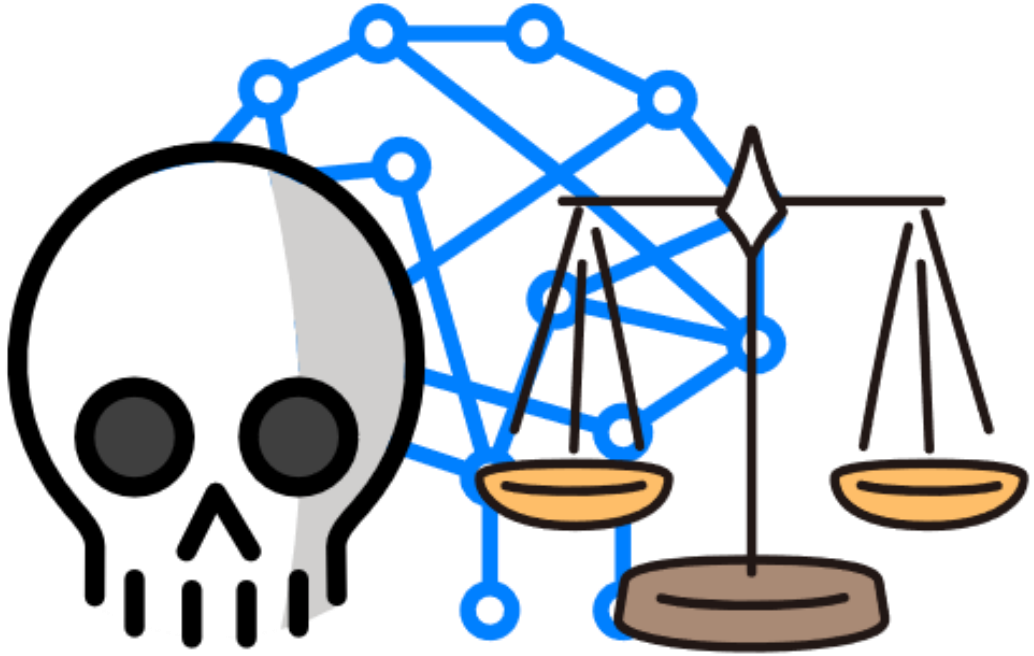
Università Ca'Foscari Venezia

Data-Independent Verification of Robustness and Fairness of Tree-Based Classifiers

Lorenzo Cazzaro (Università Ca' Foscari Venezia)

Sapienza-Università di Roma, 09/11/2023



$ whoami

I am a Ph.D. student in Computer Science under the supervision of prof. Stefano Calzavara.

Main research interest: **Security of AI (and viceversa):**

- Design and verification of (security and fairness) properties of Machine Learning (ML) models.
- Evasion Attacks against ML.
- Applying AI for improving the security of web applications.

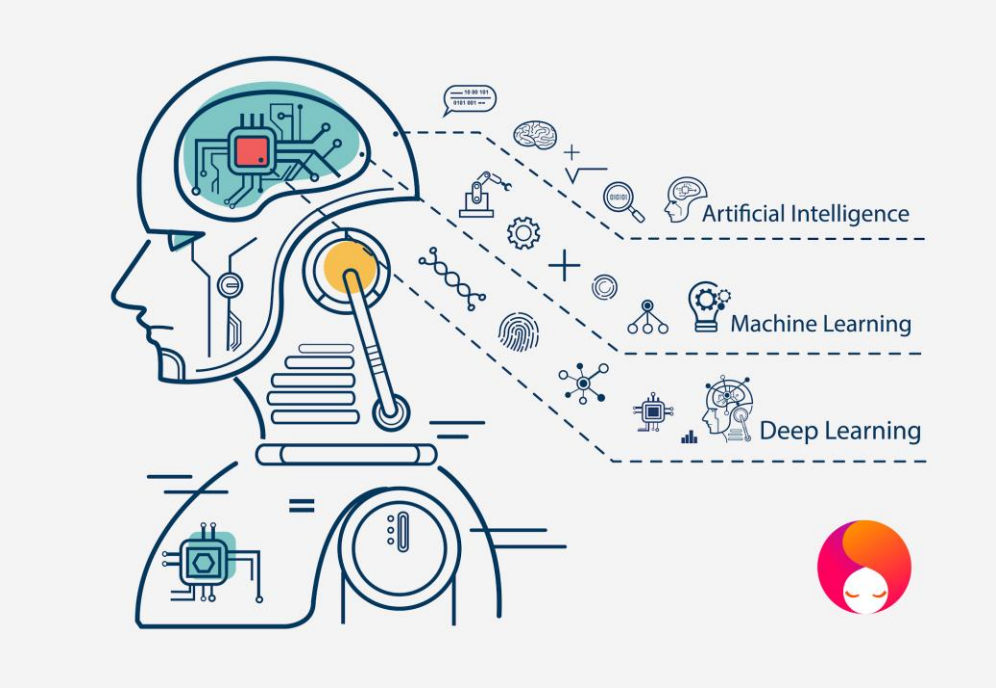
Artificial Intelligence and Machine Learning

We can build a much brighter future where humans are relieved of menial work using AI capabilities.

Andrew Ng., 2018

Applications:

- Image recognition (Google Lens)
- Natural language translation (DeepL)
- Recommender systems
- Malware detection
- …

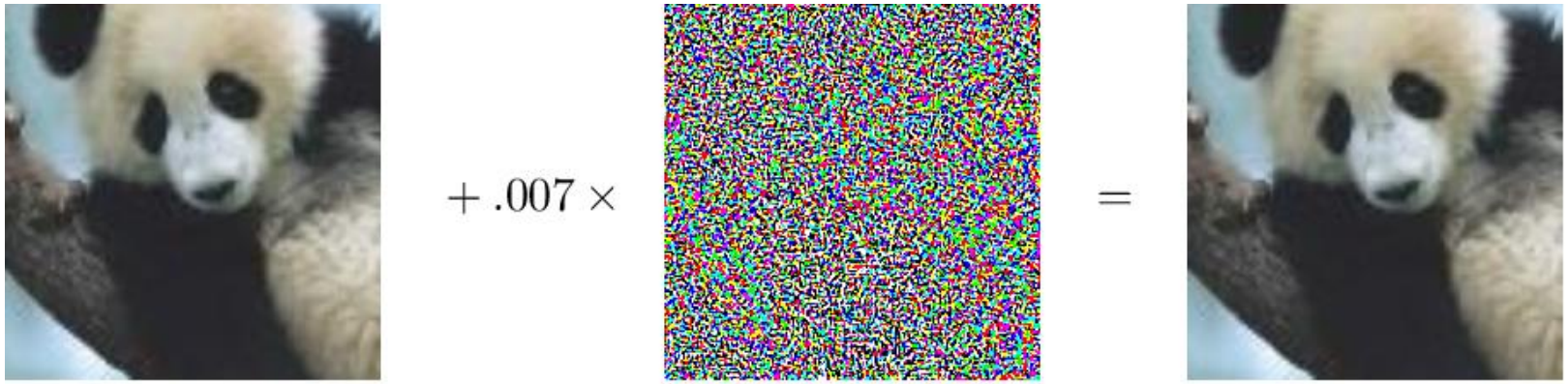


Given its pervasivity, AI (then, ML) must be trustworthy!

We are going to focus on ML in this talk.

Is Machine Learning Secure (Robust)?

Adversarial Examples are a serious threat to ML robustness…

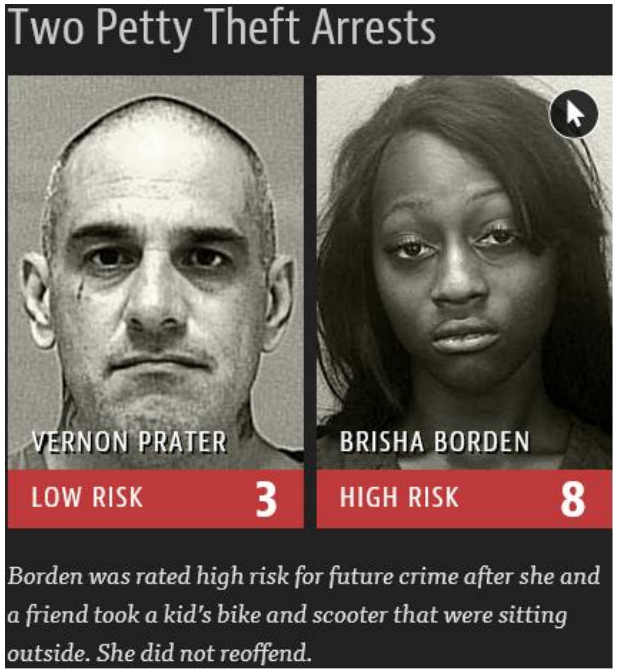


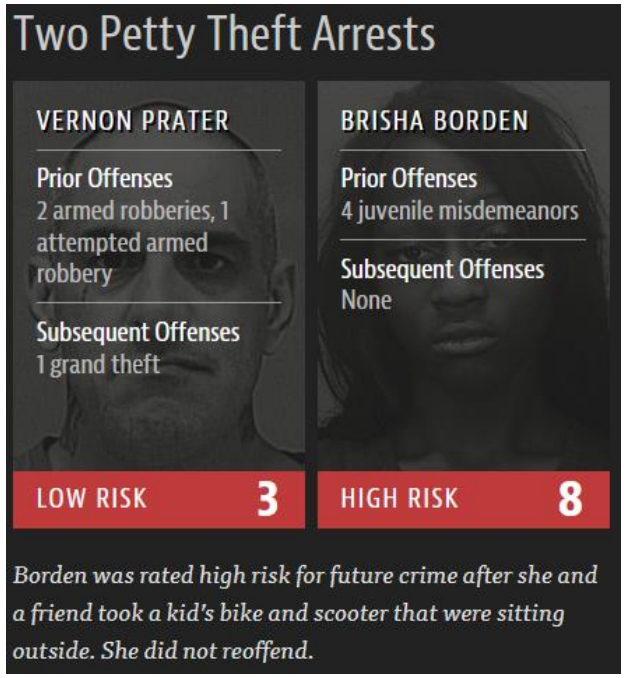
Credits: *Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In ICLR (2015)*

They can be generated also in other domains than computer vision, e.g., malware detection.

