Robustness Verification of Multi-Class Tree Ensembles

Laurens Devos*, **Lorenzo Cascioli***, **Jesse Davis**

KU Leuven, Belgium,
laurens.devos@kuleuven.be, lorenzo.cascioli@kuleuven.be, jesse.davis@kuleuven.be

Abstract

Tree ensembles are one of the most widely used model classes. However, these models are susceptible to adversarial examples, which are slightly perturbed examples that elicit a misprediction. There has been significant research on designing approaches to verify the robustness of tree ensembles to such attacks. However, existing verification algorithms for tree ensembles are only able to analyze binary classifiers and hence address multiclass problems by reducing them to binary ones using a one-versus-other strategy. In this paper, we show that naively applying this strategy can yield incorrect results in certain situations. We address this shortcoming by proposing a novel approximate heuristic approach to verification for multiclass tree ensembles. Our approach is based on a novel generalization of the verification task, which we show emits other relevant verification queries.

Introduction

Tree ensembles such as gradient boosted trees and random forests are a powerful model class. They are frequently used in practice because they can tackle a variety of important real-world problems in diverse areas such as medicine (Ghiasi and Zendehboudi 2021), finance (Davis et al. 2020) and sports analytics (Decroos et al. 2019). Moreover, there are many easy to use and performant public implementations such as XGBoost (Chen and Guestrin 2016), LightGBM (Ke et al. 2017), CatBoost (Prokhorenkova et al. 2018) and BitBoost (Devos, Meert, and Davis 2020).

Unfortunately, like most flexible and expressive model classes, tree ensembles suffer from robustness issues. Namely, they are susceptible to evasion attacks (Kantchelian, Tygar, and Joseph 2016) where an adversary carefully crafts an example to elicit a misprediction from the model. This has spurred interest in developing approaches that are able to reason about learned ensembles to verify their robustness. A wide variety of approaches have been proposed for this task (Kantchelian, Tygar, and Joseph 2016; Devos, Meert, and Davis 2021b; Zhang, Zhang, and Hsieh 2020; Chen et al. 2019b).

Current algorithms are designed for binary classification problems. They only operate in multiclass settings by first reducing multiclass problems to a series binary ones. Specifically, they employ a one-versus-other strategy that contrasts the class of interest to one other class. This is then repeated for each possible other class. However, multiclass problems are an inherently richer and more complex setting. We show that the one-versus-other reduction has two substantial shortcomings. First, it does not return correct results. Essentially, the one-versus-other strategy reasons on the level of pairs of classes and cannot simultaneously reason about all classes. Second, there are additional possible attacks that existing algorithms cannot handle. For example, instead of just trying to elicit a misprediction, it is possible to attack the model in a targeted way where the constructed example will be predicted to belong to a specific class chosen by the adversary.

This paper addresses this gap by proposing a novel approach to verification in tree ensembles that directly operates in a multiclass setting. We take the approach of viewing robustness verification as a constrained optimization problem (Devos, Meert, and Davis 2021b) which can then be solved using a heuristic, anytime search. We generalize the optimization task to a multiclass setting and show that this yields four different classes of problems that are relevant. We propose a novel heuristic for solving the corresponding search problems and prove its consistency. Empirically, we demonstrate the efficacy of our approach on tree ensembles learned using both XGBoost and random forests. On benchmark datasets, we find that the current paradigm for performing robustness checking in multiclass tree ensembles returns incorrect results between 7% to 30% of the time.

In summary, we make the following five contributions: (1) we prove that the current approach to handling multiclass verification in tree ensemble can return incorrect results, (2) we propose a novel formulation of verification for multiclass tree ensembles, (3) we propose a new search heuristic for solving this framing and prove its consistency, (4) we empirically confirm our results, and (5) we provide an open-source implementation of our framework available on https://github.com/laudv/veritas.

Preliminaries

We asssume an input space $\mathcal{X} \subseteq \mathbb{R}^d$ and an output space $\mathcal{Y} = \{0, 1, \ldots, C-1\}$, with C the number of classes in the multiclass classification problem.

*These authors contributed equally.

Multiclass Decision Tree Ensembles

We will focus on multiclass verification of decision tree ensembles which includes popular machine learning algorithms such as gradient boosted decision trees (GBDTs) (Friedman 2001) and random forests (RFs) (Breiman 2001).

A decision tree consists of terminal *leaf nodes* storing a leaf value (scalar or vector) and *internal nodes* storing references to a left and right subtree and a less-than comparison $X_f < \tau$ on some attribute f and threshold τ. A tree is evaluated on an example x by recursively evaluating the less-than comparisons for x, starting from the *root node*, until a leaf node is reached whose value is the result of the evaluation. The evaluation moves left if the comparison is true, and right otherwise. The conjunction of the satisfied comparisons in the internal nodes along the root-to-leaf path denote a hypercube in the input space. We call this the leaf's *box*. For a leaf l, all examples $x \in \text{box}(l)$ evaluate to l. For example, the box of the leaf with value -14 in Figure 2 is $(\text{HEIGHT} \geq 200 \wedge \text{AGE} < 50)$. In general, a box has the form $\bigwedge_f (l_f \leq X_f < u_f)$, with l_f the lower and u_f the upper bound on the feature X_f (possibly $\pm\infty$).

An ensemble of M decision trees $\boldsymbol{T}$ combines the predictions of individual trees. We distinguish three output forms of the ensemble: the raw scores $\boldsymbol{T}^{\text{raw}}(x) = \nu_0 + \sum_{m=1}^{M} T_m(x)$ which are simply the sum of the predictions of the trees plus a constant base score ν_0, the derived class probabilities $\boldsymbol{T}^{\text{prob}}(x)$, and the hard labels $\boldsymbol{T}^{\text{label}}(x) = \arg\max_c \boldsymbol{T}_c^{\text{prob}}(x)$, with $\boldsymbol{T}_c^{\text{prob}}(x)$ the probability of class c for example x, $c = 0, \ldots, C-1$. How $\boldsymbol{T}^{\text{prob}}(x)$ is derived from the raw scores depends on ensemble type.

The leaf values in a RF are vectors of size C storing the probabilities of each class as illustrated in Figure 1 for a multiclass problem with three classes. A probability vector over the full ensemble is obtained by taking the average of the trees' predicted probability vectors: $\boldsymbol{T}^{\text{prob}}(x) = \boldsymbol{T}^{\text{raw}}(x)/M$.

A multiclass GBDT ensemble $\boldsymbol{T}$ consists of C one-versus-rest ensembles $\boldsymbol{T}_c(x)$ with $\boldsymbol{T}_c^{\text{raw}}(x) = \sum_m T_{c,m}(x)$. The one-versus-rest classifier for class c $\boldsymbol{T}_c(x)$ considers class c as the positive class and all remaining classes as one combined negative class. The raw scores $\boldsymbol{T}^{\text{raw}}(x)$ for the multiclass ensemble are obtained by separately summing up the trees of the class-specific ensembles and combining them into a vector: $\boldsymbol{T}^{\text{raw}}(x) = [\boldsymbol{T}_0^{\text{raw}}(x), \ldots, \boldsymbol{T}_{C-1}^{\text{raw}}(x)]$. The probability vector is derived from the raw scores by applying the softmax: $\boldsymbol{T}^{\text{prob}}(x) = \text{softmax}(\boldsymbol{T}^{\text{raw}}(x))$. For binary classification ($C = 2$), building both $\boldsymbol{T}_0$ and $\boldsymbol{T}_1$ would be redundant, so GBDT systems drop $\boldsymbol{T}_0$ and only construct $\boldsymbol{T}_1$ for the positive class. The probabilities are then $\boldsymbol{T}_1^{\text{prob}}(x) = \text{sigmoid}(\boldsymbol{T}_1^{\text{raw}}(x))$ and $\boldsymbol{T}_0^{\text{prob}}(x) = 1 - \boldsymbol{T}_1^{\text{prob}}(x)$.

