Abstract Interpretation-Based Feature Importance for Support Vector Machines

Abhinandan Pal, Francesco Ranzato, Caterina Urban, Marco Zanella

Abstract Interpretation-based Feature Importance for Support Vector Machines

Abhinandan Pal[1][0000−0002−4122−5092], Francesco Ranzato[2][0000−0003−0159−0068], Caterina Urban[3][0000−0002−8127−9642], and Marco Zanella[2][0000−0002−6164−6169]

[1] School of Computer Science, University of Birmingham, United Kingdom
a.pal@bham.ac.uk
[2] Dipartimento di Matematica, University of Padova, Italy
{francesco.ranzato,marco.zanella}@unipd.it
[3] INRIA and Ecole Normale Supérieure | Université PSL, France
caterina.urban@inria.fr

Abstract. We study how a symbolic representation for support vector machines (SVMs) specified by means of abstract interpretation can be exploited for: (1) *enhancing the interpretability* of SVMs through a novel feature importance measure, called abstract feature importance (AFI), that does not depend in any way on a given dataset or the accuracy of the SVM and is very fast to compute; and (2) *certifying individual fairness* of SVMs and producing concrete counterexamples when this verification fails. We implemented our methodology and we empirically showed its effectiveness on SVMs based on linear and nonlinear (polynomial and radial basis function) kernels. Our experimental results prove that, independently of the accuracy of the SVM, our AFI measure correlates much strongly with stability of the SVM to feature perturbations than major feature importance measures available in machine learning software such as permutation feature importance, therefore providing better insight into the trustworthiness of SVMs.

1 Introduction

Machine learning (ML) software is increasingly being employed in high-stakes or sensitive applications. [11,26, etc.]. As a consequence, research in ML verification rapidly gained popularity [28,45], and the quest for interpretable ML models is becoming more and more pressing [43].

A fundamental and popular interpretability methodology is *feature importance*, that is, techniques for measuring the contribution of each input feature to a model prediction [5]. Nowadays, the most influential and used feature importance measures are Permutation Feature Importance (PFI) [6,18], Local Interpretable Model-agnostic Explanations (LIME) [38], and SHapley Additive exPlanations (SHAP) [29]. PFI observes the decrease in predictive performance when a feature value is randomly shuffled: an increased loss is indicative of how much that feature is important for the predictive model. LIME approximates the prediction model locally by training an interpretable surrogate model on points in a meaningful neighborhood around a given input. SHAP is a framework based on locally estimating so-called Shapley values [42]. The downsides of PFI and SHAP are that their outcome may greatly vary depending on the dataset and

may be misleading when features are correlated [25]. Furthermore, they also have a high computational cost when the number of features is large, even for small models. Moreover, the quality of the output of PFI strongly depends on the accuracy of the model. Notably, model variance to feature perturbations [23] and PFI are strongly correlated only when the model generalizes well. Similarly, it is unclear how to define a meaningful optimal neighborhood for LIME and this may lead to explanations that are unstable and manipulable. More importantly, LIME assumes that the decision boundary of the underlying model is locally linear, but there is no guarantee that this actually happens.

Contributions. In this work, we focus on the interpretability of Support Vector Machines (SVMs), which nowadays are used in extensive repertoire of critical and high-stakes applications such as credit card fraud detection, facial recognition, and melanoma classification [9]. In particular, we propose a novel feature importance measure for SVMs, called *Abstract Feature Importance* (AFI), that: (a) leverages an approximation of the underlying predictive model which is *formally correct-by-construction*; (b) does *not* depend on a given dataset or on model accuracy; and (c) is extremely *fast to compute*, independently of the number of features. We support both linear and nonlinear kernels, in particular the polynomial and radial basis function (RBF) kernels.

We derive our importance measure from a symbolic representation of a SVM based on *abstract interpretation* [12]. Specifically, the concrete vectors being manipulated by model computations are symbolically represented through an abstract domain, which defines their abstract counterparts and their data structure representations, as well as algorithms to process them by approximating the behaviour of operations, such as additions and dot products, used in model computations.

We leverage existing numerical abstract domains such as intervals (i.e., geometric hyperrectangles) [13] and affine forms (i.e., geometric zonotopes) [31] that we combine with a novel abstract domain tailored for precisely representing computations with one-hot encoded categorical input features. We show the effectiveness of this new combined abstract domain for certifying model stability against feature perturbations. In particular, we focus on verifying *individual fairness* [16] of SVMs, which, to the best of our knowledge, has not been investigated before. We evaluate our approach by certifying SVMs trained on the reference datasets in the literature on ML fairness [30] and by considering different similarity relations. Our approach is *sound by design*, meaning that an individually fair abstract representation of a SVM implies that this SVM is actually fair. Thus, the fraction of successful fairness verifications over a test dataset turns out to be a *lower bound* on the real individual fairness of a SVM. On the other hand, our approach is *incomplete*, as there are cases in which the SVM is fair whereas the verification of its abstract representation fails, due to imprecisions introduced by the abstraction process. Our third contribution is a method to leverage this abstract representation of SVMs to generate concrete counterexamples when the abstraction is unable to prove individual fairness, i.e., we deliver concrete similar inputs to a SVM that result in different classifications. The fraction of successful counterexample searches over a test dataset yields a lower bound on how biased an SVM is, and therefore, by complement, an *upper bound* on the real individual fairness of a SVM.

Finally, we conduct an extensive experimental comparison between our new feature importance measure AFI and popular interpretability methods like PFI and LIME. The

experimental results show that AFI is better correlated with stability of a SVM model to feature perturbations, independently of the accuracy of the model.

Related Work. Feature importance measures can be *local*, i.e., measuring feature importance for a specific prediction, or *global*, i.e., measuring importance over the entire input space of the predictive model. We also distinguish *model-agnostic* measures, which can be applied to any model, and *model-specific* measures. Finally, we classify importance measures in *performance-based*, i.e., measuring importance w.r.t. the predictive performance of the model (thus requiring knowledge of the ground truth values), and *effect-based*, measuring importance based on the magnitude of change in the predicted outcome due to changes in the feature value (requiring no knowledge of the ground truth values).

PFI [6,18] is a global, model-agnostic, performance-based measure. LIME [38] and SHAP [29] are both local, model-agnostic, effect-based measures. Our novel feature importance measure AFI is *specific for SVMs* but can be used *both* as *global and local* measure, is *effect-based* and is *dataset independent*, thus removing bias to the dataset. LIME can also be extended to a global measure by considering several local instances [38, Section 4]. However, the choice of how many and which instances to select remains not obvious, thus not providing correctness guarantees. By contrast, our AFI measure is based on a formally correct-by-construction approximation of the underlying predictive model.

Several other model-agnostic importance measures have been proposed in the literature. Prominent effect-based measures are visual tools such as partial dependence (PD) [19], individual conditional expectation (ICE) [22], and accumulated local effects (ALE) [4] plots. Visual tools, such as individual conditional importance (ICI) and partial importance (PI) curves [8], are also proposed for local performance-based measures. [8] proposes a Shapley feature importance measure, called SFIMP, that allows comparing feature importance across different models. Input gradient [24] is a local measure that can be both effect-based and performance-based. Feature importance measures specific for SVMs are typically limited to linear SVMs or face scalability issues w.r.t. the number of features, e.g. [10,32]. By contrast, our AFI measure supports nonlinear kernels and has no scalability issues.

Our work generally contributes to the research ecosystem around the verification of ML models using formal methods [2,28,45]. Most approaches consider (deep) neural networks [40,41,44,48, etc.], while here we focus on SVMs. Our work leverages a SVM verifier called SAVer [36]. In addition, we introduce here a more precise abstraction for one-hot encoded features that we integrated within SAVer. Our fairness analysis is in line with the approach investigated in [35], that evaluated the individual fairness of decision tree ensembles trained by a new fairness-aware learning technique. Similar works either consider an orthogonal notion of fairness or a different "threat model", in most cases both. [47] evaluates security against flipping a few labels to maximize classification error. [21] considers group and causal fairness metrics. [27] deals with robustness of SVMs against adversarial attacks. [17] proposes a new fairness metric where it adds a new feature with random values and bias individuals on this feature: the model is fair when it recovers the original labels. [34] puts forward a protocol to protect sensitive information

and trains a fair model using homomorphic encryption. [30,46] discuss several fairness metrics used to verify a variety of ML models.

2 Background

Support Vector Machines. SVMs [14] are supervised machine learning models based on separation curves that partition the input vector space into regions that best fit binary classification labels $L = \{-1, +1\}$. Separation curves are computed by maximizing their distance (margin) from the closest vectors in the training dataset. The simplest SVM is linear, which in its primal form boils down to a hyperplane $\mathbf{w}\cdot\mathbf{z} - b = 0$, where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are learned parameters, that determines whether an input $\mathbf{z} \in \mathbb{R}^n$ falls above/below (i.e., $\mathrm{sgn}(\mathbf{w}\cdot\mathbf{z} - b) = \pm 1$) w.r.t. the hyperplane. This approach is extended to nonlinear SVMs through a projection to a high-dimensional space via a kernel function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. More precisely, given an input space $X \subseteq \mathbb{R}^n$, a training dataset $T = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\} \subseteq X \times \{-1, +1\}$, a kernel function k and parameters $w_i, b \in \mathbb{R}$, with $i \in [1, N]$, a SVM classifier $C\colon \mathbb{R}^n \to \{-1, +1\}$ is represented in its dual form by the function $C(\mathbf{z}) \triangleq \mathrm{sgn}\left(\sum_{i=1}^{N} (w_i y_i k(\mathbf{x}_i, \mathbf{z})) - b\right)$. Most common kernels are: (i) *linear*, where $k(\mathbf{x}, \mathbf{z}) = \mathbf{x}\cdot\mathbf{z}$; (ii) *polynomial*, where $k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}\cdot\mathbf{z} + c)^p$, for some hyperparameters $c \in \mathbb{R}$ and $p \in \mathbb{N}$; (iii) *radial basis function* (RBF), where $k(\mathbf{x}, \mathbf{z}) = e^{-\gamma\|\mathbf{x}-\mathbf{z}\|_2^2}$, for some hyperparameter $\gamma > 0$. In multi-classification for labels $L = \{y_1, ..., y_m\}$, with $m > 2$, the standard approach is a reduction into multiple binary classification problems combined by leveraging a voting over different labels.