Is Machine Learning Fair?

Example: Machine Learning (ML) used to predict recidivity in USA*





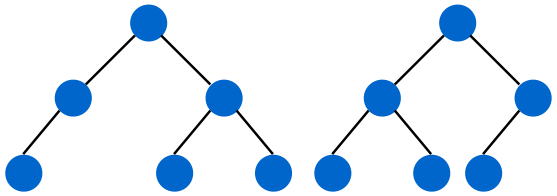
Non-recidivist black people were twice as likely to be labelled high risk than non-recidivist white people.

*https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

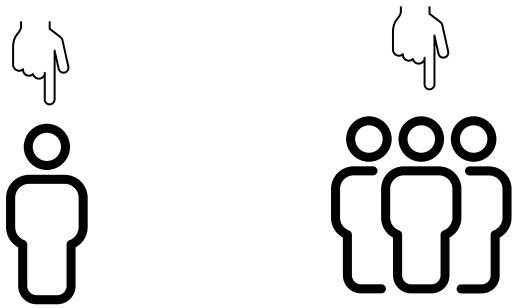
The Need For (Expressive) Properties

We need to describe the trustworthy behaviour of a ML model by defining some **properties.**

Local Properties

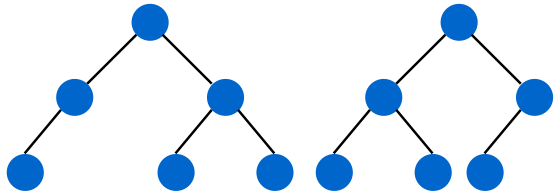


It's robust/fair on



Test set

Global / Data-Independent Properties



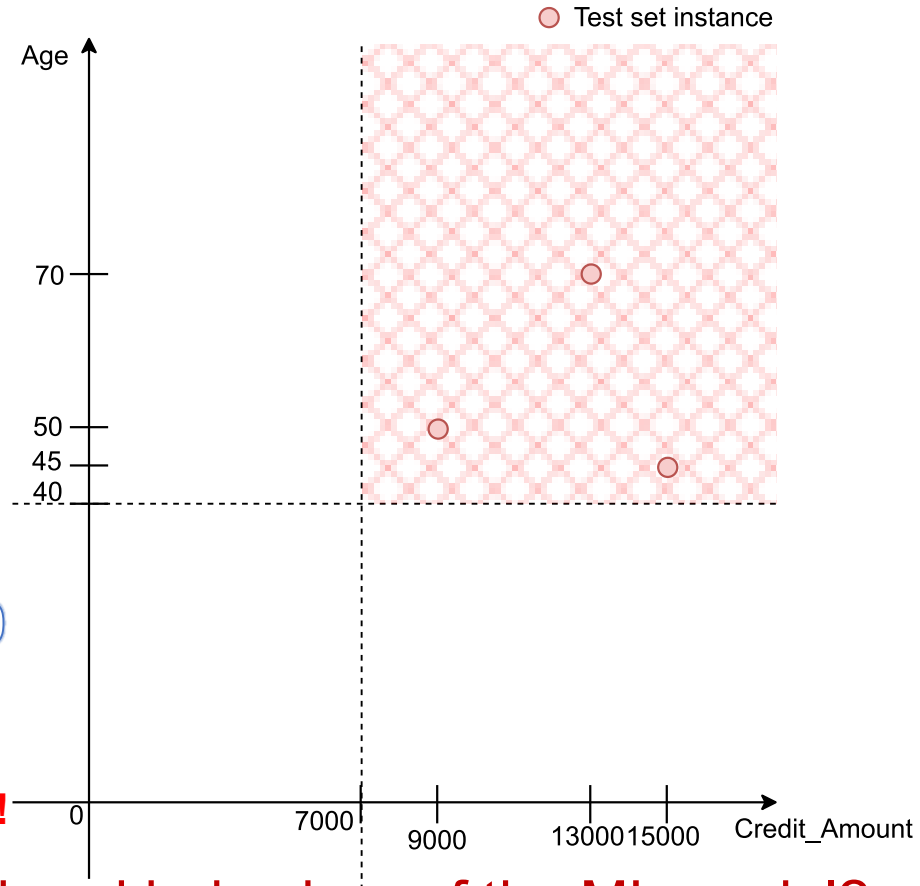
It is robust/fair on people described by

Age >= 40

and

Credit_Amount >=7000

Potentially continuous and unbounded subset of instances!

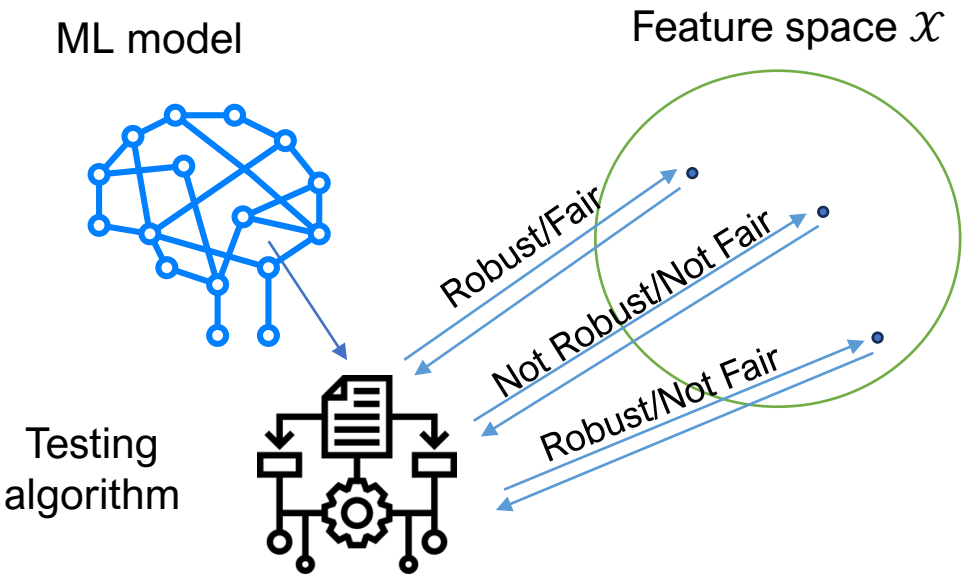


Is the considered property sufficient to describe the desidered behaviour of the ML model?

The Need For (Formal) Verification.

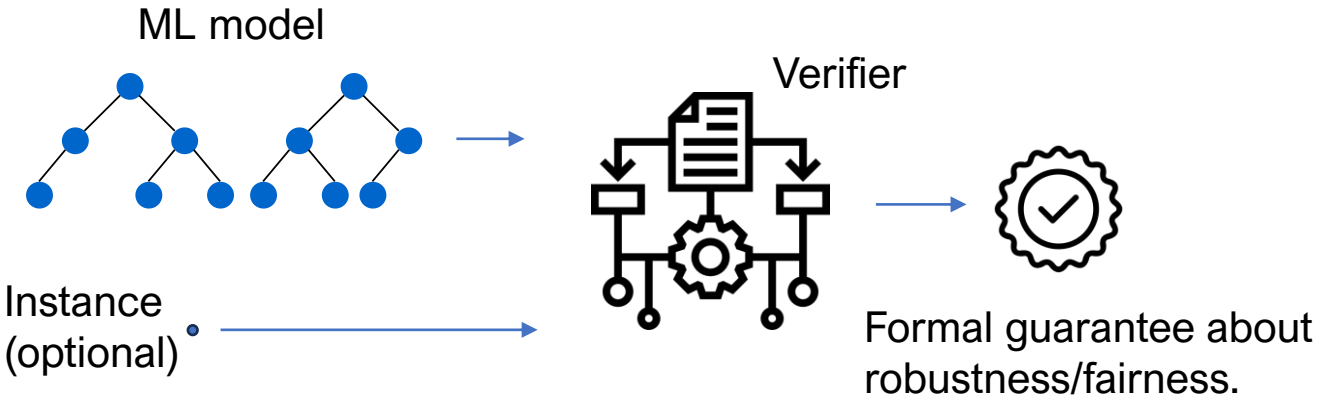
How can we prove the security and/or fairness of ML models?

Empirical Approach / Testing



- **Pros: fast execution.**
- **Question: is it sufficient to prove the property of interest?**

Formal Approach



- **Pros: formal guarantees in output + it can cover more properties than the empirical approach.**
- **Questions: Soundness? Completeness? Scalability? Is the guarantee local or global?**

Talk Outline

We will see:

1. An introduction to tree-based models, the (local) robustness property and a popular fairness property, lack of causal discrimination.

2. The description of the shortcomings of the (local) robustness property for ML models, its data-independent generalization, called *resilience,* and a sound algorithmic way to prove it.

3. How fairness testing approaches fail to verify the lack of causal discrimination and how to verify this property by giving explainable formal guarantees in output .

Thanks to: Stefano Calzavara, Claudio Lucchese, Federico Marcuzzi and Salvatore Orlando.