Robustness Verification of Binary Classification Ensembles

A major weakness of many machine learning algorithms is their susceptibility to *evasion attacks*. In an evasion attack, an adversary slightly perturbs a valid base example x_b into an example x in order to elicit a misprediction. More formally, we call x a valid adversarial example for x_b and binary classifier $\boldsymbol{T}$ when (1) $\boldsymbol{T}^{\text{label}}(x_b)$ is the correct label, (2) $\boldsymbol{T}^{\text{label}}(x) \neq \boldsymbol{T}^{\text{label}}(x_b)$, and (3) $\|x - x_b\|_\infty < \delta$, i.e., the two are sufficiently close using the ∞-norm with δ a predefined maximum allowed distance.

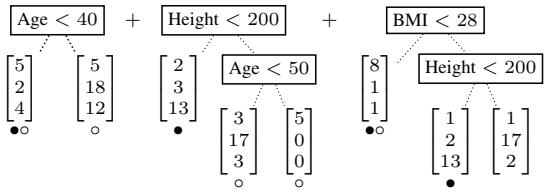


Figure 1: An example multiclass RF ensemble with $C = 3$. For ease of readability of the example, the leaf values are absolute counts for the classes. Note that the counts in the leaves do not have to be consistent between the trees because RFs use bootstrapping and each tree is trained on a different subset of the data.

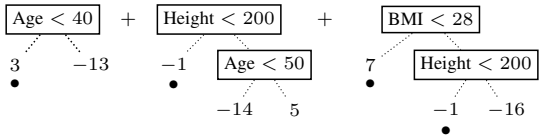


Figure 2: The 0-versus-1 ensemble derived from Figure 1

An ensemble's robustness against evasion attacks is often assessed by measuring how easy it is to generate adversarial examples using metrics like adversarial robustness (fraction of correctly classified example for which no adversarial example exists (Vos and Verwer 2021)) and empirical robustness (average distance to the closest adversarial example (Kantchelian, Tygar, and Joseph 2016)). Evaluating these metrics requires generating a large number of adversarial examples on a test set. Consider a base example x_b with correctly predicted label c. Take $c' = 1 - c$ the opposite label, then adversarial example generation for binary classification can be formulated as the following search problem:

$$\text{find } x \in \mathcal{X} \text{ such that } \boldsymbol{T}^{\text{label}}(x) \neq \boldsymbol{T}^{\text{label}}(x_b) \text{ and } \|x - x_b\|_\infty < \delta. \quad (1)$$

If an x is found so that $\boldsymbol{T}^{\text{label}}(x) \neq \boldsymbol{T}^{\text{label}}(x_b)$, then x is an adversarial example of x_b. If no such x can be found, then x_b is verified to be δ-robust because no adversarial example exists within distance δ from x_b. This concept can be extended to the multiclass case. A multiclass model is δ-robust for a base example x_b when it is impossible to find an adversarial example x such that $\|x - x_b\|_\infty < \delta$ and $\boldsymbol{T}^{\text{label}}(x) \neq \boldsymbol{T}^{\text{label}}(x_b)$. A multiclass model is δ-robust to a *targeted attack* on class c for a base example x_b when it is impossible to find an x such that $\|x - x_b\|_\infty < \delta$ and $\boldsymbol{T}^{\text{label}}(x) = c$.

VERITAS: Approximate Search-Based Tree Ensemble Verification

VERITAS (Devos, Meert, and Davis 2021b) is a state-of-the-art robustness verification approach. It reformulates the ad-

versarial search problem in Equation 1 as an optimization problem constrained by $\mathcal{C}$. Take $\boldsymbol{T}$ to be a binary classifier:

$$\underset{x \in \mathcal{X}}{\text{optimize}}\, \boldsymbol{T}_1^{\text{prob}}(x) \quad \text{subject to} \quad \mathcal{C}(x). \tag{2}$$

VERITAS uses an approximate heuristic search to solve Equation 2 and constructs an example x which satisfies $\mathcal{C}(x)$ and optimizes (maximize or minimize) the ensemble's output $\boldsymbol{T}_1^{\text{prob}}(x)$. This can be used to solve Equation 1. Take negatively classified base example x_b, and let $\mathcal{C}(x)$ constrain x to $\|x - x_b\|_\infty < \delta \wedge \boldsymbol{T}_1^{\text{prob}}(x) > 0.5$. Then maximize $\boldsymbol{T}_1^{\text{prob}}(x)$ using VERITAS. If it finds such an x, then x is an adversarial example for x_b. For a positive base example, the problem is changed to minimize $\boldsymbol{T}_1^{\text{prob}}(x)$ with the constraint $\boldsymbol{T}_1^{\text{prob}}(x) < 0.5$.

While $\mathcal{C}(x)$ can represent more complex constraints (see Devos, Meert, and Davis (2021b)), for ease of presentation, we limit $\mathcal{C}(x)$ to box constraints like the ∞-norm, and to constraints on the ensemble's output. We call the box constraint the **prune box** PB.

VERITAS uses a search procedure to find such an x. Search procedures are generally defined over a search space consisting of **search states**. The search maintains an *open* list of states – initially only containing the **initial state** – and repeatedly **expands** the current best state in the *open* list according to a heuristic **scoring function**. The search repeats this procedure until a **solution state** is found (Hart, Nilsson, and Raphael 1968). For a $\max$ problem, these components are defined for VERITAS as follows:

1. A **search state** s has a $\mathrm{box}(s)$ which constrains the input space $\mathcal{X}$. The box restricts which leaves are still reachable in the trees of the ensemble (i.e., the leaves' boxes are compatible with $\mathrm{box}(s)$). Let $L_m(s)$ denote the set of reachable leaves in tree T_m given state s: $L_m(s) = \{l_m \mid l_m \text{ is a leaf of tree } T_m \text{ and } \mathrm{box}(l_m) \cap \mathrm{box}(s) \neq \emptyset\}$. We then partition the trees of the ensemble into a set of *resolved* trees T_s^{R} and another of *unresolved* trees T_s^{UR}. A tree T_m is resolved if all possible $x \in \mathrm{box}(s)$ lead to one unique leaf l_m, i.e., $|L_m(s)| = 1$, and equivalently $\mathrm{box}(l_m) \supseteq \mathrm{box}(s)$. A tree T_m is unresolved if multiple leaves are still reachable, i.e., $|L_m(s)| > 1$. For example, assume a state with box $(\text{AGE} < 40 \wedge \text{HEIGHT} < 200)$. Then, the first two trees in Figure 1 are resolved. The third tree is unresolved because there are two reachable leaves (indicated with a •).
2. The **initial state** is the state with the prune box as its box.
3. The **scoring function** $f(s) = g(s) + h(s)$ for a state s guides the search towards promising search states leading to solutions with high values for $\boldsymbol{T}(x)$. The first part $g(s)$ sums up the base score ν_0 and the leaf values of the leaves reached in the resolved trees. The second part is the heuristic $h(s)$ which overestimates the value that can still be added to the output of the ensemble. This is computed by summing up the maximum of the leaf value ν_m of the still reachable leaves $l_m \in L_m(s)$ in each unresolved tree $T_m \in T_s^{\mathrm{UR}}$:

$$g(s) = \nu_0 + \sum_{T_m \in T_s^{\mathrm{R}}} \nu_m, \tag{3}$$

$$h(s) = \sum_{T_m \in T_s^{\mathrm{UR}}} \max_{l_m \in L_m(s)} \nu_m. \tag{4}$$

4. The **expand function** $E(s)$ produces new successor states s', one for each reachable leaf $l_m \in L_m(s)$ of an unresolved tree $T_m \in T_s^{\mathrm{UR}}$, with $\mathrm{box}(s') = \mathrm{box}(s) \cap \mathrm{box}(l_m)$. The new states are added to the *open* list and evaluated by the scoring function. The tree T_m is a resolved tree in all expanded states s'. States are constructed by repeated expansion, so $\mathrm{box}(s) = \mathrm{PB} \cap \bigcap_{T_m \in T_s^{\mathrm{R}}} \mathrm{box}(l_m)$, with l_m the only reachable leaf in the resolved tree T_m.
5. A **solution state** s is reached when all trees are resolved. The output value of this solution state is $f(s) = g(s) =$ the sum of the base score and the leaf values. Just as any other state, a solution state has a box. Any $x \in \mathrm{box}(s)$ has output $f(s)$, that is, one can pick any example $x \in \mathrm{box}(s)$ to obtain a concrete solution.

The VERITAS paper proves that the heuristic is consistent for the $\max$ problem (replace $\max$ by $\min$ in $h(s)$ for a $\min$ problem). Hence, the search is guaranteed to always return the optimal solution if given enough time and memory. In practice, because this is an NP-hard problem (Kantchelian, Tygar, and Joseph 2016), this can take excessively long, so an approximation is used (Pearl and Kim 1982). When looking for the next state to expand, all states with a score within ε of the current best state are re-evaluated by a non-consistent scoring function. The best state according to the second score is expanded and its successor states are added to the open list, which is sorted again by the consistent heuristic. The second heuristic can, for example, give precedence to deeper solutions. This guarantees that (1) a solution is found quickly, and (2) this solution is no more than ε from the optimal solution.

Multiclass Verification in Tree Ensembles

Understanding robustness verification in a multiclass setting requires generalizing our notion of an evasion attack. While an adversary may have multiple goals, we will consider the following two objectives. Given a correctly classified base example x_b, generate a small perturbation Δ to:

- **A1.** Simply elicit a misprediction, i.e., $\boldsymbol{T}^{\text{label}}(x_b + \Delta) \neq \boldsymbol{T}^{\text{label}}(x_b)$;
- **A2.** Elicit a targeted misprediction, i.e., $\boldsymbol{T}^{\text{label}}(x_b + \Delta) = c \neq \boldsymbol{T}^{\text{label}}(x_b)$, where c is a class selected by the adversary.

The second goal, which we call the *targeted evasion problem*, is a stronger requirement than the first one. In order to predict class c, it must be the case that $\boldsymbol{T}_c^{\text{prob}}(x) > \boldsymbol{T}_{c'}^{\text{prob}}(x)$ for all $c' \neq c$. We can write this as the following double optimization problem. Without loss of generality and for ease

of notation, we take the target class $c = 0$ (you can always reorder the classes):

$$\max_{x \in \mathcal{X}} \left[\boldsymbol{T}_0^{\text{prob}}(x) - \max_{c \neq 0} \boldsymbol{T}_c^{\text{prob}}(x) \right] \quad \text{subject to} \quad \mathcal{C}(x). \quad (5)$$

If it is possible to find an example x for which this objective is greater than 0, then the model is guaranteed to predict class 0 for that example. If no such x can be found, then the ensemble never predicts class 0 for any x that satisfies the constraints $\mathcal{C}(x)$.

Previous work has reduced multiclass problems to multiple binary ones using a one-versus-other strategy (Chen et al. 2019b; Zhang, Zhang, and Hsieh 2020; Devos, Meert, and Davis 2021b).[1] The one-versus-other strategy spawns $C-1$ subproblems, each one contrasting the target class c with another class $c' \neq c$ by creating a new ensemble $\boldsymbol{T}_{c\text{-vs-}c'}(x)$ that outputs the difference between $\boldsymbol{T}_c^{\text{raw}}(x)$ and $\boldsymbol{T}_{c'}^{\text{raw}}(x)$, i.e., a binary classifier that outputs the positive class if $\boldsymbol{T}$ selects class c over c'. For a RF, this is simply an ensemble with the same structure as the multiclass ensemble, but the leaf values are the difference between the class probabilities for c and c'. As an example, $\boldsymbol{T}_{0\text{-vs-}1}$ is shown in Figure 2 for the RF in Figure 1. For a GBDT, this new ensemble is the difference between the one-versus-rest classifiers so that $\boldsymbol{T}_{c\text{-vs-}c'}^{\text{raw}}(x) = \sum_m T_{c,m}(x) + \sum_m -T_{c',m}(x)$, where $-T_{c',m}(x)$ is the tree with the sign of each leaf value flipped.

Any verification method that works on a binary classification ensemble can be applied to this new ensemble. For example, VERITAS can be used to optimize Equation 2. If an x is found for which the objective value is greater than 0, then we know that $\boldsymbol{T}_c^{\text{prob}}(x) > \boldsymbol{T}_{c'}^{\text{prob}}(x)$. If no such x can be found, then it is never possible that the ensemble outputs a probability for target class c that is greater than that for the other class c' for any x that satisfies $\mathcal{C}(x)$.

Consider the *targeted evasion decision problem* which asks the question whether for ensemble $\boldsymbol{T}$ an x exists such that (1) x satisfies a set of constraints $\mathcal{C}(x)$, and (2) $\boldsymbol{T}$ predicts class 0. This is the case when $\boldsymbol{T}_0^{\text{prob}}(x) > \boldsymbol{T}_c^{\text{prob}}(x)$ for all $c \neq 0$. We prove that the targeted evasion decision problem cannot be reduced to $C-1$ one-versus-other problems.