Example 2.1. Let us consider a space $X \subseteq \mathbb{R}^2$ of values normalized in the range $[-1, 1]$, thus $X = \{\mathbf{x} \in \mathbb{R}^2 \mid -1 \leq \mathbf{x}_1, \mathbf{x}_2 \leq 1\}$. A toy linear SVM $C\colon X \to \{-1, +1\}$ with two support vectors $\mathbf{v}_1 = (-0.5, 1), \mathbf{v}_2 = (0.5, -1) \in X$, respectively labeled as $-1, +1$, with weights $w_1 = w_2 = 0.5$ and bias $b = 0$, is represented in its dual form by the function $C(\mathbf{z}) = \mathrm{sgn}(-1 * 0.5(\mathbf{v}_1\cdot\mathbf{z}) + 1 * 0.5(\mathbf{v}_2\cdot\mathbf{z}))$. □

Abstract Interpretation. A tuple $\langle A, \sqsubseteq^A, \gamma^A\rangle$ is a *numerical abstract domain* (or abstraction) when $\langle A, \sqsubseteq^A\rangle$ is a partially ordered set of abstract values and $\gamma^A : A \to \wp(\mathbb{R}^n)$ is a concretization function which maps an abstract value a to the set $\gamma^A(a)$ of real vectors represented by a, and monotonically preserves the ordering relation, i.e., $a_1 \sqsubseteq^A a_2$ implies $\gamma^A(a_1) \subseteq \gamma^A(a_2)$. Intuitively, an abstract domain defines a symbolic representation of some sets of vectors in $\wp(\mathbb{R}^n)$: given $X \in \wp(\mathbb{R}^n)$ and $a \in A$, if $X \subseteq \gamma^A(a)$ then a is a *sound* representation, or approximation, of the set X, while if $X = \gamma^A(a)$ holds then a is called a *precise* (or *exact*) representation of X. We sometimes use the notation A_n for highlighting that A_n is an abstraction of sets of n dimensional vectors ranging in $\wp(\mathbb{R}^n)$.
Given a k-ary operation $f : (\mathbb{R}^n)^k \to \mathbb{R}^n$, for some $k > 0$, a corresponding abstract function $f^A : A^k \to A$ is a *sound* approximation of f on $(a_1, ..., a_k) \in A^k$ when $\{f(\mathbf{x}_1, ..., \mathbf{x}_k) \mid \mathbf{x}_i \in \gamma^A(a_i)\} \subseteq \gamma^A(f^A(a_1, ..., a_k))$ holds. Moreover, f^A is a *complete* approximation of f on its input $(a_1, ..., a_k)$ when equality holds. In words, soundness means that $f^A(a_1, ..., a_k)$ never misses a concrete computation of f on some input $(\mathbf{x}_1, ..., \mathbf{x}_k)$ abstractly represented by $(a_1, ..., a_k)$, while completeness entails that

$f^A(a_1, ..., a_k)$ is precisely a symbolic abstract representation of these concrete computations of f. When soundness/completeness of f^A holds for any abstract input then f^A is called a sound/complete approximation of f.

Abstract Domains. We consider the well-known abstract domain of *hyperrectangles* (or *intervals*) [12,13]. The hyperrectangle abstract domain HR_n consists of n-dimensional vectors of real intervals $h = \big([l_1, u_1], ..., [l_n, u_n]\big) \in \mathrm{HR}_n$, with lower and upper bounds $l_i, u_i \in \mathbb{R} \cup \{-\infty, +\infty\}$ such that $l_i \leq u_i$. Hence, the concretization function $\gamma^{\mathrm{HR}} : \mathrm{HR}_n \to \wp(\mathbb{R}^n)$ is defined by $\gamma^{\mathrm{HR}}(h) \triangleq \{\mathbf{x} \in \mathbb{R}^n \mid \forall i.\, l_i \leq \mathbf{x}_i \leq u_i\}$. Conversely, hyperrectangles also have an abstraction map $\alpha^{\mathrm{HR}} : \wp(\mathbb{R}^n) \to \mathrm{HR}_n$ that provides the smallest hyperrectangle approximating a set of vectors: $\alpha^{\mathrm{HR}}(X) \triangleq \big([\inf\{\mathbf{x}_i \in \mathbb{R} \mid \mathbf{x} \in X\}, \sup\{\mathbf{x}_i \in \mathbb{R} \mid \mathbf{x} \in X\}]\big)_{i=1}^{n}$. Abstract operations are defined by extending componentwise the following additions and multiplications of intervals:

$$[l_1, u_1] +^{\mathrm{HR}} [l_2, u_2] \triangleq [l_1 + l_2, u_1 + u_2],$$
$$[l_1, u_1] *^{\mathrm{HR}} [l_2, u_2] \triangleq [\min(l_1 l_2, l_1 u_2, l_2 u_1, l_2 u_2), \max(l_1 l_2, l_1 u_2, l_2 u_1, l_2 u_2)].$$

It is known that a compositional abstract evaluation on HR of an expression can be imprecise, e.g., the evaluations of the simple expressions $x - x$ and $x \cdot x$ on an input interval $[-c, c]$, with $c > 0$, yield, resp., $[-2c, 2c]$ and $[-c^2, c^2]$, rather than the exact intervals $[0, 0]$ and $[0, c^2]$. This *dependency problem* can trigger a significant source of imprecision for the hyperrectangle abstraction of a nonlinear (i.e., polynomial/RBF) SVM classifier. Thus, following [36], for our SVM abstract representations, we leverage the *reduced affine form* (RAF) abstraction, which is a domain of zonotopes representing dependencies between components of input vectors. A RAF for vectors in $\mathbb{R}^n$ is given by an expression $a_0 + \sum_{i=1}^{n} a_i \epsilon_i + a_r \epsilon_r$, where the ϵ_i's are symbolic variables ranging in the real interval $[-1, 1]$ representing a dependence from the i-th component of the vector, while ϵ_r is a further symbolic variable in $[-1, 1]$ that accumulates all the approximations introduced by nonlinear operations such as multiplications and exponentials. Thus, $\mathrm{RAF}_n \triangleq \{a_0 + \sum_{i=1}^{n} a_i \epsilon_i + a_r \epsilon_r \mid a_0, a_1, ..., a_n \in \mathbb{R},\, a_r \in \mathbb{R}_{\geq 0}\}$, where $a_r \in \mathbb{R}_{\geq 0}$ is the radius of the accumulative error of approximating all nonlinear operations during abstract computations. A RAF represents a real interval through the concretization map $\gamma^{\mathrm{RAF}} : \mathrm{RAF}_n \to \wp(\mathbb{R})$ defined by $\gamma^{\mathrm{RAF}}(a_0 + \sum_{i=1}^{n} a_i \epsilon_i + a_r \epsilon_a) \triangleq \{x \in \mathbb{R} \mid a_0 - \sum_{i=1}^{n} |a_i| - |a_r| \leq x \leq a_0 + \sum_{i=1}^{n} |a_i| + |a_r|\}$. RAFs are a restriction to a given length—in our case to the dimension n of $\mathbb{R}^n$—of the zonotope domain used in numerical program analysis [20]. Linear operations, namely additions and scalar multiplications, admit a complete approximation on the RAF abstraction, so that RAFs resolve the aforementioned dependency problem for linear expressions. Instead, nonlinear abstract operations, such as multiplications and exponentials, must still necessarily be approximated.

Example 2.2. Let us consider again the toy linear SVM C from Example 2.1. Its input space $X = [-1, 1]^2$ is abstracted as the RAF $a = (0 + \epsilon_1, 0 + \epsilon_2) \in (\mathrm{RAF}_2)^2$. □

Robustness. We consider an input space $X \subseteq \mathbb{R}^n$, a set of labels $L = \{y_1, ..., y_m\}$, and a dataset $T = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\} \subseteq X \times L$. A classifier trained on the dataset T is modeled as a map $C_T : X \to L$.

An adversarial region for an input sample $\mathbf{x} \in X$ is designated by a perturbation $P(\mathbf{x}) \subseteq X$ such that $\mathbf{x} \in P(\mathbf{x})$. Usually, a perturbation function $P : X \to \wp(X)$ is defined through a metric μ to measure similarity between inputs as their distance w.r.t. μ. The most common metric in ML [7] is induced by the ℓ_∞ maximum norm defined as $\|\mathbf{x}\|_\infty \triangleq \max\{|\mathbf{x}_1|, ..., |\mathbf{x}_n|\}$, so that the corresponding perturbation $P^\epsilon_\infty(\mathbf{x})$ includes all the vectors $\mathbf{z} \in X$ whose ℓ_∞ distance from $\mathbf{x}$ is bounded by a magnitude $\epsilon \in \mathbb{R}^+$, that is, $P^\epsilon_\infty(\mathbf{x}) \triangleq \{\mathbf{z} \in X \mid \|\mathbf{x} - \mathbf{z}\|_\infty \leq \epsilon\}$. Given a perturbation function P, a classifier C is *robust* (or *stable*) on $P(\mathbf{x})$, denoted by $\mathrm{robust}(C, P(\mathbf{x}))$, when for all $\mathbf{z} \in P(\mathbf{x})$, $C(\mathbf{z}) = C(\mathbf{x})$ holds. Robustness to a perturbation P is used as a major metric [23] to assess a classifier C on a test set $T \subseteq X \times L$ as follows: $\mathrm{rob}_T(C, P) \triangleq |\{(\mathbf{x}, y) \in T \mid \mathrm{robust}(C, P(\mathbf{x}))\}|/|T|$.

SAVer. Our work leverages SAVer (**S**VM **A**bstract **Ver**ifier), an automatic tool for robustness certification of SVMs [36,37]. Given a SVM $C : X \to L$, SAVer leverages an n-dimensional abstraction A_n of $\wp(\mathbb{R}^n)$ to first achieve a sound abstraction $P^\sharp(\mathbf{x}) \in A_n$ of a perturbation $P(\mathbf{x})$, i.e., $P(\mathbf{x}) \subseteq \gamma^A(P^\sharp(\mathbf{x}))$ must hold, and then applies sound abstract versions of the numerical functions used in C—notably, vector additions and dot products, scalar multiplications, exponentials—to design an abstract SVM classifier $C^\sharp : A \to \wp(L)$ that computes an over-approximation of the labels assigned to inputs in $P(\mathbf{x})$, i.e., $\{C(\mathbf{z}) \in L \mid \mathbf{z} \in P(\mathbf{x})\} \subseteq C^\sharp(P^\sharp(\mathbf{x}))$. If $C^\sharp(P^\sharp(\mathbf{x})) = \{y_i\}$, then every input in $P(\mathbf{x})$ is classified as y_i, so C is proved robust on the perturbation $P(\mathbf{x})$. In the binary classification case $L = \{-1, +1\}$, the abstract SVM $C^\sharp$ consists of an abstract function $\mathcal{A}^\sharp_C : A_n \to A_1$ that computes an over-approximation $\mathcal{A}^\sharp_C(P^\sharp(\mathbf{x}))$ of the distances between samples in $P^\sharp(\mathbf{x})$ and the SVM separation curve, namely, it computes an abstract value $a = \mathcal{A}^\sharp_C(P^\sharp(\mathbf{x})) \in A_1$ such that $\{\sum_{i=1}^N (w_i y_i k(\mathbf{x}_i, \mathbf{z})) - b \mid \mathbf{z} \in \gamma^{A_n}(P^\sharp(\mathbf{x}))\} \subseteq \gamma^{A_1}(a)$ holds. Afterwards, an over-approximation of the set of labels is inferred as follows:

$$
\begin{aligned}
C^\sharp(P^\sharp(\mathbf{x})) \triangleq\ & \textbf{if } \gamma^{A_1}(\mathcal{A}^\sharp_C(P^\sharp(\mathbf{x}))) \subseteq \mathbb{R}_{<0} \textbf{ then } \{-1\} \\
& \textbf{elseif } \gamma^{A_1}(\mathcal{A}^\sharp_C(P^\sharp(\mathbf{x}))) \subseteq \mathbb{R}_{>0} \textbf{ then } \{+1\} \\
& \textbf{else } \{-1, +1\}.
\end{aligned}
$$

Example 2.3. Let us consider again the toy linear SVM C from Example 2.1. The abstraction $\mathcal{A}^{\mathrm{RAF}}_C : (\mathrm{RAF}_n)^n \to \mathrm{RAF}_n$ of C in the RAF abstract domain is $\mathcal{A}^{\mathrm{RAF}}_C(a) = -1 * 0.5(\mathbf{v}_1 \cdot^{\mathrm{RAF}} a) + 1 * 0.5(\mathbf{v}_2 \cdot^{\mathrm{RAF}} a)$. By performing the abstract computations of $\mathcal{A}^{\mathrm{RAF}}_C$ on the abstraction $a \in (\mathrm{RAF}_2)^2$ of its input space $X = [-1, 1]^2$ from Example 2.2, we obtain:

$$
\begin{aligned}
\mathcal{A}^{\mathrm{RAF}}_C(a) &= -0.5(-0.5(0 + \epsilon_1) + 1(0 + \epsilon_2)) + 0.5(0.5(0 + \epsilon_1) - 1(0 + \epsilon_2)) \\
&= -0.5(0 + (-0.5)\epsilon_1 + \epsilon_2) + 0.5(0 + 0.5\epsilon_1 + (-1)\epsilon_2) \\
&= 0 + 0.5\epsilon_1 + (-1)\epsilon_2
\end{aligned}
$$

□

3 Methodology

We delve into the four primary contributions, each elaborated within distinct subsections. Figure 1 delineates these contributions, categorizing them into improvements to SAVer

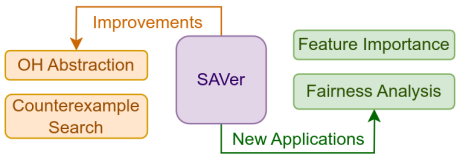

Fig. 1: Improvements to and new applications of SAVer

and applications thereof. Section 3.1 introduces a method for assessing feature importance, drawing upon the RAF abstract transformers embedded in SAVer. Section 3.2 refines the aforementioned abstract transformers for both interval and RAF abstract domains by introducing One-Hot constraints. Section 3.3 exemplifies how SAVer can be used to verify individual fairness of SVM models leveraging the most common kernel functions. Lastly, Section 3.4 shows how to extend SAVer to achieve a counterexample search, thus providing additional insights on the analysis of fairness.