Technical Background

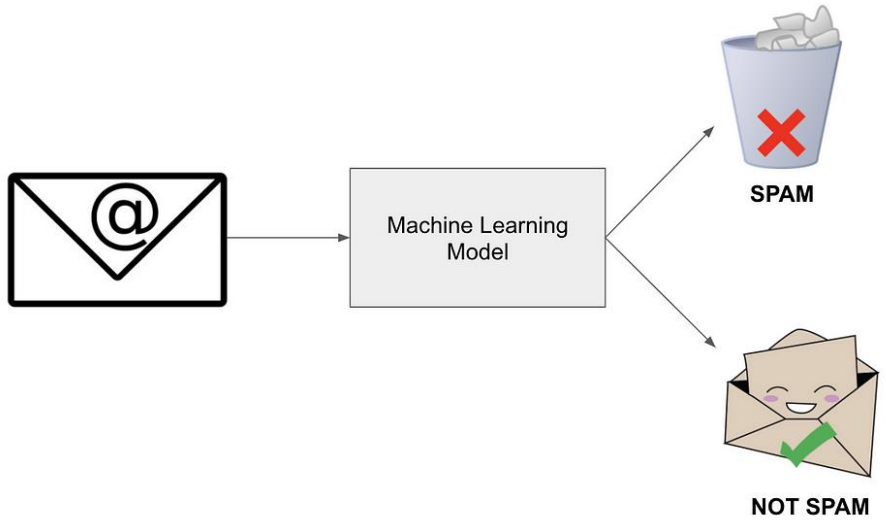
Classifiers

We focus on *ML classifiers*:

- $X \sqsubseteq \mathbb{R}^d$ is the vector space of features.
- $Y = \{-1,1\}$ a finite set of labels.
- $\vec{x} = (x_1, x_2, \ldots, x_d)$ is an instance.

- A classifier is a function $f : X \rightarrow Y$ that assigns a class label to an instance $\vec{x}$. It approximates the unknown target function $g : X \rightarrow Y$.

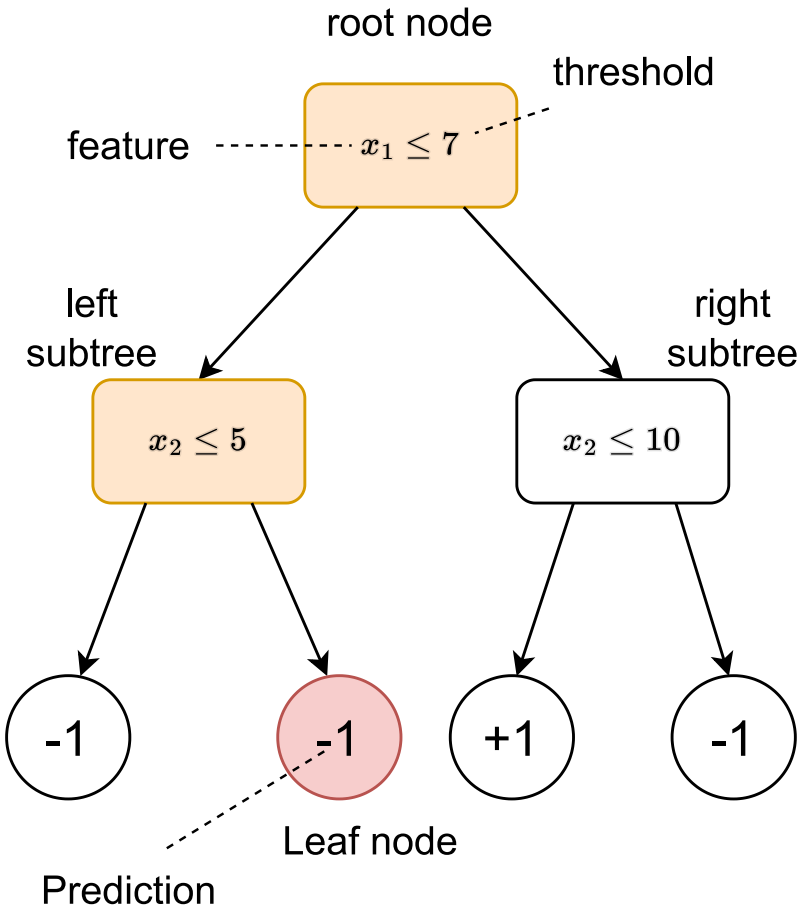
- f is automatically trained by a supervised learning algorithm using a training set $D_{train} = \{(\overrightarrow{x_i}, g(\overrightarrow{x_i}))\}_i$.



Tree-Based Classifiers

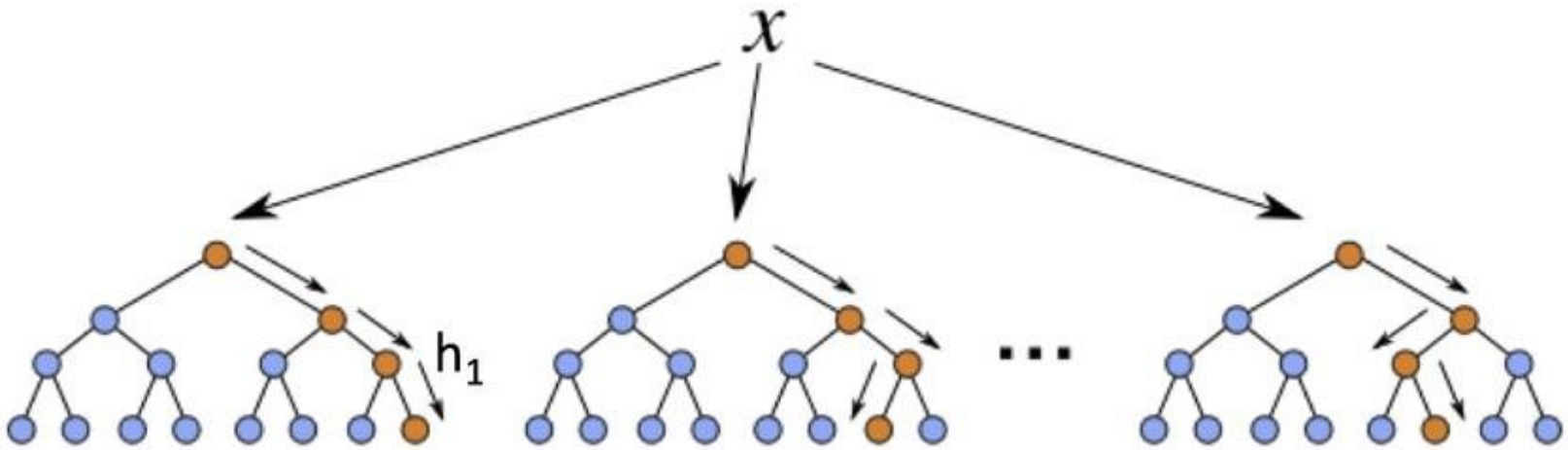
The condition at each node is a logic formula $x_f \leq v$, where:

- x_f is a feature.
- v is a threshold.



Decision Tree Ensembles

Decision Tree Ensembles (Forests) $T = \{t_1, t_2, \dots, t_n\}$ are collections of decision trees.



The ensemble predictions is the combination of the predictions of the single trees: *majority voting* (used today), weight sum, etc…

Data-Independent Verification of Robustness of Tree-Based Classifiers

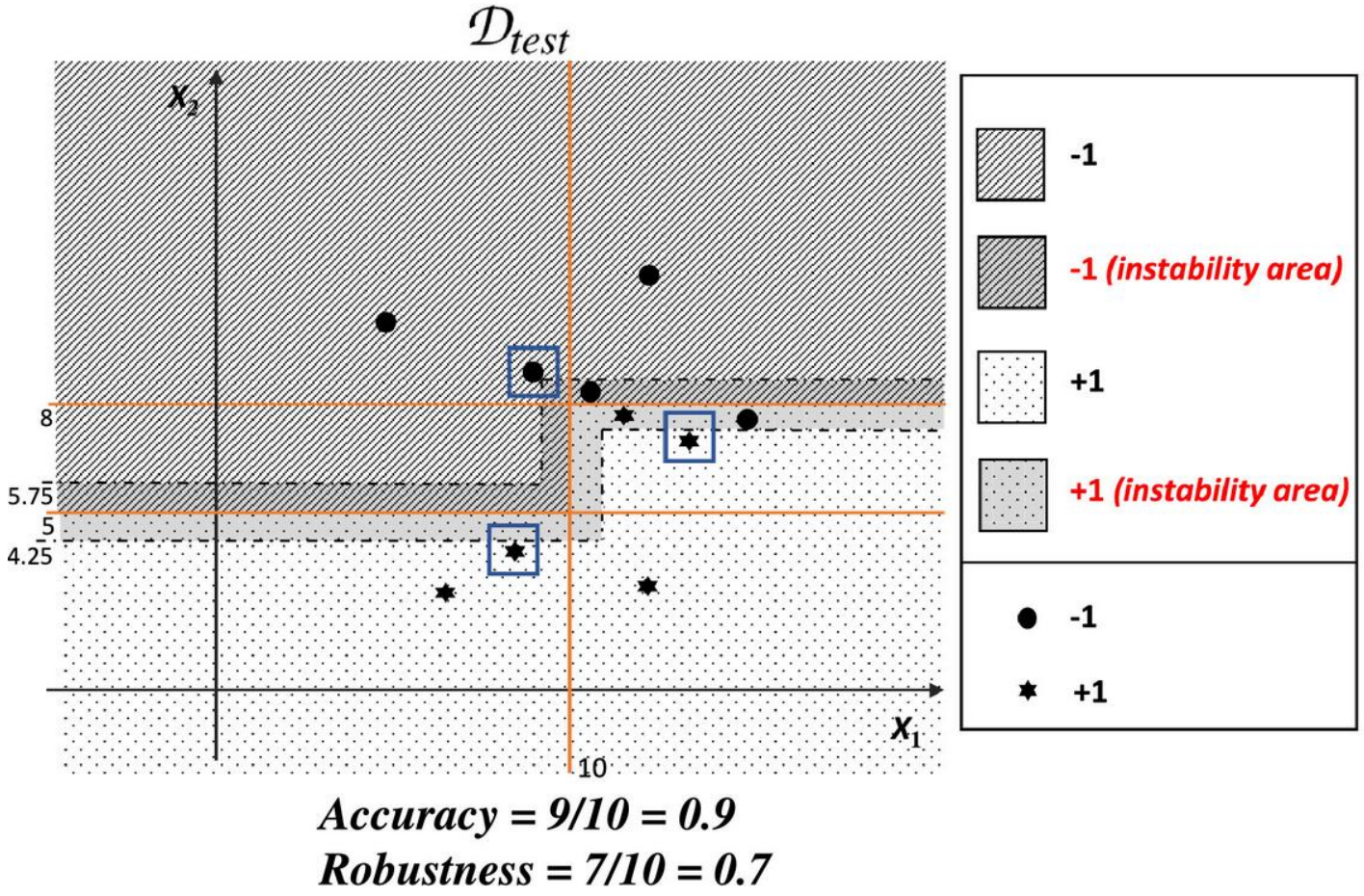
Calzavara S., Cazzaro L., Lucchese C., Marcuzzi F., Orlando S. - *Beyond Robustness: Resilience Verification of Tree-Based Classifiers*, in Computers & Security (2022)