Theorem 1. *Consider a multiclass ensemble $\boldsymbol{T}(x)$ with $C > 2$ classes. Assume there exist examples $x_{0\text{-vs-}c} \in \mathcal{X}$, one for each $c \in \mathcal{Y}, c \neq 0$, such that $\boldsymbol{T}_{0\text{-vs-}c}^{raw}(x_{0\text{-vs-}c}) > 0$, or also, $\boldsymbol{T}_0^{prob}(x_{0\text{-vs-}c}) > \boldsymbol{T}_c^{prob}(x_{0\text{-vs-}c})$. The existence of $x_{0\text{-vs-}c}$, $c = 1, \ldots, C-1$ does not imply that there exists an $x_0 \in \mathcal{X}$ such that $\boldsymbol{T}_0^{prob}(x_0) > \boldsymbol{T}_c^{prob}(x_0)$ for all $c \neq 0$. That is, their existence does **not** imply that a solution to the targeted evasion decision problem exists.*

Proof. We prove Theorem 1 by counter-example using the ensemble in Figure 1. We describe the subset of the input space for which $\boldsymbol{T}_{0\text{-vs-}1}^{\text{raw}}(x) > 0$ and $\boldsymbol{T}_{0\text{-vs-}2}^{\text{raw}}(x) > 0$ respectively:[2]

$$\boldsymbol{T}_{0\text{-vs-}1}^{\text{raw}}(x) > 0 \text{ iff } x \in (\text{AGE} < 40 \wedge \text{HEIGHT} < 200),$$
$$\boldsymbol{T}_{0\text{-vs-}2}^{\text{raw}}(x) > 0 \text{ iff } x \in$$
$$(\text{AGE} < 40 \wedge \text{BMI} < 28 \wedge \text{HEIGHT} \geq 200) \cup$$
$$(\text{AGE} \geq 50 \wedge \text{BMI} < 28 \wedge \text{HEIGHT} \geq 200).$$

The reachable leaves are indicated with a • (0-versus-1) and a ∘ (0-versus-2). Picking any other leaf pushes the prediction below 0. So there exist $x_{0\text{-vs-}1}$ and $x_{0\text{-vs-}2}$ for which $\boldsymbol{T}$ prefers class 0 over class 1 and 2 respectively.

A solution to the targeted evasion problem is an x_0 for which both $\boldsymbol{T}_{0\text{-vs-}1}^{\text{raw}}(x_0) > 0$ *and* $\boldsymbol{T}_{0\text{-vs-}2}^{\text{raw}}(x_0) > 0$, i.e., any x_0 that is in both the subsets above. However, the intersection of the two subsets is empty because of the contradicting constraint on HEIGHT. This proves that no such x_0 exists despite the fact that $x_{0\text{-vs-}1}$ and $x_{0\text{-vs-}2}$ exist. □

None of the examples in either of the subsets above is a solution to the targeted evasion problem because their predicted label is not class 0. For example, the following two examples have $\boldsymbol{T}_{0\text{-vs-}1}^{\text{raw}}(x) > 0$ and $\boldsymbol{T}_{0\text{-vs-}2}^{\text{raw}}(x) > 0$ respectively, but neither of them has predicted label 0:

$$\boldsymbol{T}^{\text{raw}}(\text{AGE} = 32, \text{HEIGHT} = 175, \text{BMI} = 20) = [15, 6, 18],$$
$$\boldsymbol{T}^{\text{raw}}(\text{AGE} = 32, \text{HEIGHT} = 201, \text{BMI} = 20) = [16, 20, 8].$$

The numbers inside the square brackets are the class counts returned by the RF. For the first example, $15-6 > 0$, i.e., the count for class 0 is higher than the count for class 1, so it is a solution to the 0-versus-1 subproblem. However, the highest count, 18 is associated with class 2, so the ensemble will predict this class and not class 0. For the second, $16 - 8 > 0$, so, again, it is a solution to the 0-versus-2 subproblem. However, the highest count is 20, which means that class 1 will be the predicted label and not class 0.

Appendix shows an alternative counter-example where, just like in the example above, two one-versus-other solutions $x_{0\text{-vs-}1}$ and $x_{0\text{-vs-}2}$ exist and $\boldsymbol{T}$ predicts class 0 for neither of them. However, in that example, there still exists another x for which the predicted label is class 0. This illustrates that, unless one of the one-versus-other solutions is a satisfying solution to the targeted evasion problem, knowing that each one-versus-other subproblem is satisfiable is **inconclusive** with respect to the targeted evasion problem: both outcomes are still possible.

Intuitively, the root issue is that each one-versus-other subproblem is solved independently. For example, the subproblem for class $c = 1$ involves searching for an x such that $\boldsymbol{T}_0^{\text{prob}}(x) > \boldsymbol{T}_1^{\text{prob}}(x)$ while ignoring $\boldsymbol{T}_2^{\text{prob}}(x)$. Consequently, if the search finds an x where $\boldsymbol{T}_0^{\text{prob}}(x) > \boldsymbol{T}_1^{\text{prob}}(x)$, then it is possible that $\boldsymbol{T}_{c'}^{\text{prob}}(x) > \boldsymbol{T}_0^{\text{prob}}(x)$ for some $c \neq 1$. More generally, the solutions to all $C-1$ subproblems are disconnected from each other which leads to solutions as illustrated in the counter-examples where each subproblem has a solution, but none of them simultaneously satisfies all constraints.

[1] Note the difference between one-versus-rest and one-versus-other. The former is a learning strategy used by GBDT systems. The latter is a multiclass verification strategy.

[2] In practice, generating these subsets is intractable, even for one-versus-other subproblems.

Method

As attack **A2** cannot simply be reduced to multiple one-versus-other subproblems, we need approaches that are able to reason about all classes simultaneously. To address this shortcoming, we propose a novel generalization of VERITAS that allows it to reason about multiclass problems. Specifically, this entails modifying the heuristic scoring function used during its search process. Moreover, this generalization also permits reasoning about several other verification tasks.

We reuse the search procedure of VERITAS as outlined in the preliminaries, and modify the scoring function $f(s) = g(s) + h(s)$ in order to optimize the multiclass objective in Equation 5:

$$g_c(s) = \nu_{0,c} + \sum_{T_m \in T_s^{\mathrm{R}}} \nu_{m,c} \tag{6}$$

$$h_0(s) = \sum_{T_m \in T_s^{\mathrm{UR}}} \max_{l_m \in L_m(s)} \nu_{m,0} \tag{7}$$

$$h_c(s) = \sum_{T_m \in T_s^{\mathrm{UR}}} \min_{l_m \in L_m(s)} \nu_{m,c}, \quad c \neq 0 \tag{8}$$

$$c^* = \arg\max_{c \neq 0} g_c(s) + h_c(s) \tag{9}$$

$$g(s) = g_0(s) - g_{c^*}(s) \tag{10}$$

$$h(s) = h_0(s) - h_{c^*}(s). \tag{11}$$

We use $\nu_{m,c}$ to denote the leaf value for class c in a leaf l_m in tree T_m. $L_m(s)$ is the set of reachable leaves in the unresolved tree T_m in the search state s.

Equation 6 sums the leaf values for class c of the leaves visited in the resolved trees. The heuristic $h_c(s)$ in Equations 7 and 8 looks at which leaves are still reachable in each unresolved tree. It takes the maximum of the leaf values for class 0, and the minimum for class $c \neq 0$. We compute the minimum of the remaining reachable leaf values for $c \neq 0$ to get a heuristic that overestimates the outer maximization problem: the *best case* for the outer maximization in Equation 5 occurs when the subtracted inner maximization term is minimized. Once we have the values $g_c(s)$ and $h_c(s)$ for each class c, we determine the winning class c^* of the inner maximization problem (Equation 9), and compute the $g(s)$ and $h(s)$ values using c^*.