3.1 Abstract Feature Importance

Let us define our central notion of abstract feature importance (AFI).

Definition 3.1 (Abstract Feature Importance). Let $C : \mathbb{R}^n \to L$ be a SVM classifier and let $\mathcal{A}_C^{\mathrm{RAF}} : (\mathrm{RAF}_n)^n \to \mathrm{RAF}_n$ be the abstraction of C in the RAF abstract domain. Let $\mathcal{A}_C^{\mathrm{RAF}}(f_1, ..., f_n) \triangleq a_0 + \sum_{i=1}^{n} a_i\epsilon_i + a_r\epsilon_r$ be the abstract computation output for an abstract input $(f_1, ..., f_n)$, with $f_i \in \mathrm{RAF}_n$. The importance of every input feature $i \in [1, n]$ is defined as the absolute value $|a_i| \geq 0$. □

This definition purposely ignores the accumulative error due to the approximations of all nonlinear operations performed by C, i.e., the term $a_r\epsilon_r$, influenced by all input features. When the input $(f_1, ..., f_n) \in (\mathrm{RAF}_n)^n$ abstracts the whole input space $X \subseteq \mathbb{R}^n$, AFI measures the *global* feature importance. Otherwise, AFI measures the *local* importance on the output label.

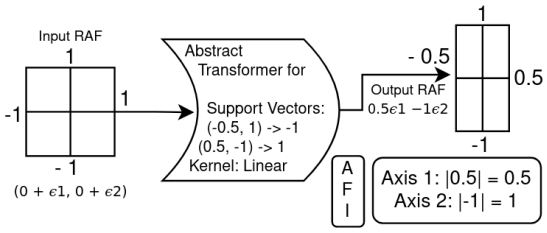

Fig. 2: Toy example of AFI for a linear SVM

Example 3.2. Let us consider again the toy linear SVM C from Example 2.1, and its abstract computation output $\mathcal{A}_C^{\mathrm{RAF}}(a) = 0 + 0.5\epsilon_1 + (-1)\epsilon_2$ from Example 2.3, for its abstract input $a \in (\mathrm{RAF}_2)^2$ from Example 2.2.
Based on Definition 3.1, we infer the importance indices $|a_1| = 0.5$ and $|a_2| = 1$ for, resp., $\mathbf{x}_1$ and $\mathbf{x}_2$, and conclude that $\mathbf{x}_2$ is twice as important as $\mathbf{x}_1$, as also shown by the picture in Figure 2.
Note that, since C is a linear SVM, it can be rewritten in primal form as: $C'(\mathbf{x}) = -0.5(\mathbf{v_1} \cdot \mathbf{x}) + 0.5(\mathbf{v_2} \cdot \mathbf{x}) = -0.5(-0.5\mathbf{x}_1 + \mathbf{x}_2) + 0.5(0.5\mathbf{x}_1 - \mathbf{x}_2) = 0.5\mathbf{x}_1 - \mathbf{x}_2 = (0.5, -1) \cdot \mathbf{x}$, thus obtaining an explicit weight $\mathbf{w} = (0.5, -1)$ for the input features, whose absolute values 0.5 and 1 coincide with our importance indices. □

For linear SVMs, as in the above example, the abstraction in the RAF abstract domain is *sound* and *complete* yielding an output RAF that matches the primal form of the SVM, ensuring the full accuracy of AFI. Instead, nonlinear SVMs cannot be represented in primal form and thus the output RAF acts as a pseudo-primal form, soundly over-approximating the true output region of the SVM. In this case, the accuracy of AFI hinges upon the precision of the SVM abstraction. Our work builds upon the RAF abstract transformers for linear and nonlinear (polynomial and RBF) kernels initially introduced within SAVer [36] and refines them by incorporating further constraints to better handle categorical input features, as elaborated in the next Section 3.2. Our experimental evaluation in Section 4 shows that our refined abstractions offer higher accuracy, meanwhile remaining computationally efficient. The potential introduction of more precise SVM abstractions in the future will directly benefit the accuracy of AFI.

Example 3.3. Let us consider a toy SVM C similar to that of Example 2.1, sharing the same support vectors, labels and weights but using using the polynomial kernel $k(\mathbf{z}, \mathbf{v}) \triangleq (\mathbf{z} \cdot \mathbf{v} + c)^d$, where $c = 1, d = 2$. Its abstract computation output turns out to be $\mathcal{A}_C^{\mathrm{RAF}}(a) = 0 + 1\epsilon_1 - 2\epsilon_2 + 1\epsilon_r$ for its abstract input $a \in (\mathrm{RAF}_2)^2$ from Example 2.2. We infer the importance indices $|a_1| = 1$ and $|a_2| = 2$ for, resp., $\mathbf{x}_1$ and $\mathbf{x}_2$, and, once again, infer that $\mathbf{x}_2$ is twice as important as $\mathbf{x}_1$. In this case, we also observe that the nonlinear noise accumulation term ϵ_r is greater than zero, meaning that some approximations happened through the computation, as expected for nonlinear kernels. □

Feature Grades. The AFI indices $a_i \geq 0$ (we assume that a_i implicitly denotes its absolute value) depend on the size of the abstract input, and this can make them harder to read and interpret, especially when the number of input features is high. To address this issue, we use a simple clustering strategy to assign an importance score to the input features. We consider the distribution of the AFI importance indices $(a_i)_{i=1}^n$ and compute its mean μ and standard deviation σ. Then, for each feature i, we consider its z-score $z_i \triangleq \frac{a_i - \mu}{\sigma}$, and the least integer greater than or equal to z_i as a corresponding rating, i.e., $score_i \triangleq \lceil \frac{a_i - \mu}{\sigma} \rceil$: this has the effect of standardizing the distribution into a normal distribution, slicing the distribution at every unit, and then labeling every slice with a progressive number. By doing so, features moderately influencing the result will have a score close to zero, relevant features will have higher scores, and those not influencing the outcome will have a negative score. Lastly, we shift and clip such distribution to achieve

what we call *feature grades* ranging, e.g., in an interval $[a, b]$, as obtained by an easy transformation: $grade_i \triangleq \max(\min(b, score_i + shift), a)$. For instance, let us consider a distribution of AFI indices for 10 features given by $(1, 6, 2, 5, 6, 1, 6, 7, 8, 9)$, where $\mu = 5.1$ and $\sigma = 2.85$: the corresponding scores are $(-1, 1, -1, 0, 1, -1, 1, 1, 2, 2)$, we shift them by $+6$ and then we clip these shifted values in $[3, 10]$, thus achieving the grades $(5, 7, 5, 6, 7, 5, 7, 7, 8, 8)$. Hence, e.g., x_1 and x_3 have similar impact on the classification, although they have different AFI indices.

Finally, let us remark that AFI does not require any knowledge on the ground truth values, nor the actual output of the SVM classifier, as AFI focuses on the computation process performed by the classifier, rather than how the result of such computation is used to assign a label to an input, thus making this approach applicable to scenarios where the correct prediction is not known in advance.

3.2 Abstracting One-Hot Encoding

Most ML algorithms need a way to represent categorical data in numerical form. Let $F = \{c_1, c_2, ..., c_k\}$ be the set of values of some categorical feature f. A naïve approach assigning a different number to each value in F introduces an unwanted ordering relation among features, that often induces bias or poor accuracy. A better and well-known method is *one-hot encoding*, that is, replacing f with k binary features $(x_1^f, x_2^f, ..., x_k^f) \in \{0, 1\}^k$ such that $\forall i \in [1, k]. x_i^f = 1 \Leftrightarrow f = c_i$. This sequence $(x_i^f)_{i=1}^k$ of k bits is also referred to as a *tier* of f. Numerical abstractions, such as the hyperrectangle and RAF abstract domains, are likely to suffer from a significant loss of precision when dealing with these one-hot encoded features, as they are not able to keep track of the relation $\sum_{i=1}^k x_i^f = 1$ existing between the binary features resulting from one-hot encoding.

Example 3.4. Consider a categorical feature $f \in F = \{red, green, blue\}$, and let $(x_r, x_g, x_b) \in \{0, 1\}^3$ denote the corresponding one-hot encoded tiers. Consider the set of categories $\{red, green\}$, represented by the set of tiers $X = \{(1, 0, 0), (0, 1, 0)\}$. The most precise hyperrectangle abstraction of X is $h = (x_r \in [0, 1], x_g \in [0, 1], x_b \in [0, 0]) \in \mathrm{HR}_3$. Observe that h also represents infinitely many vectors in $\mathbb{R}^3$ that do not belong to X and are illegal one-hot encodings, such as $(0.4, 0.8, 0)$, $(1, 1, 0)$ or $(0, 0, 0)$. □

To hinder this loss of precision, we define the *One-Hot abstraction*, a novel numerical abstraction tailored for one-hot encoded values.
Firstly, we recall the *constant propagation* abstract domain $\mathrm{CP} \triangleq \mathbb{R} \cup \{\bot^{\mathrm{CP}}, \top^{\mathrm{CP}}\}$, ordinarily used for the constant folding optimization by modern compilers [1]. CP is a flat domain whose partial order $\sqsubseteq^{\mathrm{CP}}$ is defined by $\bot^{\mathrm{CP}} \sqsubseteq^{\mathrm{CP}} z \sqsubseteq^{\mathrm{CP}} \top^{\mathrm{CP}}$, for all $z \in \mathbb{R}$. The concretization map $\gamma^{\mathrm{CP}} : \mathrm{CP} \to \wp(\mathbb{R})$ is as follows: $\gamma^{\mathrm{CP}}(z) \triangleq \{z\}$, for all $z \in \mathbb{R}$, meaning that a given numerical feature can only assume a constant value z; $\gamma^{\mathrm{CP}}(\top^{\mathrm{CP}}) \triangleq \mathbb{R}$ representing no constancy information; $\gamma^{\mathrm{CP}}(\bot^{\mathrm{CP}}) \triangleq \varnothing$ encodes unfeasibility. CP also has an abstraction map $\alpha^{\mathrm{CP}} : \wp(\mathbb{R}) \to \mathrm{CP}$ that provides the best (i.e. least w.r.t. the order $\sqsubseteq^{\mathrm{CP}}$) approximation in CP of a set of values, which is as follows: $\alpha^{\mathrm{CP}}(\varnothing) \triangleq \bot^{\mathrm{CP}}$, $\alpha^{\mathrm{CP}}(\{z\}) \triangleq z$, and $\alpha^{\mathrm{CP}}(X) \triangleq \top^{\mathrm{CP}}$ otherwise.