Robustness

The attacker $A(\vec{x}): X \rightarrow P(X)$ maps each input to the adversarial manipulations of the instance $\vec{x}$.

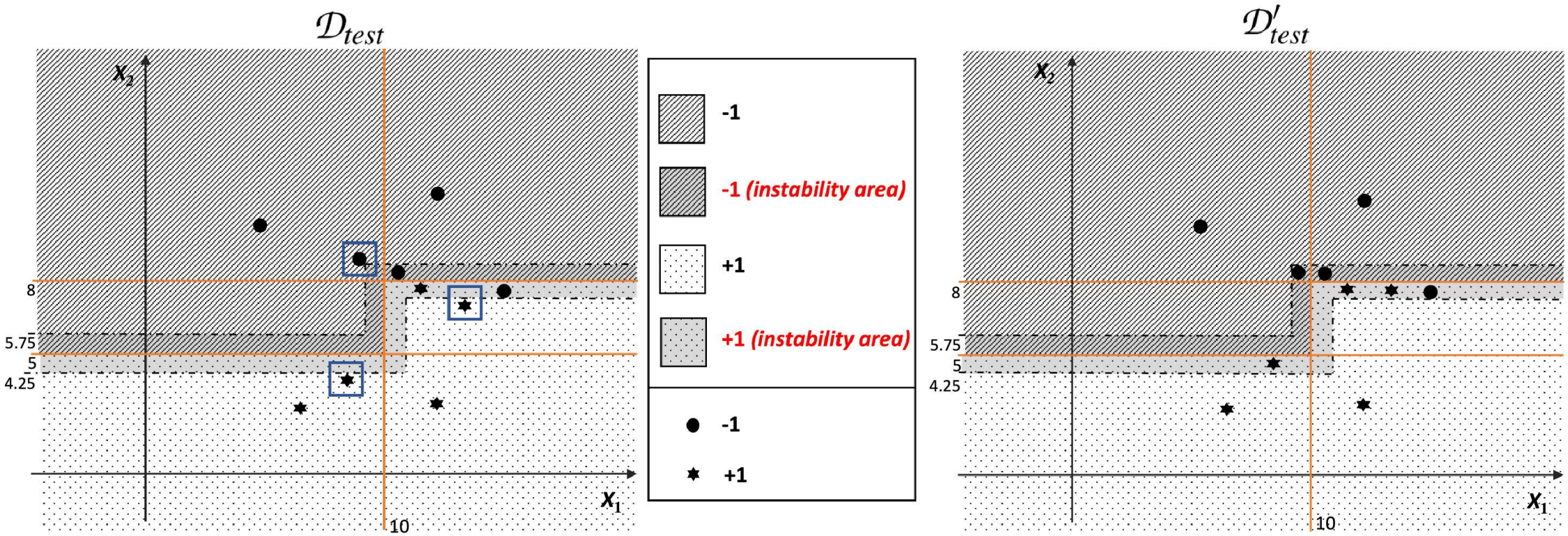
The classifier f is **robust** on the instance $\vec{x}$ with label y if:

1. $f(\vec{x}) = y$.
2. For all $\vec{z} \in A(\vec{x})$ we have $f(\vec{z}) = y$ (**stability** property).



Shortcomings of Robustness

A key problem of robustness is its ***data-dependence***.
Tiny difference between two test sets ⟶ *quite different values* of robustness!



Accuracy = 9/10 = 0.9
Robustness = 7/10 = 0.7

Accuracy = 9/10 = 0.9
Robustness = 4/10 = 0.4

Prior Work: Global Robustness

Leggere paper e fixare per renderla più perso

Prior research proposed **global robustness** properties to mitigate this issue:

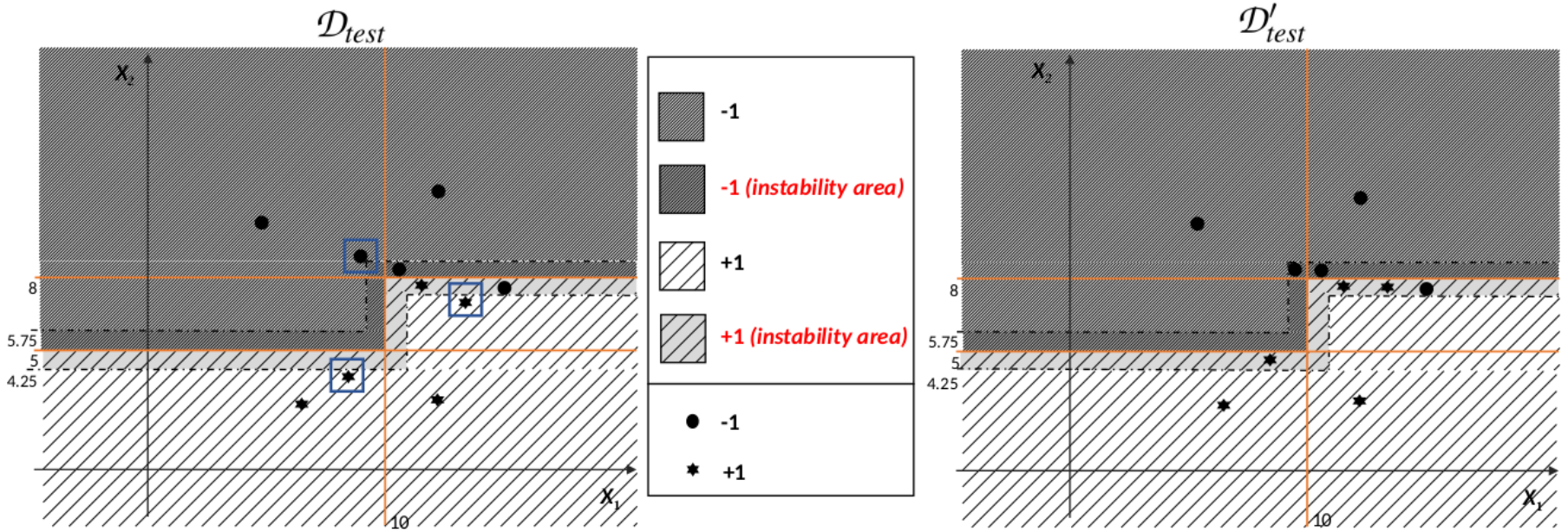
- Leino et al., ICML'21: every input in the instability area must be classified into the special class $\perp$.
  - Pros: great idea.
  - Cons: it only applies to trained classifiers which can output $\perp$.
- Chen et al., CCS'21: any pair of inputs which differ just for the value of the attackable features must be assigned the same class.
  - Pros: Intuitive generalization of robustness.
  - Cons: too strong in practice because this requirement transitively propagates; none of the standard ML models used in their experiments satisfies it.

We look for a security definition which keeps the intuitive flavour of robustness and is applicable to arbitrary classifiers.

Resilience

$N(\vec{x})$ is the set of neighbours of $\vec{x}$, instances that could have been sampled in place of $\vec{x}$ ⟶ it helps to generalize robustness beyond the test-set.

Resilience: a classifier f is **resilient** on the instance $\vec{x}$ if and only if f is robust on $\vec{x}$ and f is stable on all the instances $\vec{z} \in N(\vec{x})$.



Robustness = 7/10 = 0.7

Resilience = 4/10 = 0.4

Robustness = 4/10 = 0.4

Resilience = 4/10 = 0.4

Resilience Verification

A classifier f is **resilient** on the input $\vec{x}$ if and only if:

1. f is robust on $\vec{x} \rightarrow$ existing methods allow to address this problem.
2. f is stable on all the instances $\vec{z} \in N(\vec{x}) \rightarrow$ more challenging, because $N(\vec{x})$ is generally an infinite set of instances.

