y. 2001. Bounded Model Checking Using Satisfiability Solving. *Form. Methods Syst. Des.*, 19(1): 7–34.

Dagum, P.; Karp, R.; Luby, M.; and Ross, S. 2000. An optimal algorithm for Monte Carlo estimation. *SIAM Journal on Computing*, 29(5): 1484–1496.

Darwiche, A. 2020. Three Modern Roles for Logic in AI. *CoRR*, abs/2004.08599.

Darwiche, A.; and Hirth, A. 2020. On The Reasons Behind Decisions. *CoRR*, abs/2002.09284.

Deutch, D.; and Frost, N. 2019. Constraints-based explanations of classifications. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 530–541. IEEE.

Doshi-Velez, F.; and Kim, B. 2017. A Roadmap for a Rigorous Science of Interpretability. *CoRR*, abs/1702.08608.

Fong, R. C.; and Vedaldi, A. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, 3429–3437.

Ghorbani, A.; Abid, A.; and Zou, J. Y. 2019. Interpretation of Neural Networks Is Fragile. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 3681–3688. AAAI Press.

Guidotti, R.; Monreale, A.; Ruggieri, S.; Turini, F.; Giannotti, F.; and Pedreschi, D. 2019. A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv.*, 51(5): 93:1–93:42.

Gupta, R.; Sharma, S.; Roy, S.; and Meel, K. S. 2019. WAPS: Weighted and Projected Sampling. In *Proceedings of Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*.

Hoffman, R. R.; and Klein, G. 2017. Explaining Explanation, Part 1: Theoretical Foundations. *IEEE Intelligent Systems*, 32(3): 68–73.

Hoffman, R. R.; Mueller, S. T.; and Klein, G. 2017. Explaining Explanation, Part 2: Empirical Foundations. *IEEE Intelligent Systems*, 32(4): 78–86.

Hu, X.; Rudin, C.; and Seltzer, M. 2019. Optimal Sparse Decision Trees. In *Advances in Neural Information Processing Systems 32*, 7267–7275. Curran Associates, Inc.

Huang, X.; Izza, Y.; Ignatiev, A.; Cooper, M. C.; Asher, N.; and Marques-Silva, J. 2022. Tractable Explanations for d-DNNF Classifiers. In *AAAI*.

Huang, X.; Izza, Y.; Ignatiev, A.; and Marques-Silva, J. 2021. On Efficiently Explaining Graph-Based Classifiers. In *KR*, 356–367.

Ignatiev, A.; Izza, Y.; Stuckey, P.; and Marques-Silva, J. 2022. Using MaxSAT for Efficient Explanations of Tree Ensembles. In *AAAI*.

Ignatiev, A.; and Marques-Silva, J. 2021. SAT-Based Rigorous Explanations for Decision Lists. In *24th International Conference on Theory and Applications of Satisfiability Testing (SAT 2021)*, volume 12831 of *Lecture Notes in Computer Science*, 251–269. Springer.

Ignatiev, A.; Narodytska, N.; and Marques-Silva, J. 2019a. Abduction-Based Explanations for Machine Learning Models. In *AAAI*, 1511–1519.

Ignatiev, A.; Narodytska, N.; and Marques-Silva, J. 2019b. On Validating, Repairing and Refining Heuristic ML Explanations. *CoRR*, abs/1907.02509.

Izza, Y.; Ignatiev, A.; and Marques-Silva, J. 2020. On Explaining Decision Trees. *CoRR*, abs/2010.11034.

Izza, Y.; and Marques-Silva, J. 2021. On Explaining Random Forests with SAT. In *IJCAI*, 2584–2591.

Kohavi, R. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. In *KDD*, 202–207.

Lakkaraju, H.; Arsov, N.; and Bastani, O. 2020. Robust and Stable Black Box Explanations. *ICML 2020*.

Lakkaraju, H.; Kamar, E.; Caruana, R.; and Leskovec, J. 2019. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 131–138.