The One-Hot abstract domain for a k-dimensional, i.e. with k original categories, one-hot encoded feature space is $\mathrm{OH}_k \triangleq (\mathrm{CP} \times \mathrm{CP})^k$. Thus, abstract values are k-tuples of pairs of values in CP, that keep track of the numerical information originated from a single one-hot k-encoded feature, both when this was originally false, i.e. set to 0, or true, i.e. set to 1. Given $a \in \mathrm{OH}_k$ and one generic component $a_i \in \mathrm{CP} \times \mathrm{CP}$, with $i \in [1, k]$, let $a_{i,f/t} \in \mathrm{CP}$ denote, resp., the first/second element of a_i, i.e., $a_i = (a_{i,f}, a_{i,t})$. The partial order $\sqsubseteq$ of OH_k is induced componentwise by $\sqsubseteq^{\mathrm{CP}}$, namely, for all $a, b \in \mathrm{OH}_k$, $a \sqsubseteq b \Leftrightarrow \forall i \in [1, k].\, a_{i,f} \sqsubseteq^{\mathrm{CP}} b_{i,f}$ & $a_{i,t} \sqsubseteq^{\mathrm{CP}} b_{i,t}$. Then, for each component $i \in [1, k]$, the map $\hat{\gamma}_i : \mathrm{OH}_k \to \wp(\mathbb{R}^k)$ is defined as:

$$\hat{\gamma}_i(a) \triangleq \{\mathbf{x} \in \mathbb{R}^k \mid \mathbf{x}_i \in \gamma^{\mathrm{CP}}(a_{i,t}),\, \forall j \neq i.\, \mathbf{x}_j \in \gamma^{\mathrm{CP}}(a_{j,f})\}.$$

Therefore, $a \in \mathrm{OH}_k$ symbolically represents through $\hat{\gamma}_i$ the set of tiers whose i-th component was originally set to true. Note that if, for some $i \in [1, k]$, either $a_{i,f} = \bot^{\mathrm{CP}}$ or $a_{i,t} = \bot^{\mathrm{CP}}$, then $\hat{\gamma}_i(a) = \varnothing$. To retrieve all the concrete vectors represented by a, we collect all the vectors obtained by assuming that any component of the tier was originally set to true, namely, the concretization map $\gamma^{\mathrm{OH}_k} : \mathrm{OH}_k \to \wp(\mathbb{R}^k)$ is defined by $\gamma^{\mathrm{OH}_k}(a) \triangleq \cup_{i=1}^{k} \hat{\gamma}_i(a)$.

Example 3.5. Let us continue Example 3.4 by considering the abstract element $a = \big((0,1),(0,1),(0,\bot^{\mathrm{CP}})\big) \in \mathrm{OH}_3$. It turns out that a represents the set of tiers $X = \{(1,0,0),(0,1,0)\}$, which, in turn, is the one-hot encoding of $\{red, green\}$. In fact, its concretization is: $\gamma^{\mathrm{OH}_3}(a) = \hat{\gamma}_1(a) \cup \hat{\gamma}_2(a) \cup \hat{\gamma}_3(a) = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}_1 \in \{1\}, \mathbf{x}_2 \in \{0\}, \mathbf{x}_3 \in \{0\}\} \cup \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}_1 \in \{0\}, \mathbf{x}_2 \in \{1\}, \mathbf{x}_3 \in \{0\}\} \cup \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}_1 \in \{0\}, \mathbf{x}_2 \in \{0\}, \mathbf{x}_3 \in \varnothing\} = \{(1,0,0),(0,1,0)\}$. Thus, a precisely represents X. □

Example 3.5 is not fortuitous. In fact, for any set X of one-hot encoded tiers there always exists an abstract value a in OH which precisely represents this set, i.e., such that $\gamma^{\mathrm{OH}}(a) = X$.

Theorem 3.6. *If $X \in \wp(\mathbb{R}^k)$ is such that every vector of X is a one-hot encoded tier $(0,...,0,1,0,...0) \in \{0,1\}^k$, then the abstract value $a^X \in \mathrm{OH}_k$ defined as $a_i^X \triangleq \big(\alpha^{\mathrm{CP}}(\{\mathbf{x}_i \mid \mathbf{x} \in X, \mathbf{x}_i = 0\}), \alpha^{\mathrm{CP}}(\{\mathbf{x}_i \mid \mathbf{x} \in X, \mathbf{x}_i = 1\})\big)$, for all $i \in [1, k]$, precisely represents X.*

It is worth remarking that this abstraction OH allows us to represent one-hot encoded information in a compact way. For example, given three categorical features with five possible values each, an abstract value in OH consists of $5 + 5 + 5 = 15$ pairs of type $(a_{i,f}, a_{i,t}) \in \mathrm{CP}$, whereas the size of the concrete set of all the possible values for these three categorical features is $5^3 = 125$. In general, the size of an OH representation is in $O(NV)$ where N is the number of different categorical features and V is the maximum number of different categorical values for a given feature, whereas the size of the actual concrete values is in $O(V^N)$.

A value $a \in \mathrm{OH}_k$ such that, for all $i \in [1, k]$ and $u \in \{f, t\}$, $a_{i,u} \in \mathrm{CP} \setminus \{\top^{\mathrm{CP}}\}$ holds, is called *top-less*. It is important to note that the abstract value $a^X \in \mathrm{OH}_k$ defined in Theorem 3.6 is always top-less because the components of each pair a_i^X range in $\{0, 1, \bot^{\mathrm{CP}}\}$.

Given a numerical function $f : \mathbb{R} \to \mathbb{R}$, its sound abstract counterpart $f^{\mathrm{CP}} : \mathrm{CP} \to \mathrm{CP}$ on the CP abstraction is defined as follows:

$$f^{\mathrm{CP}}(a) \triangleq \begin{cases} \bot^{\mathrm{CP}} & \text{if } a = \bot^{\mathrm{CP}} \\ f(z) & \text{if } a = z \text{ for some } z \in \mathbb{R} \\ \top^{\mathrm{CP}} & \text{if } a = \top^{\mathrm{CP}} \end{cases}$$

In turn, f^{CP} allows us to define a sound abstract counterpart of f on our OH abstract domain.

Theorem 3.7. *A sound approximation of f on OH_k is the function $f^{\mathrm{OH}} : \mathrm{OH}_k \to \mathrm{OH}_k$ defined, for all $i \in [1, k]$, as follows:*

$$(f^{\mathrm{OH}}(a))_i \triangleq \big(f^{\mathrm{CP}}(a_{i,f}), f^{\mathrm{CP}}(a_{i,t})\big).$$

Example 3.8. Let us carry on Example 3.5. We assume a SVM with a polynomial kernel of degree two, whose abstract computation includes applying the nonlinear function $f(x) \triangleq x^2 - 3x + 1$ to each 0/1 component of the tiers in $\gamma^{\mathrm{OH}}(a) = \{(1,0,0),(0,1,0)\}$, so that $f(\gamma^{\mathrm{OH}}(a)) = \{(-1,1,1),\ (1,-1,1)\}$. By Theorem 3.7, we have that $a' \triangleq f^{\mathrm{OH}}(a) = \big((f^{\mathrm{CP}}(0), f^{\mathrm{CP}}(1)), (f^{\mathrm{CP}}(0), f^{\mathrm{CP}}(1)), (f^{\mathrm{CP}}(0), f^{\mathrm{CP}}(\bot^{\mathrm{CP}})\big) = \big((1,-1),$ $(1,-1), (1, \bot^{\mathrm{CP}})\big)$, whose concretization is:

$$\begin{aligned} \gamma^{\mathrm{OH}}(a') &= \hat{\gamma}_1(a') \cup \hat{\gamma}_2(a') \cup \hat{\gamma}_3(a') \\ &= \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}_1 \in \{-1\},\ \mathbf{x}_2 \in \{1\},\ \mathbf{x}_3 \in \{1\}\} \\ &\quad \cup \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}_1 \in \{1\},\ \mathbf{x}_2 \in \{-1\},\ \mathbf{x}_3 \in \{1\}\} \\ &\quad \cup \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}_1 \in \{1\},\ \mathbf{x}_2 \in \{1\},\ \mathbf{x}_3 \in \varnothing\} \\ &= \{(-1,1,1),(1,-1,1)\}. \end{aligned}$$

Notice that completeness holds because $f(\gamma^{\mathrm{OH}}(a)) = \gamma^{\mathrm{OH}}(f^{\mathrm{OH}}(a))$.
On the other hand, using the hyperrectangle abstraction, we noticed in Example 3.4 that $h = ([0,1],[0,1],[0,0]) \in \mathrm{HR}_3$ is the initial hyperrectangle representation of the set of tiers $\{(1,0,0),(0,1,0)\}$. By applying the abstract transfer function f^{HR} to h, we compute:

$$\begin{aligned} h' = f^{\mathrm{HR}}(h) = \big(& [0,1]^{2^{\mathrm{HR}}} - 3 \cdot^{\mathrm{HR}} [0,1] +^{\mathrm{HR}} [1,1], \\ & [0,1]^{2^{\mathrm{HR}}} - 3 \cdot^{\mathrm{HR}} [0,1] +^{\mathrm{HR}} [1,1], \\ & [0,0]^{2^{\mathrm{HR}}} - 3 \cdot^{\mathrm{HR}} [0,0] +^{\mathrm{HR}} [1,1]\big) \\ = \big(& [-2,2],[-2,2],[1,1]\big). \end{aligned}$$

Thus, we have that $\gamma^{\mathrm{HR}}(h') = \{\mathbf{x} \in \mathbb{R}^3 \mid -2 \le \mathbf{x}_1, \mathbf{x}_2 \le 2,\ \mathbf{x}_3 = 1\}$. This latter concrete set of vectors is much larger than $\gamma^{\mathrm{OH}}(f^{\mathrm{OH}}(a))$: while soundness is nevertheless guaranteed in HR, we have lost a good deal of information due to the imprecision of the interval abstraction. In particular, observe that for $\mathbf{x}_1$ and $\mathbf{x}_2$, OH was able to derive precisely their values ranging in $\{-1,+1\}$, while the HR analysis

computes much less precise lower/upper bounds for $\mathbf{x}_1$ and $\mathbf{x}_2$ such as -2 and 2, as a consequence of the abstract computation:

$$[0,1]^{2^{\mathrm{HR}}} - 3 \cdot^{\mathrm{HR}} [0,1] +^{\mathrm{HR}} [1,1] = [0,1] + [-3,0] + [1,1] = [-2,2].$$

For instance, the lower bound -2 for $\mathbf{x}_1$ is obtained by adding the lower bounds 0 (for $\mathbf{x}_1^2$), -3 (for $-3\mathbf{x}_1$), 1 (for $+1$), namely, by requiring that $\mathbf{x}_1$ simultaneously assumes both values 0 and 1, which is an unfeasible spurious case.
The relational RAF abstraction induces more precise approximations w.r.t. HR, although this increase of precision is limited to linear dependencies only, including the intermediate values of the abstract computations. Thus, the OH abstraction shows a clear precision gain over both domains HR and RAF for abstract computations on one-hot encoded vectors. □

In Example 3.8, we observed that f^{OH} is a complete approximation of f on a. This is a consequence of the following general result.