Solution to point 2: use a Data-Independent Stability Analysis (DISA), which symbolically identifies a subset $S \subseteq X$ where f is proved to be stable:

- Point 2 of resilience requires that $N(\vec{x}) \subseteq S$
- Since S is also infinite in general, we characterize it using a closed form

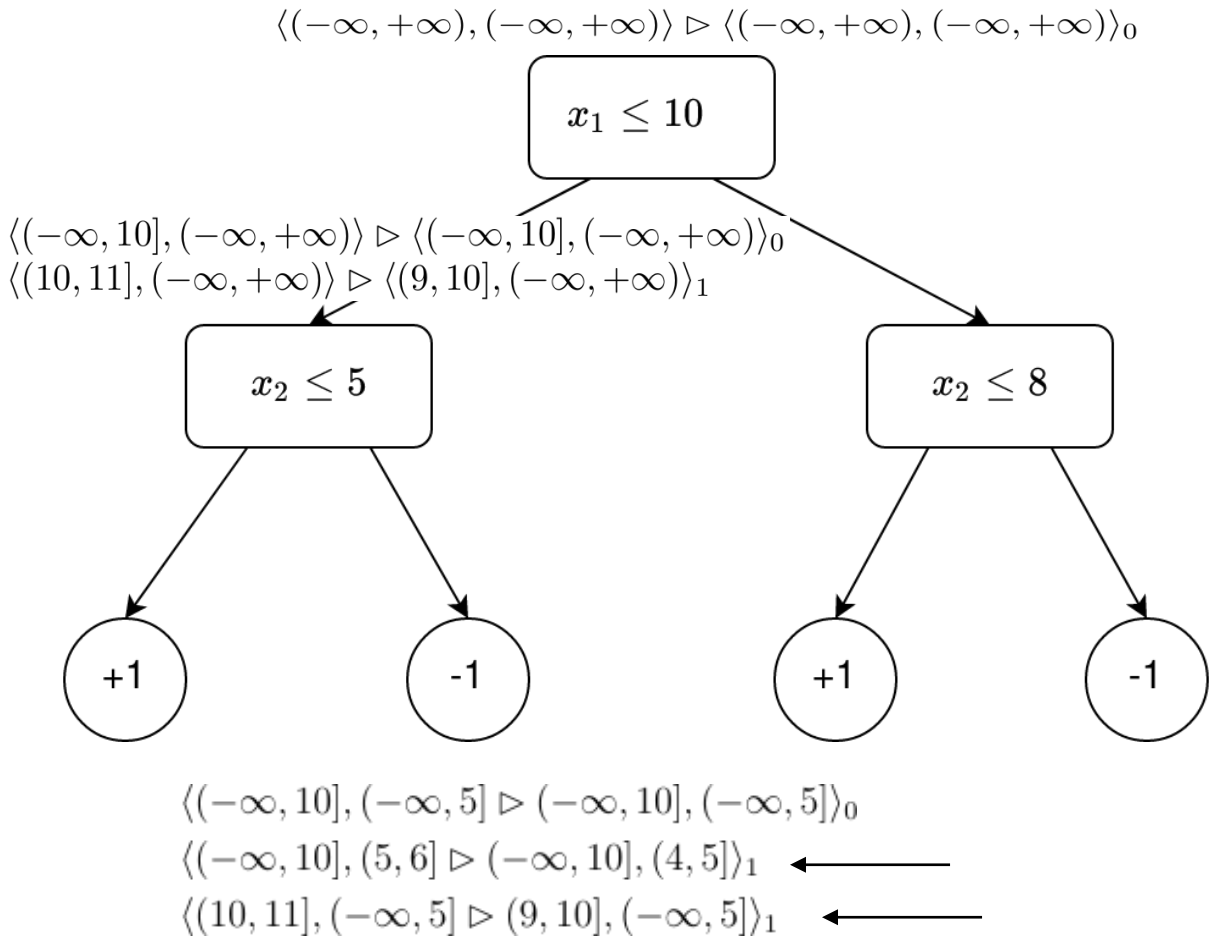
The analysis is **data-independent**: it depends on the classifier, not its inputs!

DISA for Tree Classifiers

Scenario: budget $b = 1$, perturbation in $[-1,1]$.

The subset S where f is stable is easy to identify when f is a decision tree:

- Linear-time tree traversal algorithm, which recursively annotates the nodes with a set of **symbolic attacks** (with pre-image, post-image and budget).
- Afterwards: look for leaves with budget 0 and leaves with budget > 0 with different labels and overlapping pre-images ⟶ their ∩ is part of the instability area.
- Incrementally extend the instability area until all leaves have been processed.



DISA for tree ensembles

The stability area S is harder to identify when g is a tree ensemble:

- We can prove **NP-hardness** by using the fact that robustness verification is NP-hard for decision tree ensembles [Kantchelian et al., ICML 2016]

In the paper, we present an iterative algorithm to approximate S:

- Fixed-point algorithm computing ever-increasing subsets of S.
- Early stopping does not sacrifice soundness, but will break completeness

The analyzer in C++ is available on Github: https://github.com/FedericoMarcuzzi/resilience-verification

Shortcomings of Robustness in Practice

Methodology:

- We use 100 synthetic test sets, where each instance $\vec{x}$ is replaced by one in $N(\vec{x})$.
- We compute the min and max accuracy and robustness over all the synthetic test sets.

Result: **robustness is sensible to small amount of noise**, while the variation of the accuracy is relatively small!

		Standard Models						Robust Models					
Dataset	ε	a	a_{min}	a_{max}	r	r_{min}	r_{max}	a	a_{min}	a_{max}	r	r_{min}	r_{max}
diabetes	0.01	0.714	0.708	0.721	0.649	0.643	0.662	0.727	0.721	0.727	0.714	0.675	0.714
	0.02	0.714	0.708	0.714	0.649	0.630	0.662	0.727	0.714	0.740	0.714	**0.669**	**0.721**
	0.03	0.714	0.688	0.714	0.649	**0.636**	**0.682**	0.727	0.721	0.747	0.714	**0.669**	**0.727**
	0.04	0.714	0.688	0.727	0.649	**0.630**	**0.701**	0.727	0.708	0.747	0.714	**0.675**	**0.734**
cod-rna	0.01	0.775	0.774	0.775	0.686	0.676	0.690	0.750	0.748	0.753	0.714	0.710	0.721
	0.02	0.775	0.773	0.775	0.686	0.665	0.686	0.750	0.749	0.758	0.714	0.711	0.725
	0.03	0.775	0.773	0.775	0.686	0.657	0.686	0.750	0.750	0.760	0.714	0.705	0.723
	0.04	0.775	0.768	0.775	0.686	0.650	0.686	0.750	0.752	0.761	0.714	0.703	0.723

Effectiveness of Resilience Verification

Methodology:

- assess whether our resilience estimate $\hat{R}$ accurately captures robustness for the "most unlucky" neighborhood of the test set, noted $\bar{r}$.

Results:

- $\hat{R}$ **is a rather precise under-approximation** of $\bar{r}$.
- The difference between the real robustness r and the resilience $\hat{R}$ can be significant.

				Standard Models					Robust Models				
Dataset	ε	# Trees	Depth	a	r	$\hat{r}$	$\bar{r}$	$\hat{R}$	a	r	$\hat{r}$	$\bar{r}$	$\hat{R}$
diabetes	0.01	5	3	0.708	0.662	0.643	0.656	0.636	0.727	**0.714**	0.701	**0.675**	**0.662**
		7	3	0.714	0.649	0.630	0.636	0.623	0.727	**0.714**	0.708	**0.675**	**0.662**
		9	3	0.747	0.656	0.630	0.623	0.617	0.753	**0.740**	0.727	**0.695**	**0.688**
cod-rna	0.01	5	3	0.775	**0.686**	0.672	**0.639**	**0.621**	0.752	0.715	0.707	0.698	0.691
		7	3	0.775	**0.686**	0.666	**0.640**	**0.612**	0.750	0.714	0.713	0.698	0.697
		9	3	0.769	**0.677**	0.663	**0.625**	**0.605**	0.750	0.714	0.713	0.698	0.697

Take-away messages

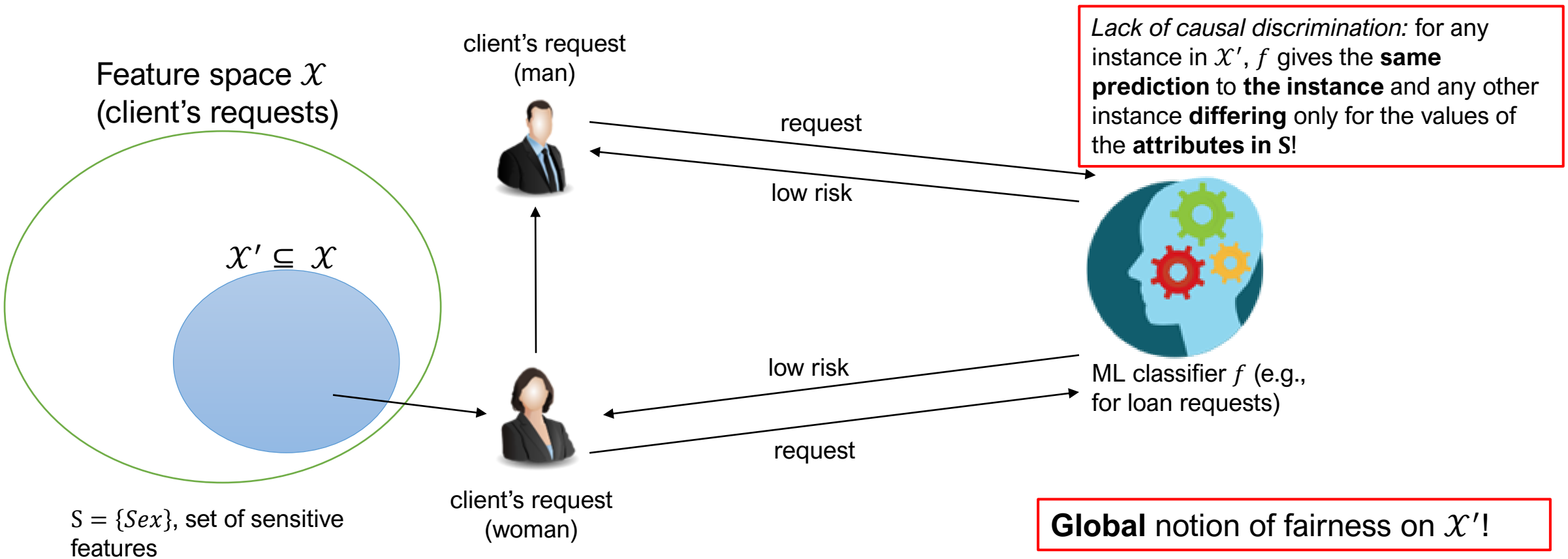
1. Experimental results show that **robustness may give a false sense of security**. More expressive properties (data-independent or global) are needed.
2. **Resilience is useful in practice**, since it gives a lower bound of the robustness computed on the “most unlucky” neighborhood of the test set.
3. Verification tools that analyze the inherent structure of a classifier, without relying on specific instances, are also needed.
4. Resilience can be estimated by extending existing robustness verifiers with a data-independent stability analysis.
5. The DISA may be expensive, but it is sufficient to run it only once! Moreover, it allows to verify a more expressive property!