Proof of Consistency

We prove that the heuristic introduced in the previous subsection is consistent with respect to the optimization problem in Equation 5.

Theorem 2 (Consistency of the multiclass heuristic)**.** *The heuristic in Equation 11 is consistent with respect to the multiclass optimization problem in Equation 5. That is, for any search state s with $\mathrm{box}(s)$ and successor state s' with $\mathrm{box}(s') = \mathrm{box}(s) \cap \mathrm{box}(l^\circ_{m'})$ obtained by expanding an unresolved tree $T^\circ_{m'}$ for leaf $l^\circ_{m'}$ with leaf values $\nu^\circ_{m',c}$, $c = 0, \ldots, C-1$, it holds that:*

$$h(s) \geq \nu^\circ_{m',0} - \nu^\circ_{m',c^*} + h(s'). \tag{12}$$

We use two following propositions to prove this:

Proposition 1 (h_0 is an overestimation)**.** *For s, s', $T^\circ_{m'}$, $l^\circ_{m'}$, $\nu^\circ_{m',c}$ as in Theorem 2, h_0 is an overestimation:*

$$h_0(s) \geq h_0(s') + \nu^\circ_{m',0}. \tag{13}$$

Proof. We look at each term in the sum of Equation 7 for $h_0(s)$, one for each $T_m \in T_s^{\mathrm{UR}}$. We split these up in three subsets. (1) For all $T_m \in T_{s'}^{\mathrm{UR}}$, $\max_{l_m \in L_m(s)} \nu_{m,0} \geq \max_{l_m \in L_m(s')} \nu_{m,0}$. This is true because $L_m(s') \subseteq L_m(s)$ since if $\mathrm{box}(s') \cap \mathrm{box}(l_m) = \mathrm{box}(s) \cap \mathrm{box}(l^\circ_{m'}) \cap \mathrm{box}(l_m) \neq \emptyset$, then $\mathrm{box}(s) \cap \mathrm{box}(l_m) \neq \emptyset$. (2) Any term for a $T_m \in T_s^{\mathrm{UR}} \setminus (T_{s'}^{\mathrm{UR}} \cup \{T^\circ_{m'}\})$ does not affect $h_0(s')$ and only makes $h_0(s)$ larger. (3) For the expanded tree $T^\circ_{m'}$, which is resolved in s' and unresolved in s, $\max_{l_m \in L_m(s)} \geq \nu^\circ_{m',0}$ because $l^\circ_{m'} \in L_m(s)$. Note that all $\nu_{m,c} \geq 0$.[3] □

Proposition 2 (h_c is an underestimation)**.** *For s, s', $T^\circ_{m'}$, $l^\circ_{m'}$, $\nu^\circ_{m',c}$ as in Theorem 2, h_c is an overestimation, $c = 1, \ldots, C-1$:*

$$h_c(s) \leq h_c(s') + \nu^\circ_{m',c}. \tag{14}$$

Proof. Analogous to the proof of Proposition 1 with $\max$ and $\geq$ replaced by $\min$ and $\leq$. □

We can now prove Theorem 2.

Proof of Theorem 2. We use Proposition 1 and 2 to rewrite the definition of $h(s)$ in Equation 11:

$$\begin{aligned} h(s) &= h_0(s) - h_c(s) \\ &\geq \big(h_0(s') + \nu^\circ_{m',0}\big) - \big(h_c(s') + \nu^\circ_{m',c}\big) \\ &= \nu^\circ_{m',0} - \nu^\circ_{m',c} + h(s'). \end{aligned} \tag{15}$$

We can substitute Equation 14 in Equation 15 because $h_c(s)$ is negated. This result holds for all $c = 1, \ldots, C-1$, including c^*. This concludes the proof. □

Variants of Multiclass Optimization Formulation

We consider three variants of the multiclass objective in Equation 5. All x, x^* and x' are assumed to satisfy $\mathcal{C}(x)$. Replacing the inner $\max$ by a $\min$ yields:

$$\max_{x \in \mathcal{X}} \left[\boldsymbol{T}_0^{\text{prob}}(x) - \min_{c \neq 0} \boldsymbol{T}_c^{\text{prob}}(x) \right] \quad \text{s.t.} \quad \mathcal{C}(x). \tag{16}$$

If an x^* exists for which the above objective value is greater than zero, then $\boldsymbol{T}$ prefers class 0 over at least one other class $c \neq 0$ for that x^*. If no such x^* exists, then for all x, every other class $c \neq 0$ (not just one) is preferred over class 0.

Replacing the outer $\max$ by $\min$ in Equation 5:

$$\min_{x \in \mathcal{X}} \left[\boldsymbol{T}_0^{\text{prob}}(x) - \max_{c \neq 0} \boldsymbol{T}_c^{\text{prob}}(x) \right] \quad \text{s.t.} \quad \mathcal{C}(x). \tag{17}$$

If an x^* exists for which the objective value is greater than zero, then that x^* has $\boldsymbol{T}^{\text{label}}(x) = 0$. Additionally, all other $x' \neq x^*$ have a greater or equal objective value. Hence, all x have $\boldsymbol{T}^{\text{label}}(x) = 0$. If no such x^* exists, then there must

[3] This can easily be achieved by exploiting the base score ν_0. Any negative leaf value is eliminated by adding a value to the tree's leaf values and subtracting it from the base score.

	Data properties		GBDT			RF	
	#F, #ex, C	δ	M	d	η	M	d
img	2, 10.0k, 4	0.10	10	6	0.5	10	5
wine	12, 32.5k, 7	0.02	20	14	0.2	40	15
bean	16, 13.6k, 7	0.05	20	4	0.5	20	10
mnist	784, 70.0k, 10	0.04	10	6	0.8	20	14
letter	16, 20.0k, 26	0.05	20	5	0.8	30	14
pend.	16, 11.0k, 10	0.12	10	5	0.8	25	12
sens.	48, 58.5k, 11	0.08	5	5	0.8	10	6

Table 1: Properties of each dataset: number of features #F, examples #ex and classes C, and the maximum perturbation size δ. Parameters: number of trees M, maximum depth d, and learning rate η. GBDTs learn C one-versus-rest classifiers, so the total number of trees is $M \times C$.

be an x' for which the predicted label is not class 0. This is an elegant formulation of multiclass robustness checking. Assume class 0 is the true label of a base example x_b. If there is an x^* that has an objective value greater than 0, then it is certain that no example exists within the region defined by $\mathcal{C}(x)$ with a different predicted label.

Finally, replacing the inner and the outer $\max$ by $\min$:

$$\min_{x \in \mathcal{X}} \left[\boldsymbol{T}_0^{\text{prob}}(x) - \min_{c \neq 0} \boldsymbol{T}_c^{\text{prob}}(x) \right] \quad \text{s.t.} \quad \mathcal{C}(x). \tag{18}$$

If an x^* exists with objective value greater than zero, then $\boldsymbol{T}$ prefers class 0 over at least one other class. Also, all other $x' \neq x^*$ have a greater or equal objective value. Hence, class 0 is never the least likely class for any x. If no such x^* exists, then there must be at least one x' with class 0 the least likely class.