Corollary 3.9. *Let $a \in \mathrm{OH}_k$ be top-less. Then:* (i) *f^{OH} is a complete abstraction of f on a;* (ii) *Given $f_1, f_2, ..., f_p : \mathbb{R} \to \mathbb{R}$, $f_1^{\mathrm{OH}} \circ f_2^{\mathrm{OH}} \circ ... \circ f_p^{\mathrm{OH}}$ is a complete abstraction of $f_1 \circ f_2 \circ ... \circ f_p$ on a.*

Using OH in SAVer. We implemented in SAVer our OH abstraction for categorical features on top of the interval and RAF abstract domains for numerical features: these instances of SAVer are denoted in Section 4, resp., by Interval+OH and RAF+OH. In what follows, we focus on the RAF abstraction, since the interval domain conceptually follows the same pattern. Assume a vector of numerical features $\mathbf{x} \in \mathbb{R}^q$ and, for simplicity, a single categorical feature f having k categories[4]. We first perform the one-hot encoding of f, that is, f is transformed into a tier in $\{0,1\}^k$. We consider the so-called CAT perturbations of a categorical feature f (cf. Section 4), where f may take the value of any of its k categories: e.g., the CAT perturbation of $color$ in Example 3.4 is $\{red, green, blue\}$. This means that in the CAT perturbation of the one-hot encoding of f, we allow every binary feature of the tier representing f to be either 0 or 1, so that the corresponding perturbed abstract value is always of the shape $oh = (0^{\mathrm{CP}}, 1^{\mathrm{CP}})^k \in \mathrm{OH}_k$. For the numerical features $\mathbf{x}$, we consider a so-called NOISE perturbation (cf. Section 4), i.e., a maximum norm perturbation $P_\infty^\epsilon(\mathbf{x})$ for some magnitude $\epsilon > 0$. Hence, these NOISE and CAT perturbations define an initial RAF value $a \in \mathrm{RAF}_q$ for $\mathbf{x}$ and an initial OH value $oh \in \mathrm{OH}_k$ for f. The abstract kernel of the SVM is then computed on the RAF_q domain for $\mathbf{x}$ and on the OH_k domain for f, and we assume that the output of this abstract computation is $\langle a', oh' \rangle \in \mathrm{RAF}_q \times \mathrm{OH}_k$. For our purpose of certifying the robustness, the output OH value oh' is converted back to a hyperrectangle representing the lower and upper bounds of the k one-hot encoded components of f, namely, we compute k intervals $\alpha^{\mathrm{HR}_k}(\gamma^{\mathrm{OH}_k}(oh')) = \langle [\alpha_j, \beta_j] \rangle_{j=1}^k \in \mathrm{HR}_k$ that provide lower and upper bounds of the numerical contributions to the SVM output of the k one-hot encoded components of f. On the other hand, the output $a' \in \mathrm{RAF}_q$ is converted to

[4] For multiple categorical features, we keep track of the relation between all the categorical features and their corresponding tiers through a global lookup table

the interval $[l, u] = \gamma^{\mathrm{RAF}}(a') \in \mathrm{HR}_1$ that gives a lower and upper bound to the sum of the contributions to the SVM output of the q numerical features $\mathbf{x}$. Hence, the sum of these $k+1$ intervals provides a single interval $[(l + (\sum_{j=1}^{k} \alpha_j), u + (\sum_{j=1}^{k} \beta_j)] \in \mathrm{HR}_1$, which is a sound approximation of the sign of the SVM output.

Example 3.10. Consider a vector $\mathbf{x} \in \mathbb{R}^5$ where $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}$ are two numerical features and $\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5 \in \{0, 1\}$ are three binary values deriving from the one-hot encoding of a categorical feature. Assume an input perturbation of these features such that the output of the abstract computation for the numerical features is given by $a' = 1 + \epsilon_1 + 1.5\epsilon_2 + 0.5\epsilon_r \in \mathrm{RAF}_2$, which is converted into the interval $\gamma^{\mathrm{RAF}}(a') = [-2, 4] \in \mathrm{HR}$, while on for the three one-hot encoded features $\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5$, the output on OH is $oh' = \big((2, -1), (3, \perp^{\mathrm{CP}}), (3, 2)\big) \in \mathrm{OH}_3$. Thus, we have that $\gamma^{\mathrm{OH}_3}(oh') = \{(-1, 3, 3), (2, 3, 2)\}$, which is then abstracted to the hyperrectangle $\alpha^{\mathrm{HR}_3}(\gamma^{\mathrm{OH}_3}(oh')) = ([-1, 2], [3, 3], [2, 3]) \in \mathrm{HR}_3$. Hence, we derive a range interval of the SVM output which is $[-2, 4] +^{\mathrm{HR}} [-1, 2] +^{\mathrm{HR}} [3, 3] +^{\mathrm{HR}} [2, 3] = [2, 12]$. Since $\gamma^{\mathrm{HR}_1}([2, 12]) \subseteq \mathbb{R}_{>0}$, we have that $\mathrm{sgn}([2, 12]) = +1$, meaning that the SVM classification is certified to be robust for the input perturbation. □

3.3 Individual Fairness

Several formal models of fairness have been investigated in the literature. Dwork et al. [16] point out several weaknesses of the notions of group fairness and therefore study *individual fairness* defined as "the principle that two individuals who are similar w.r.t. a particular task should be classified similarly". This is formalized as a Lipschitz condition of the classifier, that is, by requiring that two individuals $\mathbf{x}, \mathbf{y} \in X$ whose distance is $\delta(\mathbf{x}, \mathbf{y}) \geq 0$, are mapped, resp., to distributions $D_{\mathbf{x}}$ and $D_{\mathbf{y}}$ whose distance is at most $\delta(\mathbf{x}, \mathbf{y})$. Intuitively, the output distributions for $\mathbf{x}$ and $\mathbf{y}$ are indistinguishable up to their distance. Several distance metrics $\delta \colon X \times X \to \mathbb{R}_{\geq 0}$ can be used in this context, where Dwork et al. [16] study the total variation or relative ℓ_∞ distances.
Following [16], a classifier $C \colon X \to L$ is (*individually*) *fair* when C outputs the same label for all pairs of individuals $\mathbf{x}, \mathbf{y} \in X$ satisfying a similarity relation $S \subseteq X \times X$ between input samples. This relation S can be derived from a distance δ as follows: $(\mathbf{x}, \mathbf{y}) \in S \Leftrightarrow \delta(\mathbf{x}, \mathbf{y}) \leq \epsilon$, where $\epsilon \in \mathbb{R}$ is a similarity threshold.

Definition 3.11 (Individual Fairness). A classifier $C \colon X \to L$ is *fair* on an individual $\mathbf{x} \in X$ w.r.t. a similarity relation $S \subseteq X \times X$, denoted by $\mathrm{fair}(C, \mathbf{x}, S)$, when $\forall \mathbf{z} \in X. (\mathbf{x}, \mathbf{z}) \in S \Rightarrow C(\mathbf{z}) = C(\mathbf{x})$. □

To define a fairness metric for a classifier C, we compute how often C is fair on sets of similar individuals in a test set $T \subseteq X \times L$:

$$\mathrm{fair}_{T,S}(C) \triangleq |\{(\mathbf{x}, y) \in T \mid \mathrm{fair}(C, \mathbf{x}, S)\}|/|T|.$$

Hence, individual fairness for a similarity relation S boils down to robustness on the perturbation $P_S(\mathbf{x}) \triangleq \{\mathbf{z} \in X \mid (\mathbf{x}, \mathbf{z}) \in S\}$ induced by S, i.e., for all $\mathbf{x} \in X$, $\mathrm{fair}(C, \mathbf{x}, S) \Leftrightarrow \mathrm{robust}(C, P_S(\mathbf{x}))$ holds.

3.4 Searching for Counterexamples

The abstract interpretation framework described in Section 2 is sound, thus a classifier C certified to be robust over a region $P(\mathbf{x})$ guarantees that all the inputs $\mathbf{x}' \in P(\mathbf{x})$ actually receive the same label. The converse is generally not true for nonlinear kernels, due to lack of completeness: when the abstract verification is unable to certify robustness, it may be either due to a loss of precision or to a vector in $P(\mathbf{x})$ which truly receives a different label. We refer to the latter as a *counterexample* to the robustness of $\mathbf{x}$. In case of an inconclusive analysis, we can mitigate the effect of incompleteness by searching for counterexamples: if at least one is found, then the classifier can be marked as being *not robust*. Catching a counterexample within a possibly infinite set of vectors, however, is a daunting task. Let $a \in \mathrm{RAF}_n$ be a sound abstraction for $P(\mathbf{x})$, C a classifier, and $\mathcal{A}_C^{\mathrm{RAF}}$ its abstraction on RAF. We define an informed heuristic search approach leveraging our AFI measure as follows.

Definition 3.12 (Counterexample Search).
(S_1) Let $a_{\mathrm{out}} \triangleq a_0 + \sum_{i=1}^{n} a_i\epsilon_i + a_r\epsilon_r$ be the output of the abstract computation $\mathcal{A}_C^{\mathrm{RAF}}(a)$ on RAF (cf. Section 3.1);
(S_2) If $C(\mathbf{x}) < 0$, we look for a potential counterexample $\mathbf{x}^*$ by maximizing a_{out}, i.e., by selecting the maximum possible value for every $\mathbf{x}_i$ when $a_i > 0$, and the minimum when $a_i < 0$ (the converse if $C(\mathbf{x}) > 0$);
(S_3) If $C(\mathbf{x}^*) \neq C(\mathbf{x})$, then $\mathbf{x}^*$ is a counterexample for $\mathbf{x}$, and the classifier C is proved not robust on $\mathbf{x}$;
(S_4) Otherwise, we select the most influential feature $\mathbf{x}_M$, and its mean value in $P(\mathbf{x})$, which is defined by $m \triangleq \big(\min\{\mathbf{y}_M \mid \mathbf{y} \in P(\mathbf{x})\} + \max\{\mathbf{y}_M \mid \mathbf{x} \in P(\mathbf{x})\}\big)/2$, and we partition $P(\mathbf{x})$ using the cutting hyperplane $\mathbf{x}_M \leq m$, thus obtaining left and right subsets $P_l(\mathbf{x}), P_r(\mathbf{x}) \subseteq P(\mathbf{x})$;
(S_5) We consider $a_l, a_r \in \mathrm{RAF}_n$ abstracting, resp., $P_l(\mathbf{x}), P_r(\mathbf{x})$, and we recursively repeat from step (S_1) until a counterexample is found, or a user-defined timeout is met. □

Maximization in step (S_2) requires additional care for one-hot encoded features, as exactly one of them must be set to 1: we set to 1 the most influential feature only. We also observe that computing a_l, a_r during step (S_5) does not introduce losses of precision, as RAFs represent hyperrectangles exactly, and partitioning a RAF by a cutting hyperplane of the form $\mathbf{x}_i \leq k$ yields two smaller hyperrectangles. Steps (S_1) and (S_2) correspond to looking for a counterexample in the vertices of the hyperrectangle represented by a, which have the greatest distance from the center and are therefore intuitively more likely to exhibit different labels. Since a hyperrectangle in $\mathbb{R}^n$ has 2^n vertices, it is not feasible to check them all, and we thus use our feature importance analysis to infer a gradient pointing towards the most promising one. If no counterexample is found, it may be due to the separation curve of C crossing $P(\mathbf{x})$ while leaving all the vertices on the same side. We therefore proceed to steps (S_4) and (S_5) that partition $P(\mathbf{x})$ into two smaller subsets $P_l(\mathbf{x})$ and $P_r(\mathbf{x})$ by cutting the former space in half along the axis of the most influential feature, and recursively repeating the process on the two components. Should a counterexample be found, C can be definitely marked as not robust. Otherwise, we set a timeout mechanism, such as a limit on the recursion depth, to avoid divergence.