Data-Independent Fairness Verification of Robustness of Tree-Based Classifiers

Calzavara S., Cazzaro L., Lucchese C., Marcuzzi F. – *Explainable Global Fairness Verification of Tree-Based Classifiers*, in IEEE Conference on Secure and Trustworthy Machine Learning (SaTML 2023)

Causal Discrimination

We focus on **individual fairness***: give similar predictions to similar individuals.
In particular, we focus on **lack of causal discrimination****.

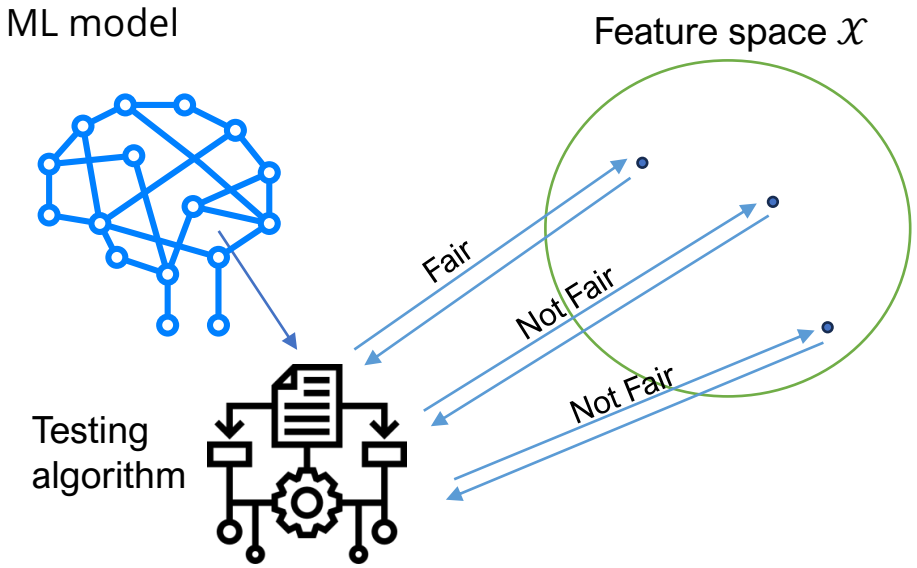


*S. Caton and C. Haas, *Fairness in machine learning: A survey*, 2020
**S. Galhotra, Y. Brun, and A. Meliou, *Fairness testing: testing software for discrimination*, ESEC/FSE 2017

SOTA of Fairness Verification

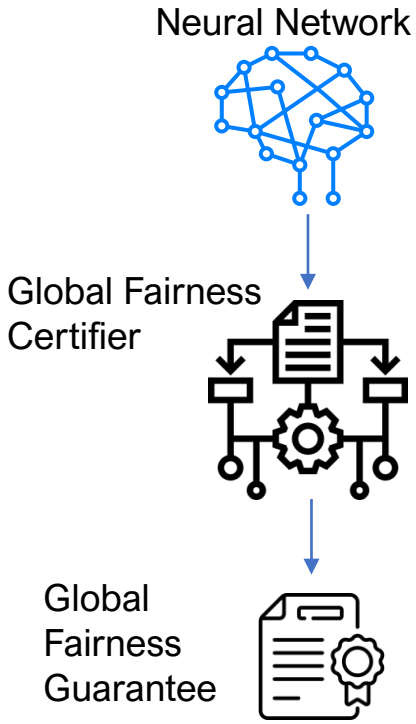
The **explainability** of the **guarantees** is **usually neglected…**

Fairness Testing*[1]

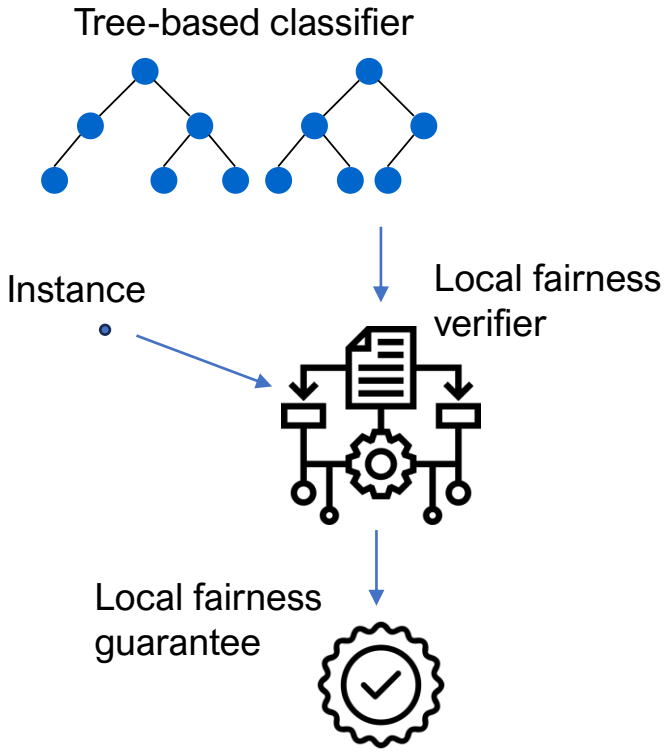


Under-approximated analysis!

Formal Fairness Verification*[2-3]



Only for Neural Networks!



Support only local properties!

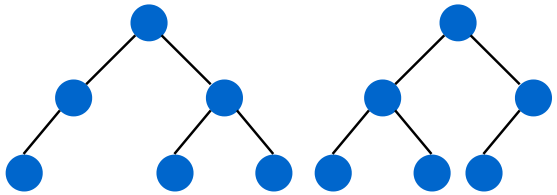
*[1]A. Aggarwal, P. Lohia, S. Nagar, K. Dey, and D. Saha, *Black-box fairness testing of machine learning models,* ESEC/SIGSOFT FSE 2019.

*[2]H. Khedr and Y. Shoukry, *Certifair: A framework for certified global fairness of neural networks,* 2022.
*[3]F. Ranzato, C.Urban and M.Zanella, *Fairness-aware training of decision trees by abstract interpretation,* CIKM '21 (2021).

Research problem

Problem:

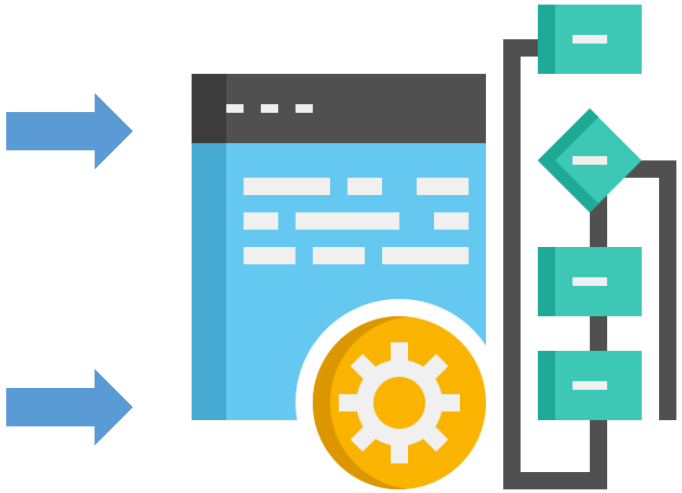
Tree based classifier T



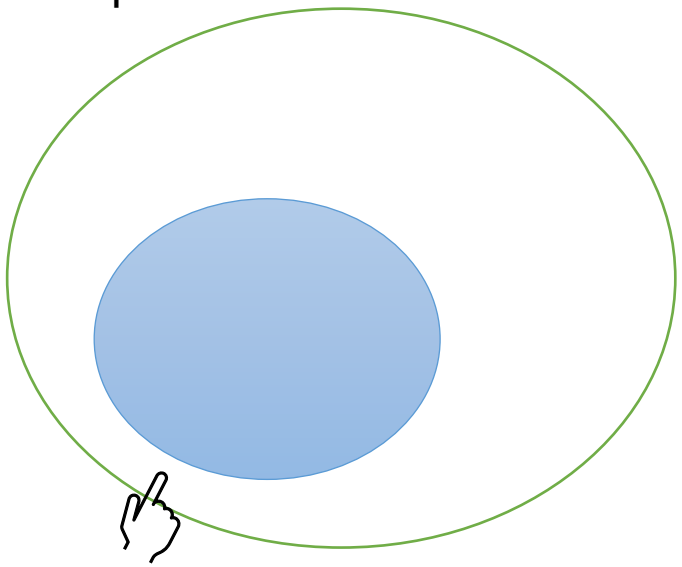
Set of sensitive features S

{sex, race}

Algorithm



Feature space $\mathcal{X}$



In this subset $\mathcal{X}' \subseteq \mathcal{X}$ T does not perform causal discimination w.r.t. S. The guarantee must be *explainable*.