Our implementation supports all these variants.

Experiments

The experiments address the following two questions:

Q1. How often does the one-versus-other reduction fail to solve targeted evasion decision problem?

Q2. What is the run time performance of our proposed method?

Datasets and Methodology Our evaluation considers seven multiclass datasets: *img* (synthetic, see appendix), *wine* (Cortez et al. 2009), *bean* (Koklu and Ozkan 2020), *mnist* (LeCun, Cortes, and Burges 2010), *letter* (Slate 1991), *pendigits* (Alpaydin and Alimoglu 1998), and *sensorless* (Bator 2015). Table 1 shows the properties of the datasets. The predictive performance of the learned ensembles are provided in the appendix.

We use 4-fold cross validation: 3/4 for training, and 1/4 for testing. We train tree ensembles using XGBoost 1.7.6 (GBDT) and scikit-learn 1.2.2 (RF) on the training data using the hyper-parameters in Table 1. The experiments ran on an Intel(R) E3-1225 CPU with 32GiB of memory.

We randomly select 500 correctly classified base examples from the test set of each fold. For each base example x_b with true label c_b, we examine $C - 1$ targeted evasion problems, one for each target label $c_i \neq c_b$. If we find an adversarial example x such that $\boldsymbol{T}^{\text{label}}(x) = c_i$ and $\|x - x_b\|_\infty < \delta$, then we say that the targeted evasion decision problem is **satisfiable** (*sat*) for the *problem instance* x_b, c_i. If no such adversarial example can be found, then it is **unsatisfiable** (*unsat*). Across all seven datasets and all folds, we generate 136k problem instances (34k per fold, $500 \times (C-1)$ per dataset in a single fold).

We compare the following two approaches. For each problem instance with base example x_b and target class c_i:

1. The **one-versus-other** reduction solves $C - 1$ subproblems, one for each *other* class $c_j \neq c_i$. Each subproblem solves the objective in Equation 2 for the one-versus-other ensemble $\boldsymbol{T}_{c_i\text{-vs-}c_j}$. We use VERITAS.[4] If VERITAS finds an $x_{c_i\text{-vs-}c_j}$ for which the objective is greater than zero, then the subproblem is *sat*. If not, it is *unsat*.
2. The proposed **multiclass** approach solves the multiclass objective in Equation 5. If our proposed approach finds an x for which the objective is greater than zero, then we say the *multiclass result* is *sat*, else it is *unsat*.

The goal of the experiments is to show that the theorized problems with the one-versus-other reduction also occur in practice using benchmark datasets. We do this by counting how often disagreement occurs. Because verification is NP-hard, it is possible that a verification task times out before a solution is found. Timeouts would unnecessarily complicate this, so the hyper-parameters in Table 1 are chosen such that no timeouts occur. Nevertheless, because it is implemented in the VERITAS framework, our approach generates anytime upper and lower bounds on the objective value, so difficult problems can be solved approximately (see Devos, Meert, and Davis (2021b)).

Results Q1. In this first question, we count for how many problem instances x_b, c_i the one-versus-other reduction fails to solve the targeted evasion decision problem. We count how often the multiclass and one-versus-other results belong to the following four categories:

- **Agree** (*sat*): Both approaches agree on the satisfiability of the problem. This means that the multiclass result is *sat*, all one-versus-other subproblems are *sat*, and at least one of the generated $x_{c_i\text{-vs-}c_j}$ has $\boldsymbol{T}^{\text{label}}(x_{c_i\text{-vs-}c_j}) = c_i$.
- **Agree** (*unsat*): Both approaches agree on the unsatisfiability of the problem. This means that the multiclass result is *unsat* and that at least one one-versus-other subproblem is *unsat*.
- **Inconclusive** (*sat*): The multiclass result is *sat*. All one-versus-other subproblems are *sat*, but none of the $x_{c_i\text{-vs-}c_j}$ have $\boldsymbol{T}^{\text{label}}(x_{c_i\text{-vs-}c_j}) = c_i$, i.e., none of them are a solution to the targeted evasion problem. It is impossible to conclude from the one-versus-other subproblems what the outcome of the targeted evasion decision problem is.
- **Inconclusive** (*unsat*): Same as the item above, but the multiclass result is *unsat*.

[4] Note that other methods (e.g. Kantchelian, Tygar, and Joseph (2016)) would produce exactly the same results.

GBDT	Satisfiable		Unsatisfiable	
	Agree	*Inconcl.*	*Agree*	*Inconcl.*
img	.57±.02	.00±.00	.41±.02	.01±.00
wine	.37±.02	.06±.01	.55±.03	.02±.01
bean	.32±.02	.06±.01	.62±.02	.01±.00
mnist	.60±.02	.09±.01	.31±.02	.00±.00
letter	.41±.01	.14±.01	.28±.02	.16±.01
pendigits	.23±.01	.08±.01	.65±.02	.04±.01
sensorless	.62±.02	.10±.02	.25±.03	.03±.01

RF	Satisfiable		Unsatisfiable	
	Agree	*Inconcl.*	*Agree*	*Inconcl.*
img	.42±.04	.04±.02	.52±.04	.02±.01
wine	.29±.01	.03±.00	.62±.00	.06±.01
bean	.31±.03	.04±.01	.49±.04	.16±.02
mnist	.64±.03	.11±.00	.18±.02	.07±.00
letter	.36±.01	.05±.00	.38±.01	.21±.01
pendigits	.22±.01	.06±.01	.61±.03	.11±.01
sensorless	.68±.04	.07±.02	.14±.02	.11±.02

Table 2: Relative counts for each category. *Inconcl.* is short for inconclusive, and means that it is impossible to infer from the one-versus-other subproblems whether the targeted evasion decision problem is satisfiable or not. Averages over the results of four folds. The number of tests per fold is $500C$. Standard deviations are shown in gray.

Table 2 shows how often each category is observed. Inconclusive results occur for all datasets. For the simplest synthetically generated *img* dataset with only 2 features, 1.8% (GBDT) and 5.3% (RF) of the problem instances are inconclusive. For the other datasets, these numbers range from 7% to 30% for GBDT and from 9% to 27% for RF. On average, inconclusive results occur more frequently with RF than with GBDT, especially for unsatisfiable problem instances.

Results Q2. Table 3 shows the average time taken to produce results for a single problem instance. For the *multiclass* approach, this is a single number. For the one-versus-other reduction, this is the sum of the run times of the $C-1$ subproblems. For GBDT, *multiclass* is considerably slower than the one-versus-other reduction. However, the run time for the one-versus-other reduction is not meaningful as this method returns incorrect results. The large difference is explained by the fact that a single one-versus-other subproblem reasons about $\boldsymbol{T}_{c_i\text{-vs-}c_j}$, which consists of $2M$ trees, while *multiclass* reasons about the full ensemble of $C \times M$ trees. Given the fact that verification is NP-hard (Kantchelian, Tygar, and Joseph 2016) and that the number of possible outcomes of a tree ensemble is exponential in the number of trees in practice (Devos et al. 2023), this run time difference is not surprising.