Example 3.13. Let $\mathbf{s} = (0, -\sqrt{2}), \mathbf{t} = (-1, 1), \mathbf{v} = (1, 1)$ be the support vectors of a SVM with polynomial kernel $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^2$, $w_{\mathbf{s}} = -1$, $w_{\mathbf{t}} = w_{\mathbf{v}} = 1$ their weights, and $b = 0$ the bias. The SVM classifier is therefore defined as $C(\mathbf{x}) \triangleq -(\mathbf{s} \cdot \mathbf{x} + 1)^2 + (\mathbf{t} \cdot \mathbf{x} + 1)^2 + (\mathbf{v} \cdot \mathbf{x} + 1)^2$, and can be rewritten to its primal form $C(\mathbf{x}) = 2\mathbf{x}_1^2 + 2(2 + \sqrt{2})\mathbf{x}_2 + 1$, thus highlighting the separation curve as the parabola $\Gamma \triangleq \mathbf{x}_2 = -\frac{1}{2+\sqrt{2}}\mathbf{x}_1^2 - \frac{1}{2(2+\sqrt{2})}$. We now consider $\mathbf{x}' = (0.5, -0.5)$, and let $P(\mathbf{x}')$ be the hyperrectangle of radius 0.5 centered in $\mathbf{x}'$, that is, $P(\mathbf{x}') \triangleq \{\mathbf{x} \in \mathbb{R}^2 \mid 0 \leq \mathbf{x}_1 \leq 1, -1 \leq \mathbf{x}_2 \leq 0\}$. We observe that $C(\mathbf{x}') \approx -1.91 < 0$, whereas every point $\mathbf{x} \in P(\mathbf{x}')$ having $\mathbf{x}_2 = 0$ evaluates to $2\mathbf{x}_1^2 + 1 > 0$, which is always positive, hence C is not robust over $P(\mathbf{x}')$. The parabola Γ crosses $P(\mathbf{x})$ leaving some vertices on different sides of the space.
We consider $a = (0.5 + 0.5\epsilon_1, -0.5 + 0.5\epsilon_2) \in \mathrm{RAF}_2^2$ that represents the perturbation $P(\mathbf{x}')$ exactly, and compute $a_{\text{out}} = c + 0.5\epsilon_1 + (1 + \sqrt{2})\epsilon_2 + d\epsilon_r$, where the values of the center $c \in \mathbb{R}$ and the nonlinear accumulation term $d \in \mathbb{R}$ are omitted for simplicity, as they are not relevant for our purposes. Thus, the abstract feature importance vector is $(0.5, 1 + \sqrt{2})$ whose components are both positive. Since $C(\mathbf{x}') < 0$, we are looking for positive counterexamples and, following step (S_3) above, we select the candidate counterexample $\mathbf{x}^* \in P(\mathbf{x}')$ by considering the maximum values for $\mathbf{x}_1, \mathbf{x}_2$ for vectors ranging in $P(\mathbf{x}')$, that is, $\mathbf{x}^* = (1, 0)$. Then, it turns out that $C(\mathbf{x}^*) = 2(1)^2 + 2(2 + \sqrt{(2)}0 + 1 = 3 > 0$, so that $\mathbf{x}^*$ is indeed a counterexample that makes C not robust on $P(\mathbf{x}')$. □

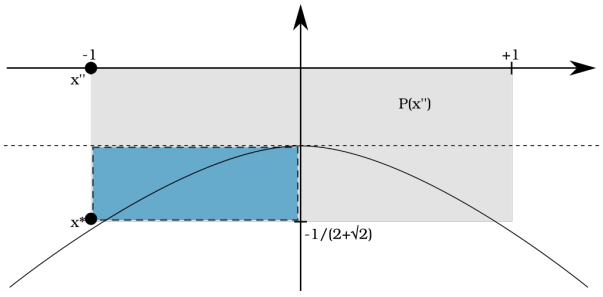

Fig. 3: Example of counterexample search

Example 3.14. Continuing with Example 3.13, we consider $\mathbf{x}'' = (-1, 0)$ and the region $P(\mathbf{x}'') = \{\mathbf{x} \in \mathbb{R}^2 \mid -1 \leq \mathbf{x}_1 \leq +1, -\frac{1}{2+\sqrt{2}} \leq \mathbf{x}_2 \leq 0\}$, as depicted in Figure 3. Observe that every vertex of $P(\mathbf{x}'')$ is on the same side of Γ and thus receives the same class $+1$, while the subregion of $P(\mathbf{x}'')$ below the parabola lays on the other side, making C not robust. We iterate the search of counterexamples illustrated in Example 3.13, providing the AFI vector $\mathbf{i} = (0, \frac{3+2\sqrt{2}}{(2+\sqrt{2})^2}) = (0, 0.5)$, which can be viewed as a gradient vector guiding the counterexample search. In this case, we look for a counterexample $\mathbf{x}^* \in P(\mathbf{x}'')$ such that $C(\mathbf{x}^*) < 0$, hence moving in the opposite direction w.r.t. the gradient vector $\mathbf{i}$. By doing so, we consider the bottom-left corner

Dataset	#Features	Training Set		Test Set	
		Size	Positive	Size	Positive
Adult	103	30162	24.9%	15060	24.6%
Compas	371	4222	53.3%	1056	55.6%
German	56	800	69.8%	200	71.0%

Table 1: Reference datasets.

$\mathbf{x}^* = (-1, -\frac{1}{2+\sqrt{2}})$ of $P(\mathbf{x}'')$ such that $C(\mathbf{x}^*) = 1$, thus meaning that we did not compute a counterexample. We therefore follow the steps (S_4) and (S_5) by partitioning $P(\mathbf{x}'')$ through the hyperplane $\mathbf{x}_2 \leq -\frac{1}{2(2+\sqrt{2})}$, as shown by the dotted line in Fig. 3, and then starting the recursive search. After the first recursive step, none of the two new (hyper)rectangles have counterexamples in their vertices, so another recursive step is applied to the lower rectangle which is partitioned by the hyperplane $\mathbf{x}_1 \leq 0$. This generates two new (hyper)rectangles, and both have now counterexamples in their vertices, which can be found through steps (S_1) and (S_2), therefore proving, through a counterexample witness, that C is not robust. □

4 Experimental Evaluation

We consider the main reference datasets used in the fairness literature [30]:(i) **Adult** [15], which labels yearly incomes, above or below 50K US$, based on personal attributes; (ii) **Compas** [3], which labels recidivism risk based on personal attributes and criminal history; (iii) **German** [15], which labels good/bad credit scores. Table 1 displays size and distribution of positive samples for these datasets. The data is preprocessed according to [41]. Some of these datasets exhibit a highly unbalanced label distribution, leading to high accuracy and 100% individual fairness for a constant classifier like $C(\mathbf{x}) = 1$. Thus, following [41], we report in Table 2, for several SVM models, both accuracy and balanced accuracy, i.e., $\frac{1}{2}\left(\frac{\mathit{truePos}}{\mathit{truePos}+\mathit{falseNeg}} + \frac{\mathit{trueNeg}}{\mathit{trueNeg}+\mathit{falsePos}}\right)$.

Similarity Relations. Let $I \subseteq \mathbb{N}$ be a set of indices of features after one-hot encoding, and $\mathbf{x}, \mathbf{y} \in X$ be two individuals. Following [41], we consider three similarity relations.
NOISE: Given a subset of numerical features $I' \subseteq I$, let $S_{\mathit{noise}}(\mathbf{x}, \mathbf{y})$ iff $|\mathbf{x}_i - \mathbf{y}_i| \leq \epsilon$ for all $i \in I'$, and $\mathbf{x}_i = \mathbf{y}_i$ for all $i \in I \setminus I'$. This means that all the features of $\mathbf{x}$ in I' are subject to a maximum norm perturbation $P^{\epsilon}_{\infty}(\mathbf{x})$. For our experiments, we consider $\epsilon = 0.05$, namely, a $\pm 5\%$ perturbation for data normalised to $[0, 1]$.
CAT: Given a subset of sensitive categorical attributes $I' \subseteq I$, let $S_{\mathit{cat}}(\mathbf{x}, \mathbf{y})$ iff $\mathbf{x}_i = \mathbf{y}_i$ for all $i \in I \setminus I'$. For Adult and German, we select the gender attribute, while for Compas, the race attribute.
NOISE-CAT: $S_{\mathit{noise\text{-}cat}}(\mathbf{x}, \mathbf{y}) \triangleq S_{\mathit{noise}}(\mathbf{x}, \mathbf{y}) \vee S_{\mathit{cat}}(\mathbf{x}, \mathbf{y})$.

Setup. We trained the SVMs used in our experiments with scikit-learn. Hyperparameters were chosen by hit-and-trial and observing trends. The final hyperparameters for each kernel and dataset were those that led to SVMs with high balanced accuracy.

Table 2: Accuracy, balanced accuracy, and verified individual fairness percentages leveraging different abstractions.

Dataset	**SVM Kernel**	**Acc.**	**Bal. Acc.**	**Interval**			**Interval+OH**			**RAF**			**RAF+OH**		
				N	C	NC	N	C	NC	N	C	NC	N	C	NC
Adult	L(1)	84.6	75.6	**96.5**	**95.2**	**91.6**	**96.5**	**95.2**	**91.6**	**96.5**	**95.2**	**91.6**	**96.5**	**95.2**	**91.6**
	R(0.05,0.01)	83.8	72.0	2.4	2.8	0.0	2.4	2.8	0.0	94.8	42.4	37.2	**97.5**	**97.9**	**95.4**
	P(0.01,3,3)	83.9	76.7	0.0	0.0	0.0	0.0	0.0	0.0	**0.5**	**0.5**	**0.03**	**0.5**	**0.5**	**0.03**
Compas	L(1)	64.7	64.1	**95.5**	**99.5**	**94.9**	**95.5**	**99.5**	**94.9**	**95.5**	**99.5**	**94.9**	**95.5**	**99.5**	**94.9**
	R(0.05,0.01)	64.5	63.1	54.3	42.5	1.0	54.3	42.5	1.0	91.8	71.6	66.9	**94.4**	**97.5**	**89.3**
	P(0.01,3,3)	64.3	63.9	0.0	0.0	0.0	0.0	0.0	0.0	**0.5**	**0.6**	**0.1**	**0.5**	**0.6**	**0.1**
German	L(1)	79.0	70.8	**87.5**	**94.5**	**81.5**	**87.5**	**94.5**	**81.5**	**87.5**	**94.5**	**81.5**	**87.5**	**94.5**	**81.5**
	R(10,0.05)	79.5	74.1	0.0	0.0	0.0	0.0	0.0	0.0	**2.0**	0.0	0.0	**2.0**	**82.0**	0.0
	P(0.01,6,6)	75.5	71.8	0.0	0.0	0.0	0.0	0.0	0.0	**78.5**	**76.0**	**10.0**	**78.5**	**76.0**	**10.0**

Data-Availability Statement. Implementation, datasets, and scripts for reproducing the experimental results, are available on GitHub [33] and Zenodo with the identifier https://doi.org/10.5281/zenodo.10053395.