Lack of Causal Discrimination and Stability

Lack of causal discrimination is connected to the **stability** property:

- Suppose to have an instance $\vec{x} \subseteq X$ and a set of possible adversarial manipulations $\mathrm{A}(\vec{x})$;
- f is *stable* on $\vec{x}$ if and only if $\forall \vec{z} \in A(\vec{x}): f(\vec{z}) = f(\vec{x})$. It's a **local** property.
- Lack of causal discrimination: changes to the sensitive features in S must not affect the predictions of the classifier.

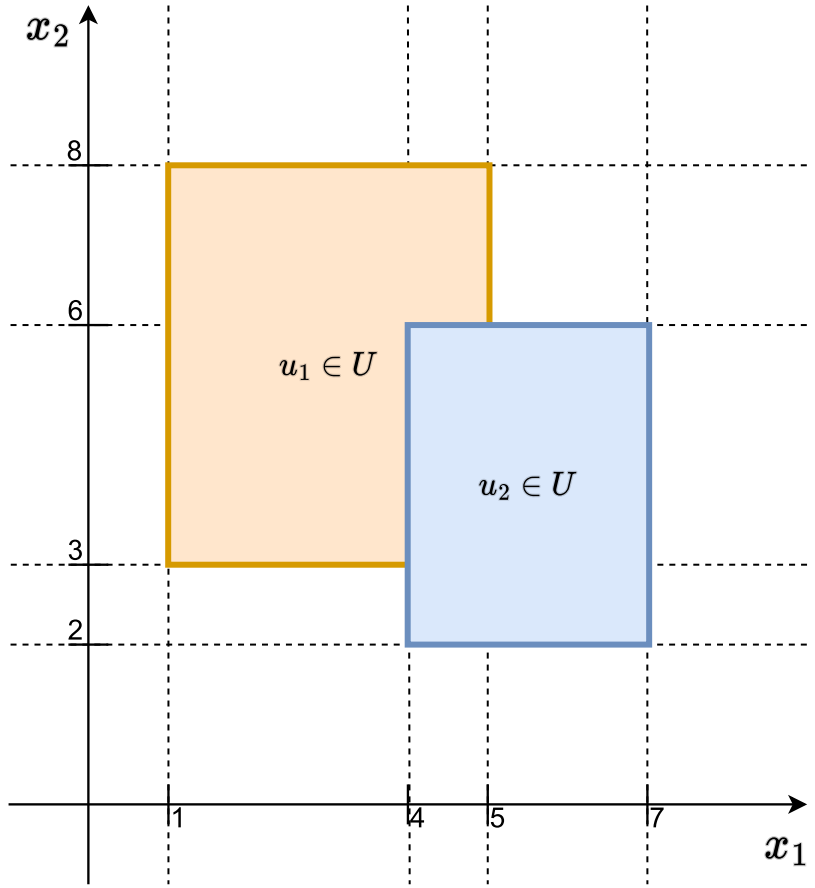
Data-Independent Stability Analysis

For tree-based models, exploit a

Data-Independent Stability Analysis algorithm (DISA)*:

- **Input**: tree-based model T and the definition of an attacker $A(\vec{x})$ (e.g., she manipulates the sensitive features of $\vec{x}$).
- **Output**: **set of hyper-rectangles U** that **over-approximates** the subsets of the feature space on which **T is unstable.**

T might perform causal discrimination on these subsets of the feature space!



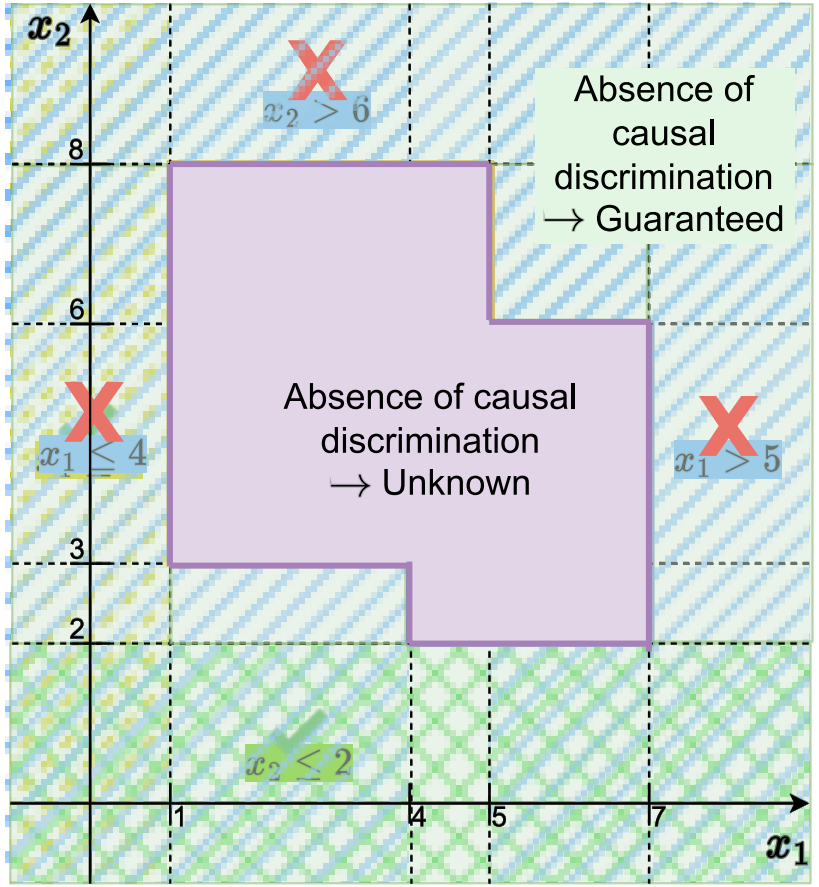
*S. Calzavara, L. Cazzaro, C. Lucchese, F. Marcuzzi, S. Orlando, *Beyond Robustness: Resilience Verification of Tree-Based Classifiers*, Computers&Security (2022)

Synthesis Algorithm – Generate Conditions

The synthesizer generates formulas **predicating on instances outside** the hyper-rectangles, i.e., where the ML classifier presents lack of causal discrimination!

The synthesis algorithm takes in input the set of hyper-rectangles U from the DISA:

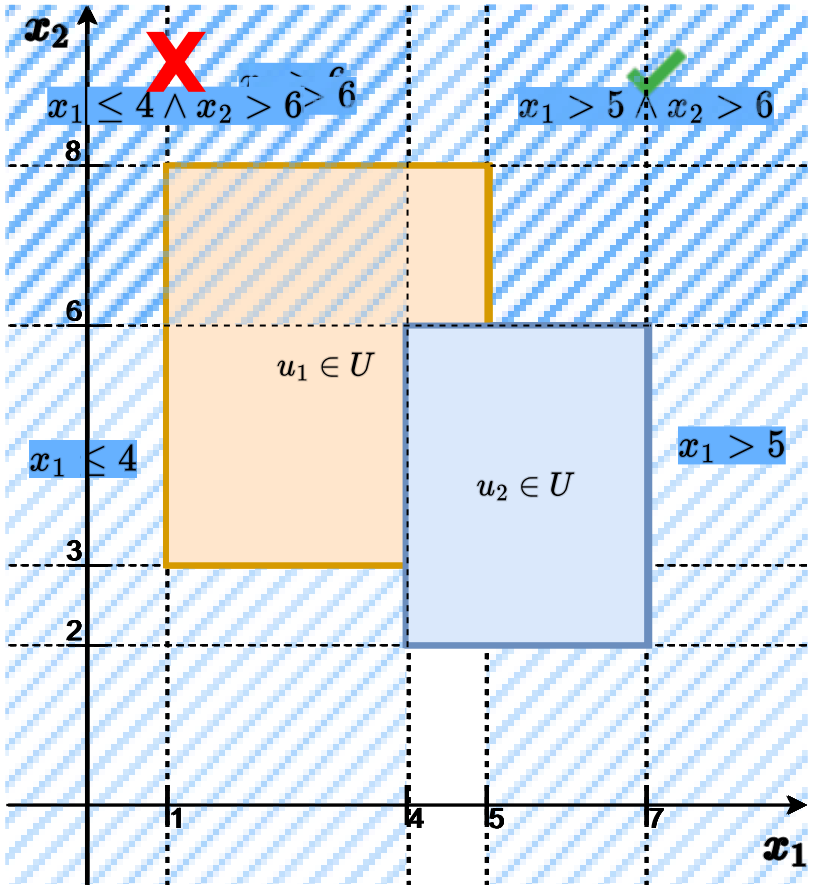
- It starts generating formulas with a predicate on **one single feature**.
- Check if some formulas of complexity 1 predicate only over instances outside the hyper-rectangles. Example: $x_1 \leq 1$.
- Some formulas may identify subsets of the feature space **that intersect some hyper-rectangles**.