Lipton, Z. C. 2018. The mythos of model interpretability. *Queue*, 16(3): 31–57.

Lundberg, S. M.; and Lee, S. 2017. A Unified Approach to Interpreting Model Predictions. In *NIPS*, 4765–4774.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Marques-Silva, J.; Gerspacher, T.; Cooper, M. C.; Ignatiev, A.; and Narodytska, N. 2020. Explaining Naive Bayes and Other Linear Classifiers with Polynomial Time and Delay. In *NeurIPS*.

Marques-Silva, J.; Gerspacher, T.; Cooper, M. C.; Ignatiev, A.; and Narodytska, N. 2021. Explanations for Monotonic Classifiers. In *ICML*, 7469–7479.

Marques-Silva, J.; and Ignatiev, A. 2022. Delivering Trustworthy AI through formal XAI. In *AAAI*.

Marques-Silva, J.; Lynce, I.; and Malik, S. 2021. CDCL SAT Solving. In *Handbook of Satisfiability: Second Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, 133–182. IOS Press.

Meel, K. S.; Shrotri, A. A.; and Vardi, M. Y. 2019. Not all FPRASs are equal: demystifying FPRASs for DNF-counting. *Constraints*, 24(3-4): 211–233.

Moro, S.; Cortez, P.; and Rita, P. 2014. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62: 22–31.

Narodytska, N.; Shrotri, A. A.; Meel, K. S.; Ignatiev, A.; and Marques-Silva, J. 2019. Assessing Heuristic Machine Learning Explanations with Model Counting. In *SAT*, 267–278. Springer.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *KDD*, 1135–1144.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*.

Rudin, C. 2019. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. arXiv:1811.10154.

Schwab, P.; and Karlen, W. 2019. CXPlain: Causal Explanations for Model Interpretation under Uncertainty. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; dAlche Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 10220–10230. Curran Associates, Inc.

Shih, A.; Choi, A.; and Darwiche, A. 2018. A Symbolic Approach to Explaining Bayesian Network Classifiers. In *IJCAI*, 5103–5111.

Shih, A.; Choi, A.; and Darwiche, A. 2019. Compiling Bayesian Network Classifiers into Decision Graphs. In *AAAI*, 7966–7974.

Shrikumar, A.; Greenside, P.; and Kundaje, A. 2017. Learning Important Features Through Propagating Activation Differences. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, 3145–3153. PMLR.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *CoRR*, abs/1312.6034.

Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; and Lakkaraju, H. 2020. Fooling LIME and SHAP: Adversarial Attacks on Post Hoc Explanation Methods. In *AIES*, AIES '20, 180–186. New York, NY, USA: Association for Computing Machinery. ISBN 9781450371100.

Soos, M.; Gocht, S.; and Meel, K. S. 2020. Tinted, Detached, and Lazy CNF-XOR solving and its Applications to Counting and Sampling. In *Proceedings of International Conference on Computer-Aided Verification (CAV)*.

Soos, M.; and Meel, K. S. 2019. BIRD: Engineering an Efficient CNF-XOR SAT Solver and its Applications to Approximate Model Counting. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, 3319–3328. PMLR.

Valiant, L. G. 1984. A theory of the learnable. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, 436–445. ACM.

Van den Broeck, G.; Lykov, A.; Schleich, M.; and Suciu, D. 2021. On the tractability of SHAP explanations. In *Proceedings of AAAI*.

Wachter, S.; Mittelstadt, B.; and Russell, C. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31: 841.

Ying, R.; Bourgeois, D.; You, J.; Zitnik, M.; and Leskovec, J. 2019. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32: 9240.

Zeiler, M. D.; and Fergus, R. 2014. Visualizing and Understanding Convolutional Networks. In Fleet, D. J.; Pajdla, T.; Schiele, B.; and Tuytelaars, T., eds., *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, 818–833. Springer.

Zhang, Y.; Song, K.; Sun, Y.; Tan, S.; and Udell, M. 2019. " Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations. *arXiv preprint arXiv:1904.12991*.