Individual Fairness. Table 2 shows accuracy, balanced accuracy, and verified individual fairness percentages $\mathrm{fair}_{T,S}(\mathrm{SVM})$ on the test sets T and similarity relations S for linear SVMs (denoted in the table as L(regularization parameter C)), nonlinear SVMs with RBF kernels (denoted as R(regularization parameter C, γ)) and polynomial kernels (denoted as P(regularization parameter C, degree, basis)) trained on each dataset. Verified individual fairness percentages are computed w.r.t. the NOISE (columns 'N'), CAT (columns 'C'), and NOISE-CAT (columns 'NC') similarity relations. This table compares results obtained using intervals and RAFs, with and without the OH abstraction. In every row corresponding to a SVM instance, we present the highest verified individual fairness percentages for each similarity relation in bold font and the lowest percentages in a faded shade. It turns out that the RAF abstraction typically outperforms intervals, and adding our OH abstraction always yields equal or higher (even much higher) verified individual fairness. For the same SVMs, in the following table:

Dataset	**Linear**		**Polynomial**		**RBF**	
	LB	UB	LB	UB	LB	UB
Adult	91.6	91.6	0.03	89.5	92.2	95.4
Compas	94.9	94.9	0.09	71.4	89.3	93.0
German	81.5	81.5	10.0	76.0	0.0	84.0

we provide lower and upper bounds on individual fairness w.r.t. the NOISE-CAT similarity relation using the RAF+OH abstraction. The lower bound (columns 'LB') is the verified individual fairness percentage as given in Table 2, while the upper bound (columns 'UB') is an estimate obtained through the counterexample search of Definition 3.12 without input partitioning, e.g., an upper bound of 76% for German/Polynomial means that we found concrete counterexamples to individual fairness for 48 over 200 test samples, thus entailing that at most 152 out of 200 test samples (i.e., 76%) are individually fair. The

Table 3: Comparison of AFI and PFI on German.

		Grade for each feature										
Linear	Baseline (13.55s)	**5**	**5**	**5**	**6**	**6**	**7**	**7**	**7**	**7**	**8**	Distance
	AFI (0.01s)	**5**	**5**	**5**	**6**	**6**	**7**	8	**7**	**7**	**8**	**1.0**
	PFI (4.07s)	**5**	**5**	6	7	7	9	6	6	**7**	7	3.16
RBF	Baseline (17.98s)	**5**	**5**	**5**	**6**	**6**	**7**	**7**	**7**	**8**	**8**	Distance
	AFI (0.02s)	**5**	6	**5**	**6**	**6**	8	**7**	**7**	**8**	7	**1.73**
	PFI (6.23s)	6	7	**5**	**6**	7	8	**7**	6	7	5	4.24
Polynomial	Baseline (15.83s)	**5**	**5**	**5**	**6**	**7**	**7**	**7**	**7**	**7**	**8**	Distance
	AFI (0.01s)	7	6	7	7	5	**7**	6	6	5	**8**	**4.47**
	PFI (4.15s)	6	7	9	7	6	**7**	5	6	6	6	5.74

Table 4: Distances of AFI and PFI from several baselines for different SVMs.

	Baseline	$N = 2k$ $\epsilon = 0.2$	$N = 10k$ $\epsilon = 0.2$	$N = 2k$ $\epsilon = 0.4$	$N = 10k$ $\epsilon = 0.4$	$N = 2k$ $\epsilon = 0.6$	$N = 5k$ $\epsilon = 0.6$	$N = 10k$ $\epsilon = 0.6$	$N = 2k$ $\epsilon = 0.8$	$N = 5k$ $\epsilon = 0.8$	$N = 10k$ $\epsilon = 0.8$
Adult	AFI (0.27s)	**0.0**	**0.0**	**1.0**	**0.0**	**1.0**	**1.41**	**1.0**	**1.0**	**1.41**	**1.0**
Linear	PFI (10009s)	2.45	2.45	2.24	2.45	2.24	**1.41**	2.24	2.24	**1.41**	2.24
Adult	AFI (0.48s)	**1.0**	**1.41**	**1.41**	**1.41**	**1.73**	**1.73**	**1.41**	**1.41**	**1.41**	**1.41**
RBF	PFI (25221s)	1.73	2.45	2.45	2.0	2.65	2.65	2.45	2.45	2.45	2.45
Adult	AFI (0.44s)	**1.0**	**1.0**	**0.0**	1.41	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Polynomial	PFI (9985s)	**1.0**	**1.0**	1.41	**1.0**	1.41	1.41	1.41	1.41	1.41	1.41
Compas	AFI (0.22s)	**1.41**	**1.41**	**1.73**	**1.73**	**1.41**	**1.73**	**1.41**	**1.41**	**1.41**	**1.73**
Linear	PFI (1953s)	1.73	1.73	2.0	2.0	2.24	2.0	2.24	2.24	2.24	2.83
Compas	AFI (0.27s)	**2.0**	**2.0**	**2.65**	**2.65**	**2.83**	**2.83**	**2.83**	**2.83**	**2.83**	**2.83**
RBF	PFI (6827s)	**2.0**	**2.0**	**2.65**	**2.65**	**2.83**	**2.83**	**2.83**	**2.83**	**2.83**	**2.83**
Compas	AFI (0.22s)	4.24	4.24	4.12	4.12	4.24	4.24	4.24	4.24	4.24	4.24
Polynomial	PFI (2069s)	**2.45**	**2.45**	**3.0**	**3.0**	**3.74**	**3.74**	**3.74**	**3.74**	**3.74**	**3.74**
German	AFI (0.01s)	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.41**	**1.73**	**1.41**
Linear	PFI (4.07s)	3.16	3.46	3.16	3.16	3.16	3.16	3.16	3.6	3.74	3.0
German	AFI (0.02s)	**1.73**	**1.0**	**1.73**	**1.73**	**2.0**	**1.41**	**1.73**	**1.73**	**2.0**	**2.24**
RBF	PFI (6.23s)	4.0	3.46	4.24	4.24	4.36	3.61	4.24	4.24	4.36	4.47
German	AFI (0.01s)	**4.90**	**4.12**	**4.47**	**3.87**	**3.87**	**4.24**	**3.46**	**3.46**	**3.46**	**3.46**
Polynomial	PFI (4.15s)	5.74	5.10	5.74	4.69	4.69	5.0	4.58	4.58	4.58	4.58

gap between these bounds is zero for linear SVMs and narrow for RBF kernels trained on the Adult and Compas datasets: in these cases, our RAF+OH abstraction turns out to be (very) precise and the counterexample search heuristics is strong. On the other hand, the gap is much wider in the remaining cases, notably for SVMs with polynomial kernels, mostly due to a lower precision of the abstraction. Using partitioning (i.e., step (S_4) of Definition 3.12) up to 3.125% of the original perturbation size, we get similar upper bounds, thus hinting the presence of a few additional counterexamples. Only partitioning up to 0.1% of the original input size, we could find substantially more counterexamples.

Global Feature Importance. We compare our abstract feature importance AFI, used as a *global* feature importance measure, with the popular global measure PFI, as implemented in the Python *sklearn.inspection* package with *n_repeat* $= 10$. For the sake of comparison with an outside baseline, we uniformly sampled N points in the input space of the SVMs and determined how often a NOISE perturbation for a single numerical input feature changed the SVM classification: the more often the classification changed, the more

Table 5: Local Comparison of AFI and LIME.

Distance between LIME and ...	**Adult**			**Compas**			**German**		
	Lin.	RBF	Poly	Lin.	RBF	Poly	Lin.	RBF	Poly
AFI ($\epsilon = 0.1$)	2.42	2.04	2.98	1.67	1.06	3.05	2.62	2.03	**5.31**
AFI ($\epsilon = 0.2$)	1.68	1.32	2.67	1.63	0.17	2.73	2.21	2.00	5.41
AFI ($\epsilon = 0.3$)	1.39	0.51	2.58	**1.57**	0.14	**2.62**	1.92	2.05	5.45
AFI (Global)	**1.37**	**0.01**	**1.01**	**1.57**	**0.13**	3.16	**1.90**	**1.89**	5.53

Table 6: Time Comparison (in sec) of AFI, PFI, LIME.

Dataset	**Linear**			**Polynomial**			**RBF**		
	AFI	PFI	LIME	AFI	PFI	LIME	AFI	PFI	LIME
Adult	$\mathbf{0.27}$	$1{\cdot}10^4$	3.78	$\mathbf{0.45}$	$1{\cdot}10^4$	6.21	$\mathbf{0.48}$	$2{\cdot}10^4$	9.82
Compas	$\mathbf{0.22}$	$2{\cdot}10^3$	2.72	$\mathbf{0.22}$	$2{\cdot}10^3$	2.89	$\mathbf{0.27}$	$6{\cdot}10^3$	8.97
German	$\mathbf{0.01}$	4.07	0.198	$\mathbf{0.01}$	4.15	0.355	$\mathbf{0.02}$	6.23	0.223

important is the input feature. As a representative example, we show in Table 3, a comparison for the SVMs trained on the German dataset. In lines 'Baseline', 'AFI' and 'PFI', we show the feature grades, as defined in Section 3.1, of the 10 non-categorical (7 numerical plus 3 binary) input features of German based on the importance scores measured by, respectively, baseline, AFI and PFI. The baseline has been computed by considering $N = 10000$ samples and a NOISE perturbation with magnitude $\epsilon = 0.4$. We also indicate in parenthesis the time needed (in seconds) to compute these scores, where for our AFI measure, we used the RAF+OH abstraction. In column 'Distance' we show the Euclidean distance between the feature grades computed by AFI and PFI w.r.t. the baseline. We can observe that AFI better correlates with model variance to feature perturbations than PFI. In fact, the correlation is almost perfect for the linear SVM. For nonlinear SVMs, the abstraction RAF+OH loses more precision, so that the correlation decreases, nevertheless the distance to the baseline is still smaller than for PFI. Note that AFI is computed in a negligible fraction of time w.r.t. PFI.
Table 4 compares the Euclidean distance between the feature grades computed by AFI and PFI w.r.t. different choices of the number of samples N and magnitudes ϵ used for computing the baseline of SVMs trained on the Adult, Compas, and German datasets. For each AFI-baseline and PFI-baseline pair, the smaller distance is made bold and the larger has a faded shade. The data indicates AFI is closer to the baseline than PFI in most cases, except for the polynomial SVM trained on Compas: for this case, the likely reason is the low precision of our polynomial SVM abstraction, as also hinted by the low verified individual fairness scores in the entries for Compas/Polynomial/RAF+OH in Table 2.

Local Feature Importance. In Table 5, we present a comparative analysis between our measure AFI, used as a *local* feature importance measure, and the extensively used local feature importance measure LIME as implemented in the Python *lime.lime_tabular* package [39], for SVMs trained on the three different datasets. The local neighborhood

used by LIME to determine the importance of features is obtained using a normal distribution, and the influence of a point in the neighborhood is dependent on its distance from the original point [38, Section 3.3]. Instead, AFI considers a local uniform distribution for the local neighborhood, i.e., a hyperrectangle abstraction where each point is equiprobable. Thus, it is not possible to find a common local neighborhood in which we can compare local AFI and LIME to a ground truth baseline. Instead, the comparison in Table 5 is based on computing the average Euclidean distance between the feature grades obtained with these two metrics for 200 inputs sampled from the test sets. For our local AFI measure, we took as input to the SVMs a local neighborhood determined by a NOISE perturbation with three different values of the magnitude ϵ. We also compared the distance of LIME with the global AFI computed over the entire input space of the SVMs. The results show that these distances are consistently minimal (except for the polynomial SVMs), implying a high degree of similarity between the two feature importance measures. Interestingly, this table shows that the local LIME feature grades are closer to the global AFI feature grades than to the local AFI feature grades. Furthermore, as the value of the magnitude ϵ used to compute local AFI increases, the distance between LIME and local AFI feature grades decreases.