Generate Longer Conditions

After the initial generation:

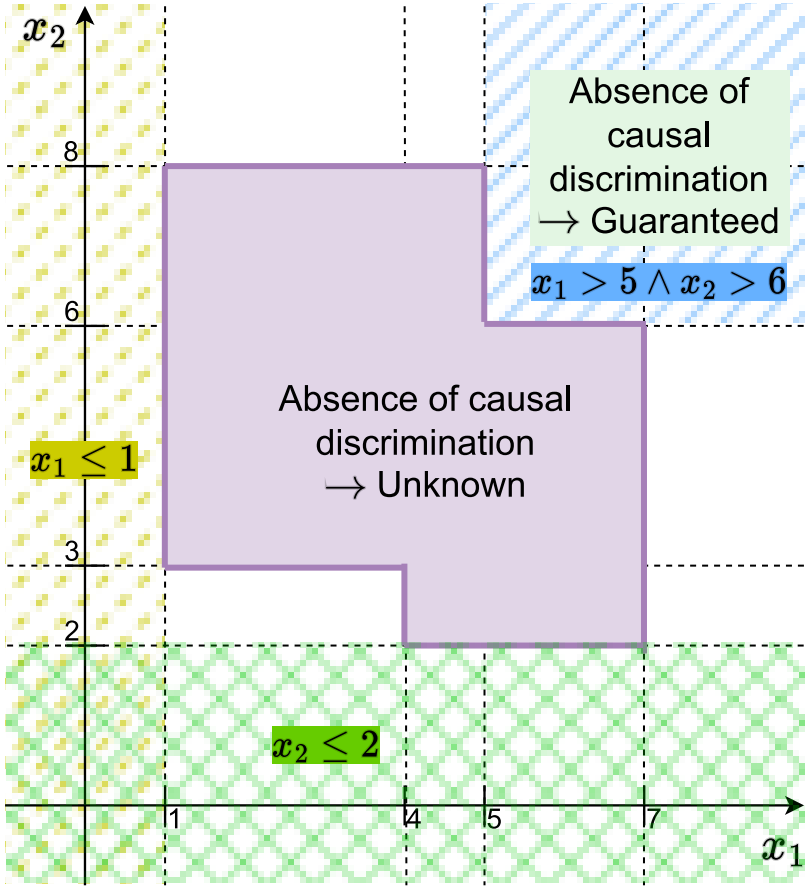
- Formulas that intersect hyper-rectangles are combined togheter to generate longer conditions. Example: $x_1 > 5 \wedge x_2 > 6$.
- **Check the new conditions** against the hyper-rectangles.
- **Continue performing the combination-check steps** until a stopping criteria is met (e.g., number of iterations).
- At the iteration k, formulas of complexity k are generated.



Synthesis algorithm - Summary

Our analyzer (based on another analyzer*):

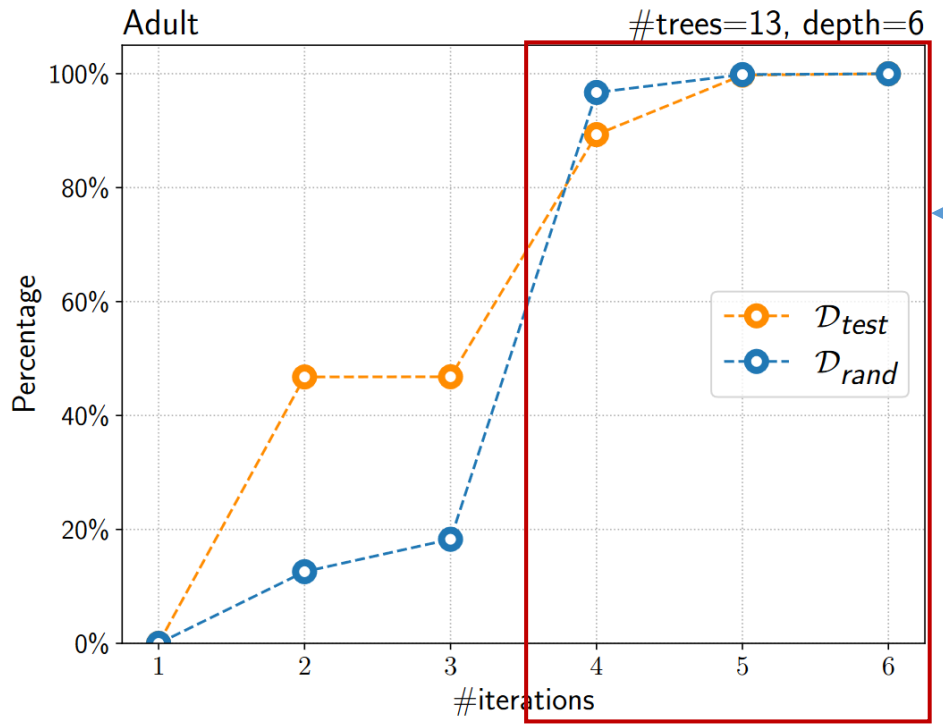
- Generates **increasingly complex sufficient conditions (logical formulas)** ensuring fairness.
- First iterations → formulas **easy to understand** (**explainable**).
- The more computational resources are available, the more complex conditions may be generated.
- We measure the **precision and the performance** of the analyzer and the **explainability of the results** of the analysis.



Experimental evaluation - Coverage

Question: how much is the subset of the feature space outside the hyper-rectangles (i.e., where the ML model is fair) covered by the conditions?

Method: we compute the **percentage of instances** covered by the fairness conditions.



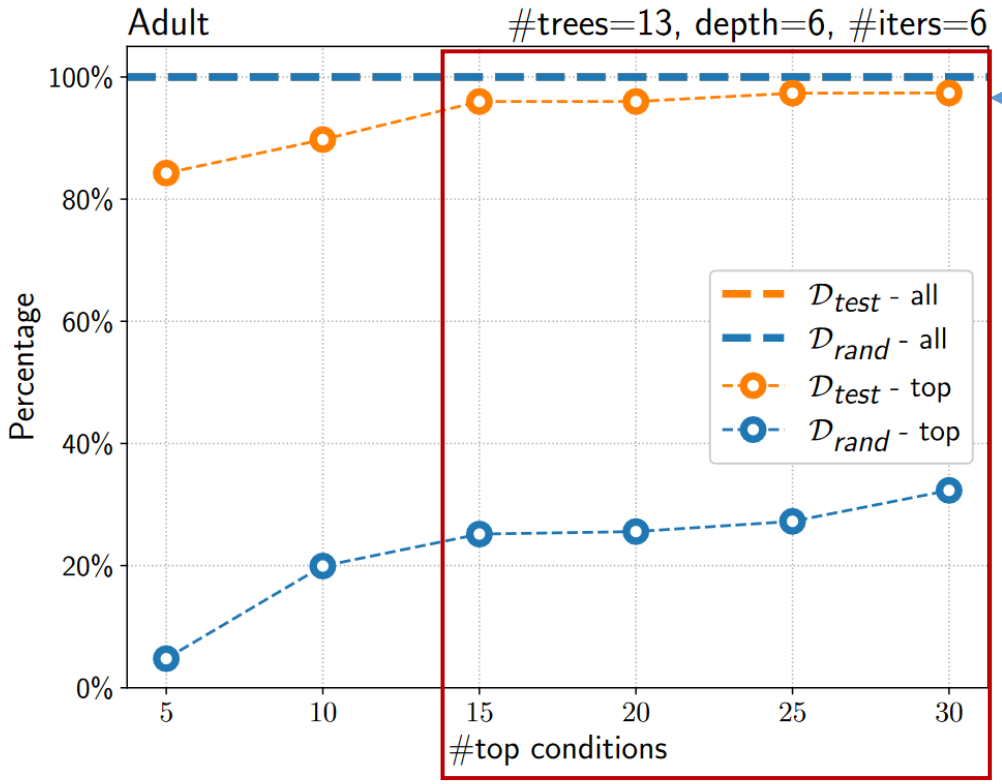
Answer: short logical formulas are expressive enough to establish useful fairness proofs!

Problem: *the number of generated formulas may increase significantly*, e.g., more than 300 formulas after 5 iterations.

Experimental evaluation – Top k formulas

Question: is a subset of the generated formulas sufficient to cover a «large» part of the subset of the feature space on which the ML model is fair?

Method: we select the set of the top k most important formulas using a greedy strategy.



A small number of formulas is sufficient to characterize the fairness guarantees on D_test.

More formulas are needed to cover synthetic instances in D_rand.

Answer: the number of important formulas **is relatively small** in practice!

Example

Our analysis synthesizes a set of sufficient conditions for fairness:

Conditions as **logical formulas**

Global conditions: predicate over the entire feature space

{age > 70 and job = «prof»,
credit_account < 4000 and age < 35 and housing = «rent»}

Explainable formulas: readily understandable

Our analysis is precise, explainable, reasonably efficient and proved sound and complete (details in the full paper)!

The analyzer is available on Github: https://github.com/LorenzoCazzaro/explainable-global-fairness-verification

Take-away messages

1. **Testing does not allow to verify global properties.**
2. The guarantees provided by a verifier should be also easily interpretable by an human and informative about the ML classifier.
3. Tools for verifying robustness (resilience) may be used to verify fairness and viceversa.
4. Our analyzer returns global fairness guarantees that are explainable, since they are logical formulas.
5. Our synthesizer requires a lot of time to return all the possible fairness guarantees, but you need to run the analysis only once!

Final Remarks

Conclusion

- We need expressive properties for defining the trustworthy behaviour of ML models and methods for verifying them.
- Defining data-independent properties that depend only on the structure of the classifiers allows us to define properties that hold globally instead of holding locally.
- A new security property, resilience, enables a more conservative security assessment of the ML models.
- A new tool for verifying fairness of tree-based classifiers enables the verification of the global fairness instead of local fairness.
- The proposed analyses are computationally expensive, but the user needs to run the analysis only once.

Future Work

- Characterize better the set of possible neighbors of an instance.
- Extend our results to gradient-boosted decision trees and other voting schemes.
- Train tree-based classifier that can exhibit a high resilience / global fairness.
- Generalize our findings to other model classes, e.g., neural networks.

Lorenzo Cazzaro
Ph.D. student in Computer Science

@LorenzoCazz

lorenzo.cazzaro@unive.it

Ca' Foscari University of Venice

Lorenzo Cazzaro

LorenzoCazzaro

https://lorenzocazzaro.github.io/

Thank you! Questions?