This difference is not observed for RFs, where for some datasets, *multiclass* even outperforms *one-versus-other* in terms of run time (*wine*, *mnist*, *letter*, *sensorless*). RFs learn a single ensemble for all classes, so *multiclass* and a single one-versus-other subproblem reason about the same M trees (though with different leaf values, see Figure 1 and 2).

	GBDT		RF	
	multi	1v-other	multi	1v-other
img	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
wine	0.16±1.39	0.02±0.03	0.17±0.67	0.40±2.34
bean	0.77±5.64	0.00±0.01	0.48±3.18	0.13±1.29
mnist	0.84±6.95	0.00±0.00	1.09±6.27	1.13±12.5
letter	0.15±0.81	0.02±0.02	0.02±0.01	0.21±0.56
pendigits	0.22±3.09	0.00±0.00	0.03±0.31	0.03±0.14
sensorless	0.36±3.90	0.00±0.00	0.00±0.03	0.05±0.80

Table 3: Average time in seconds taken to solve the targeted evasion decision problem for one problem instance for *multiclass* (*multi*) and the one-versus-other reduction (*1v-other*). Standard deviations are shown in gray.

Related Work

Considerable attention has been devoted in recent years to verification of tree ensembles. Most of the described attacks and verification frameworks are only defined for binary classification (Devos, Meert, and Davis 2021b; Zhang, Zhang, and Hsieh 2020; Vos and Verwer 2021; Calzavara et al. 2020). Several works use the one-versus-other strategy to encode a multiclass problem in terms of multiple binary classification problems (Andriushchenko and Hein 2019; Chen et al. 2019b; Devos, Meert, and Davis 2021b). To our knowledge, no approaches exist that can simultaneously reason about more than two classes.

Popular lines of work for reasoning about tree ensembles include developing algorithms for performing evasion attacks (Kantchelian, Tygar, and Joseph 2016; Einziger et al. 2019), conducting robustness checks (Chen et al. 2019b), and verifying ensemble compliance with specific criteria (Devos, Meert, and Davis 2021a,b; Ranzato and Zanella 2020; Törnblom and Nadjm-Tehrani 2020).

Kantchelian et al. (2016) first showed that tree ensembles are vulnerable to evasion attacks, similarly to neural networks. Their MILP formulation remains the most widely used approach for both robustness checking and adversarial example generation. Alternative exact methods exploit the logical structure of decision trees to encode the ensemble in a set of logical formulas using Satisfiability Modulo Theories (SMT) (Einziger et al. 2019; Devos, Meert, and Davis 2021a). Then, verification is performed using existing SMT solvers such as Z3 (De Moura and Bjørner 2008). While the SMT formulation of ensembles is elegant, its run time tends to be (much) slower than using a MILP solver, making it less relevant in practice. Both MILP and SMT approaches are exact, meaning they find optimal solutions (e.g., the closest existing adversarial example).

Employing an exact approach can be challenging and time-consuming. Moreover, an approximate result is often sufficient; for example, checking whether an adversarial example closer than δ exists does not require finding the closest one. Therefore, several approximate methods have been designed specifically for tree ensembles. Chen et al. (2019b) introduced a representation based on K-partite graphs, where each max-clique corresponds to an output of the ensemble. Although this method enables fast approxi-

mate evaluation of robustness, it does not generate concrete adversarial examples. Building upon this work, Devos et al. (2021b) improved the graph search procedure, proposing a heuristic approach that allows effectively finding concrete adversarial examples while also improving upon the run time performance of Chen et al.'s method. Zhang et al. (2020) also introduced a greedy discrete search method optimized for fast adversarial example generation by repeatedly exploring the space of neighboring leaves.

Other works attempt to mitigate the effect of evasion attacks. One line looks at building tree-based models that are inherently less susceptible to adversarial attacks, rather than verifying the robustness of existing models. These approaches include adding generated adversarial examples to the training data for model hardening (Kantchelian, Tygar, and Joseph 2016), simplifying the base learner (Andriushchenko and Hein 2019), modifying the splitting procedure (Chen et al. 2019a; Calzavara et al. 2020; Vos and Verwer 2021), employing a robust 0/1 loss (Guo et al. 2022), relabeling and pruning leaves (Vos and Verwer 2022a), and using optimal decision trees to encode robustness constraints (Vos and Verwer 2022b). Another line of work tries to attempts to identify evasion attacks post deployment (Devos et al. 2023). In this setting, when an adversarial example is detected, the model abstains from making a prediction as in a learning to reject setting (Hendrickx et al. 2021).

Conclusion

This paper explored verification in the context of multiclass tree ensembles. Crucially, it proved that the current paradigm of reducing multiclass problems to a series of binary ones using a one-versus-other strategy can yield incorrect results. The missing link is the ability to reason about all classes simultaneously. We addressed this gap by proposing a generalization of the verification problem in tree ensembles to the multiclass setting and providing a novel multiclass verification approach. Our approach is based on heuristic search, and we proved the consistency of our heuristic. Empirically, we demonstrated the ability of our approach to solve multiclass verification tasks.

Appendix: Counter-Example 2 of Theorem 1

This section shows an alternative counter-example where, just like in the previous example, two one-versus-other solutions $x_{0\text{-vs-}1}$ and $x_{0\text{-vs-}2}$ exist and $\boldsymbol{T}$ predicts class 0 for neither of them. However, in this example, there still exists another x for which the predicted label is class 0. This example uses a slightly modified version of the ensemble in the main paper. Figure 3 highlights the changes in blue.

We use VERITAS to generate two examples $x_{0\text{-vs-}1} = (\text{AGE} = 32, \text{HEIGHT} = 175, \text{BMI} = 20)$ and $x_{0\text{-vs-}2} = (\text{AGE} = 32, \text{HEIGHT} = 201, \text{BMI} = 20)$ with respective outputs [15, 6, 18] and [16, 18, 8] so that $\boldsymbol{T}_0^{\text{raw}}(x_{0\text{-vs-}1}) > \boldsymbol{T}_1^{\text{raw}}(x_{0\text{-vs-}1})$ and $\boldsymbol{T}_0^{\text{raw}}(x_{0\text{-vs-}2}) > \boldsymbol{T}_2^{\text{raw}}(x_{0\text{-vs-}2})$. The numbers inside the square brackets are the class counts returned by the RF. The leaves visited by the two examples are indicated with a • (0-versus-1) and a ○ (0-versus-2) in Figure 3. Both examples are solutions to the respective one-versus-other subproblem ($15 - 6 > 0$ and $16 - 8 > 0$), yet neither is a solution to the targeted evasion problem because class 0 never has the highest weight. The full solution subset of the input space for the two subproblems is:

$$\begin{aligned}
&\boldsymbol{T}_{0\text{-vs-}1}^{\text{raw}}(x) > 0 \text{ iff } x \in \\
&\quad (\text{AGE} < 40 \wedge \text{HEIGHT} < 200) \cup \\
&\quad (\text{AGE} \geq 50 \wedge \text{HEIGHT} \geq 200 \wedge \text{BMI} < 28). \\
&\boldsymbol{T}_{0\text{-vs-}2}^{\text{raw}}(x) > 0 \text{ iff } x \in (\text{HEIGHT} \geq 200 \wedge \text{BMI} < 28).
\end{aligned}$$