Computation Time. Finally, Table 6 presents a comparison of the computation times (in seconds) of AFI, PFI, and LIME for our SVMs. As time taken by global and local AFI is similar, we only report global AFI time. The results shows that AFI always outperforms both PFI and LIME.

5 Conclusion

We put forward a novel feature importance measure based on an abstract interpretation of SVMs, which is tailored for achieving a precise symbolic representation of one-hot encoded features. We showed that our abstraction is effective for certifying robustness properties—notably, individual fairness—of SVMs, and that our abstract feature importance measure outperforms the state-of-the-art. As future work, we plan to extend our approach to certify alternative or stronger fairness notions. We also aim to design quantitative verification methods to provide probabilistic guarantees on the behavior of SVM models.

Acknowledgements. Francesco Ranzato and Marco Zanella were partially funded by the *Italian MIUR*, under the PRIN 2017 project no. 201784YSZ5. Francesco Ranzato was partially funded by: the *Italian MUR*, under the PRIN 2022 PNRR project no. P2022HXNSC; *Meta* (formerly *Facebook*) *Research*, under a "Probability and Programming Research Award" and under a *WhatsApp Research Award* on "Privacy-aware Program Analysis"; by an *Amazon Research Award* for "AWS Automated Reasoning". Caterina Urban was partially funded by the French PEPR Intelligence Artificielle SAIF project (ANR-23-PEIA-0006).

References

1. Aho, A.V., Lam, M.S., Sethi, R., Ullman, J.D.: Compilers: Principles, Techniques, and Tools (2nd Edition). Addison-Wesley Longman Publishing Co., Inc., USA (2006)
2. Albarghouthi, A.: Introduction to neural network verification. Found. Trends Program. Lang. **7**(1-2), 1–157 (2021). https://doi.org/10.1561/2500000051
3. Angwin, J., Larson, J., Mattu, S., Kirchner, L.: Machine Bias. ProPublica **23** (2016), https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing
4. Apley, D.W., Zhu, J.: Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models. Journal of the Royal Statistical Society Series B: Statistical Methodology **82**(4), 1059–1086 (2020), https://doi.org/10.1111/rssb.12377
5. Bhatt, U., Xiang, A., Sharma, S., Weller, A., Taly, A., Jia, Y., Ghosh, J., Puri, R., Moura, J.M.F., Eckersley, P.: Explainable machine learning in deployment. In: Proc. 2020 Conference on Fairness, Accountability, and Transparency, FAT* '20. pp. 648–657. ACM (2020). https://doi.org/10.1145/3351095.3375624
6. Breiman, L.: Random Forests. Machine Learning **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
7. Carlini, N., Wagner, D.A.: Towards Evaluating the Robustness of Neural Networks. In: Proc. of 38th IEEE Symposium on Security and Privacy (S & P 2017). pp. 39–57 (2017). https://doi.org/10.1109/SP.2017.49
8. Casalicchio, G., Molnar, C., Bischl, B.: Visualizing the feature importance for black box models. In: Machine Learning and Knowledge Discovery in Databases - Proceedings of the European Conference, ECML PKDD 2018. Lecture Notes in Computer Science, vol. 11051, pp. 655–670. Springer (2018). https://doi.org/10.1007/978-3-030-10925-7_40
9. Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., Lopez, A.: A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing **408**, 189–215 (2020), https://doi.org/10.1016/j.neucom.2019.10.118
10. Chang, Y.W., Lin, C.J.: Feature ranking using linear SVM. In: Proceedings of the Workshop on the Causation and Prediction Challenge at WCCI 2008. Proceedings of Machine Learning Research, vol. 3, pp. 53–64. PMLR (2008), http://proceedings.mlr.press/v3/chang08a.html
11. Chouldechova, A.: Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments. Big Data **5**(2), 153–163 (2017). https://doi.org/10.1089/big.2016.0047
12. Cousot, P.: Principles of Abstract Interpretation. MIT Press (2021)
13. Cousot, P., Cousot, R.: Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Proc. 4th ACM Symposium on Principles of Programming Languages (POPL 1977). pp. 238–252 (1977). https://doi.org/10.1145/512950.512973
14. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press (2000). https://doi.org/10.1017/CBO9780511801389
15. Dua, D., Graff, C.: UCI Machine Learning repository (2017), https://archive.ics.uci.edu/ml
16. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.S.: Fairness through awareness. In: Innovations in Theoretical Computer Science 2012. pp. 214–226. ACM (2012), https://doi.org/10.1145/2090236.2090255
17. Fish, B., Kun, J., Lelkes, Á.D.: A confidence-based approach for balancing fairness and accuracy. In: Proceedings of the 2016 SIAM international conference on data mining. pp. 144–152. SIAM (2016), https://doi.org/10.1137/1.9781611974348.17
18. Fisher, A., Rudin, C., Dominici, F.: All Models are Wrong, but Many are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously. Journal of Machine Learning Research **20**(177), 1–81 (2019), http://jmlr.org/papers/v20/18-760.html

19. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. The Annals of Statistics **29**(5), 1189–1232 (2001), http://www.jstor.org/stable/2699986
20. Ghorbal, K., Goubault, E., Putot, S.: The zonotope abstract domain Taylor1+. In: Computer Aided Verification, 21st International Conference, CAV 2009. Proceedings. Lecture Notes in Computer Science, vol. 5643, pp. 627–633. Springer (2009), https://doi.org/10.1007/978-3-642-02658-4_47
21. Ghosh, B., Basu, D., Meel, K.S.: Algorithmic fairness verification with graphical models. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022. pp. 9539–9548 (2022), https://doi.org/10.1609/aaai.v36i9.21187
22. Goldstein, A., Kapelner, A., Bleich, J., Pitkin, E.: Peeking Inside the Black Box: Visualizing Statistical Learning With Plots of Individual Conditional Expectation. Journal of Computational and Graphical Statistics **24**(1), 44–65 (2015), https://doi.org/10.1080/10618600.2014.907095
23. Goodfellow, I., McDaniel, P., Papernot, N.: Making Machine Learning Robust Against Adversarial Inputs. Commun. ACM **61**(7), 56–66 (2018). https://doi.org/10.1145/3134599
24. Hechtlinger, Y.: Interpretation of Prediction Models Using the Input Gradient. CoRR arXiv (2016), http://arxiv.org/abs/1611.07634
25. Hooker, G., Mentch, L., Zhou, S.: Unrestricted permutation forces extrapolation: Variable importance requires at least one more model, or there is no free variable importance. Statistics and Computing **31**(6) (2021), https://doi.org/10.1007/s11222-021-10057-z
26. Khandani, A.E., Kim, A.J., Lo, A.W.: Consumer Credit-Risk Models via Machine-Learning Algorithms. Journal of Banking & Finance **34**(11), 2767–2787 (2010). https://doi.org/https://doi.org/10.1016/j.jbankfin.2010.06.001
27. Langenberg, P., Balda, E.R., Behboodi, A., Mathar, R.: On the robustness of support vector machines against adversarial examples. In: 13th International Conference on Signal Processing and Communication Systems, ICSPCS 2019. pp. 1–6. IEEE (2019). https://doi.org/10.1109/ICSPCS47537.2019.9008746
28. Liu, C., Arnon, T., Lazarus, C., Strong, C.A., Barrett, C.W., Kochenderfer, M.J.: Algorithms for verifying deep neural networks. Found. Trends Optim. **4**(3-4), 244–404 (2021), https://doi.org/10.1561/2400000035
29. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017. pp. 4765–4774 (2017), https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html
30. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A.: A survey on bias and fairness in machine learning. ACM Comput. Surv. **54**(6) (Jul 2021). https://doi.org/10.1145/3457607
31. Messine, F.: Extentions of affine arithmetic: Application to unconstrained global optimization. J. Universal Computer Science **8**(11), 992–1015 (2002). https://doi.org/10.3217/jucs-008-11-0992
32. Mladenic, D., Brank, J., Grobelnik, M., Milic-Frayling, N.: Feature selection using linear classifier weights: interaction with classification models. In: SIGIR 2004: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 234–241. ACM (2004). https://doi.org/10.1145/1008992.1009034
33. Pal, A., Ranzato, F., Urban, C., Zanella, M.: Abstract Feature Importance for SVMs. https://github.com/AFI-SVM (2023)
34. Park, S., Byun, J., Lee, J.: Privacy-preserving fair learning of support vector machine with homomorphic encryption. In: WWW '22: The ACM Web Conference 2022. pp. 3572–3583. ACM (2022). https://doi.org/10.1145/3485447.3512252

35. Ranzato, F., Urban, C., Zanella, M.: Fairness-aware training of decision trees by abstract interpretation. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, CIKM2021. pp. 1508–1517 (2021), https://doi.org/10.1145/3459637.3482342
36. Ranzato, F., Zanella, M.: Robustness Verification of Support Vector Machines. In: Proc. 26th International Static Analysis Symposium (SAS 2019). pp. 271–295. LNCS vol. 11822 (2019). https://doi.org/10.1007/978-3-030-32304-2_14
37. Ranzato, F., Zanella, M.: Saver: SVM Abstract Verifier (2019), https://github.com/abstract-machine-learning/saver
38. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. pp. 1135–1144. ACM (2016). https://doi.org/10.1145/2939672.2939778
39. Ribeiro, M.T.C.: Local Interpretable Model-agnostic Explanations (LIME) (2016), https://lime-ml.readthedocs.io
40. Roh, Y., Lee, K., Whang, S., Suh, C.: Fr-train: A mutual information-based approach to fair and robust training. In: Proc. of the 37th Int. Conf. on Machine Learning (ICML 2020). Proceedings of Machine Learning Research, vol. 119, pp. 8147–8157. PMLR (2020), http://proceedings.mlr.press/v119/roh20a.html
41. Ruoss, A., Balunovic, M., Fischer, M., Vechev, M.T.: Learning certified individually fair representations. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS 2020) (2020), https://proceedings.neurips.cc/paper/2020/hash/55d491cf951b1b920900684d71419282-Abstract.html
42. Shapley, L.S.: A Value for n-Person Games. In: Kuhn, H.W., Tucker, A.W. (eds.) Contributions to the Theory of Games II, pp. 307–317. Princeton University Press, Princeton (1953)
43. Tjoa, E., Guan, C.: A survey on explainable artificial intelligence (XAI): Toward medical XAI. IEEE Trans. Neural Networks Learn. Syst. **32**(11), 4793–4813 (2021), https://doi.org/10.1109/TNNLS.2020.3027314
44. Urban, C., Christakis, M., Wüstholz, V., Zhang, F.: Perfectly parallel fairness certification of neural networks. Proc. ACM Program. Lang. **4**(OOPSLA), 185:1–185:30 (2020). https://doi.org/10.1145/3428253
45. Urban, C., Miné, A.: A Review of Formal Methods applied to Machine Learning. CoRR arXiv (2021), https://arxiv.org/abs/2104.02466
46. Verma, S., Rubin, J.: Fairness definitions explained. In: Proceedings of the International Workshop on Software Fairness, FairWare@ICSE 2018. pp. 1–7. ACM (2018). https://doi.org/10.1145/3194770.3194776
47. Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., Roli, F.: Support vector machines under adversarial label contamination. Neurocomputing **160**, 53–62 (2015). https://doi.org/10.1016/j.neucom.2014.08.081
48. Yurochkin, M., Bower, A., Sun, Y.: Training individually fair ML models with sensitive subspace robustness. In: Proc. 8th International Conference on Learning Representations, ICLR 2020 (2020), https://openreview.net/forum?id=B1gdkxHFDH