For each example in each solution subset, the respective subproblem is satisfied. The intersection of these two solution subsets satisfies both subproblems *simultaneously*. If the intersection is non-empty, then any example in that subset is a solution to the targeted evasion problem:

$$(\text{AGE} \geq 50 \wedge \text{HEIGHT} \geq 200 \wedge \text{BMI} < 28)$$

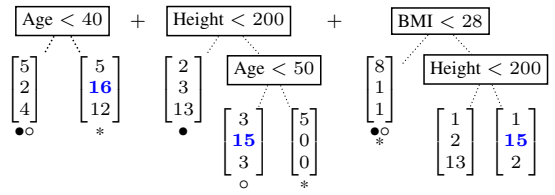
Take any example from this intersection and the multiclass label is class 0:

$$\begin{aligned}
x_0 &= (\text{AGE} = 51, \text{HEIGHT} = 204, \text{BMI} = 20), \\
\boldsymbol{T}^{\text{raw}}(x_0) &= [18, 17, 13], \\
\boldsymbol{T}^{\text{label}}(x_0) &= \arg\max([18, 17, 13]) = 0.
\end{aligned}$$

The leaves activated by this example are indicated with a * in Figure 3.

This example, together with the example in the main paper, illustrate that the one-versus-other subproblems cannot be used to reliably obtain a solution to the targeted evasion problem. That is, the existence of solutions to the $C-1$ subproblems cannot be used to infer the answer to the targeted evasion decision problem.

(a)



(b)

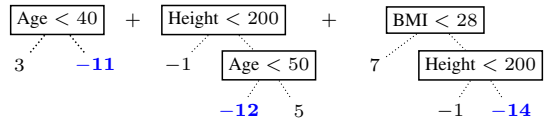


Figure 3: (a) A slightly modified version of the example RF in the main paper. The changes are indicated in bold and blue. (b) The of 0-versus-1 ensemble derived from (b), also with changed with respect to the example in the main paper indicated in bold and blue.

Acknowledgments

This research is supported by the Research Foundation-Flanders (FWO, LD: 1SB1322N), The European Union's

Horizon Europe Research and Innovation program under the grant agreement TUPLES No. 101070149 (LD, LC, JD), and the Flemish Government under the "Onderzoeksprogramma Artificële Intelligentie (AI) Vlaanderen" program (JD).

References

Alpaydin, E.; and Alimoglu, F. 1998. Pen-Based Recognition of Handwritten Digits. UCI Machine Learning Repository.

Andriushchenko, M.; and Hein, M. 2019. Provably robust boosted decision stumps and trees against adversarial attacks. In *Advances in Neural Information Processing Systems*, volume 32.

Bator, M. 2015. Dataset for Sensorless Drive Diagnosis. UCI Machine Learning Repository.

Breiman, L. 2001. Random forests. *Machine learning*, 45: 5–32.

Calzavara, S.; Lucchese, C.; Tolomei, G.; Abebe, S. A.; and Orlando, S. 2020. Treant: training evasion-aware decision trees. *Data Mining and Knowledge Discovery*, 34(5): 1390–1420.

Chen, H.; Zhang, H.; Boning, D.; and Hsieh, C.-J. 2019a. Robust decision trees against adversarial examples. In *International Conference on Machine Learning*, 1122–1131.

Chen, H.; Zhang, H.; Si, S.; Li, Y.; Boning, D.; and Hsieh, C.-J. 2019b. Robustness verification of tree-based models. *Advances in Neural Information Processing Systems*, 32.

Chen, T.; and Guestrin, C. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.

Cortez, P.; Cerdeira, A.; Almeida, F.; Matos, T.; and Reis, J. 2009. Wine Quality. UCI Machine Learning Repository.

Davis, J.; Devos, L.; Reyners, S.; and Schoutens, W. 2020. Gradient Boosting for Quantitative Finance. *Journal of Computational Finance*, 24(04).

De Moura, L.; and Bjørner, N. 2008. Z3: An Efficient SMT Solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 337–340.

Decroos, T.; Bransen, L.; Van Haaren, J.; and Davis, J. 2019. Actions Speak Louder than Goals: Valuing Player Actions in Soccer. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1851–1861.

Devos, L.; Meert, W.; and Davis, J. 2020. Fast gradient boosting decision trees with bit-level data structures. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, 590–606.

Devos, L.; Meert, W.; and Davis, J. 2021a. Verifying tree ensembles by reasoning about potential instances. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, 450–458.

Devos, L.; Meert, W.; and Davis, J. 2021b. Versatile Verification of Tree Ensembles. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 2654–2664.

Devos, L.; Perini, L.; Meert, W.; and Davis, J. 2023. Adversarial Example Detection in Deployed Tree Ensembles. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2023*.

Einziger, G.; Goldstein, M.; Sa'ar, Y.; and Segall, I. 2019. Verifying Robustness of Gradient Boosted Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33: 2446–2453.

Friedman, J. H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Ghiasi, M. M.; and Zendehboudi, S. 2021. Application of decision tree-based ensemble learning in the classification of breast cancer. *Computers in Biology and Medicine*, 128: 104089.

Guo, J.-Q.; Teng, M.-Z.; Gao, W.; and Zhou, Z.-H. 2022. Fast Provably Robust Decision Trees and Boosting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 8127–8144.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2): 100–107.

Hendrickx, K.; Perini, L.; Van der Plas, D.; Meert, W.; and Davis, J. 2021. Machine learning with a reject option: A survey. *arXiv preprint arXiv:2107.11277*.

Kantchelian, A.; Tygar, J. D.; and Joseph, A. 2016. Evasion and hardening of tree ensemble classifiers. In *International Conference on Machine Learning*, 2387–2396.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems*, volume 30.

Koklu, M.; and Ozkan, I. A. 2020. Dry Bean Dataset. UCI Machine Learning Repository.

LeCun, Y.; Cortes, C.; and Burges, C. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Pearl, J.; and Kim, J. H. 1982. Studies in Semi-Admissible Heuristics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(4): 392–399.

Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A. V.; and Gulin, A. 2018. CatBoost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, volume 31.

Ranzato, F.; and Zanella, M. 2020. Abstract interpretation of decision tree ensemble classifiers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 5478–5486.

Slate, D. 1991. Letter Recognition. UCI Machine Learning Repository.

Törnblom, J.; and Nadjm-Tehrani, S. 2020. Formal verification of input-output mappings of tree ensembles. *Science of Computer Programming*, 194: 102450.

Vos, D.; and Verwer, S. 2021. Efficient Training of Robust Decision Trees Against Adversarial Examples. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 10586–10595.

Vos, D.; and Verwer, S. 2022a. Adversarially Robust Decision Tree Relabeling. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*.

Vos, D.; and Verwer, S. 2022b. Robust optimal classification trees against adversarial examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 8520–8528.

Zhang, C.; Zhang, H.; and Hsieh, C.-J. 2020. An Efficient Adversarial Attack for Tree Ensembles. In *Advances in Neural Information Processing Systems*, volume 33, 16165–16176.