The Rashomon Set Has It All: Analyzing Trustworthiness of Trees under Multiplicity

Ethan Hsu[1] **Tony Cao**[1] **Lesia Semenova**[2*] **Chudi Zhong**[3]

[1] Duke University [2] Rutgers University [3] UNC Chapel Hill
{Ethan.Hsu, tony.cao}@duke.edu
lesia.semenova@rutgers.edu, chudi@unc.edu

Abstract

In practice, many models from a function class can fit a dataset almost equally well. This collection of near-optimal models is known as the Rashomon set. Prior work has shown that the Rashomon set offers flexibility in choosing models aligned with secondary objectives like interpretability or fairness. However, it is unclear how far this flexibility extends to different trustworthy criteria, especially given that most trustworthy machine learning systems today still rely on complex specialized optimization procedures. *Is the Rashomon set all you need for trustworthy model selection? Can simply searching the Rashomon set suffice to find models that are not only accurate but also fair, stable, robust, or private, without explicitly optimizing for these criteria?* In this paper, we introduce a framework [2] for systematically analyzing trustworthiness within Rashomon sets and conduct extensive experiments on high-stakes tabular datasets. We focus on sparse decision trees, where the Rashomon set can be fully enumerated. Across seven distinct metrics, we find that the Rashomon set almost always contains models that match or exceed the performance of state-of-the-art methods specifically designed to optimize individual trustworthiness criteria. These results suggest that for many practical applications, computing the Rashomon set once can serve as an efficient and effective method for identifying highly accurate and trustworthy models. Our framework can be a valuable tool for both benchmarking Rashomon sets of decision trees and studying the trustworthiness properties of interpretable models.

1 Introduction

With the increasing use of machine learning (ML) in high-stakes domains such as healthcare, lending, and criminal justice, the demand for models that satisfy multiple trustworthiness criteria, such as interpretability, robustness, fairness, privacy, and regulatory compliance, has grown substantially [56, 70, 87, 92]. However, achieving these properties in practice remains difficult, especially when they must be satisfied simultaneously. Most algorithms address only one aspect of trustworthiness at a time, typically by adding a new objective or constraint to the loss function. This often requires solving a specialized (usually non-convex) optimization problem which is tailored to that specific criterion, and is rarely transferable across objectives. As a result, developing trustworthy ML systems today often means solving a different optimization problem for each property, which is computationally expensive, resource-intensive, and can be infeasible in legally constrained environments.

Recent work challenges the assumption that separate optimization is always necessary. Building on the Rashomon Effect [10], which describes the existence of many models that can perform nearly as

[*]The work was done while at Microsoft Research NYC.
[2]https://github.com/EtHsu0/rashomon-framework

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Track on Datasets and Benchmarks.

well as the best one for a given dataset, researchers have proposed methods to construct and analyze the Rashomon set, the collection of near-optimal models [6, 29, 45, 46, 67, 79, 100, 103]. These methods make it possible to enumerate or approximate the Rashomon set across different hypothesis spaces, enabling new approaches to model selection that do not require additional retraining [75]. This raises a fundamental question: *Can the Rashomon set already contain models that satisfy trustworthiness goals, without the need for separate, objective-specific optimization?*

To answer the question, we introduce a framework for systematically evaluating trustworthiness within Rashomon sets. We focus on decision trees and their Rashomon sets since trees are interpretable, simple, and are well-suited for high-stakes decision-making problems [74]. We use TreeFARMS [100], which can enumerate all sparse trees within the epsilon loss of the optimal tree, to study whether this full set of near-optimal models inherently contains ones that naturally satisfy a wide range of trustworthiness properties. Our framework supports seven different trustworthy measures, including (1) adversarial robustness [50], (2) stability to data pertubations [49], (3) protection of privacy against membership attacks [80], (4) unlearning a small portion of data [8], and fairness metrics, such as (5) statistical parity [26], (6) equalized odds and (7) equal opportunity [41]). We systematically compare the sparse trees in the Rashomon set with trees optimized for specific criteria.

Our contributions include (1) introducing an open and extensible evaluation framework with standard datasets, baselines, trustworthy metrics and attacks, and evaluation protocols, enabling reproducible research on trustworthy model selection and interpretable model evaluation; (2) showing that the Rashomon set of sparse decision trees often contains models that match or exceed the performance of specialized models across multiple trustworthiness criteria, easing the need for specialized optimization per criterion; (3) showing that models optimized for one property (e.g., fairness) do not always generalize to others (e.g., robustness), motivating the explicit model selection within the Rashomon set instead of separate optimization.

Our findings suggest that enumerating near-optimal models, rather than retraining for each new objective, offers a practical and principled strategy for building responsible and trustworthy ML systems. By leveraging the natural diversity within the Rashomon set, practitioners can select models that align with application-specific constraints and learn trade-offs between trustworthiness criteria, thereby bridging the gap between theoretical insight and real-world deployment.

2 Related Works

We discuss the related works about the Rashomon Effect, decision trees, and trustworthy benchmarks and frameworks.

Rashomon Effect. The Rashomon set, a formal quantification of the Rashomon Effect, contains multiple different models that achieve approximately equal performance [29, 45, 67, 75, 79, 100]. Recent work in this area can be broadly categorized into those that focus on computing and characterizing the Rashomon Effect and the Rashomon set [45, 67, 93, 100, 103] and those that study the implications of large Rashomon set for different applications and trustworthy machine learning as a whole [6, 32, 94]. In this paper, we focus on TreeFARMS [100], which finds the Rashomon set of sparse decision trees. Many works in this domain that focus on understanding fairness and less discriminative hypothesis in the presence of a large Rashomon set are the closest to our work [7, 19, 21, 33, 55, 69]. However, none of the prior works consider multiple trustworthy criteria within one Rashomon set of interpretable models.

Decision trees. Decision trees are among the most popular methods in interpretable machine learning. Recent advances can find sparse optimal trees using either mathematical programming solvers [1, 4, 5, 27, 37, 85, 86] or dynamic programming with branch-and-bound [2, 22, 62, 68]. Recent research also incorporates other metrics, such as fairness [48, 84], robustness [12, 13, 14, 38, 49, 88, 89], and privacy [90], into the optimization problem, aiming to make sparse decision trees align with more trustworthy principles. Despite these advancements, there has been no systematic evaluation of these algorithms, nor has any study specifically examined if sparse decision trees that achieve high accuracy naturally exhibit trustworthy properties without being explicitly optimized for them.

Trustworthy Benchmarks and Frameworks. Trustworthiness in ML has emerged as a critical concern, especially as AI systems are increasingly deployed in high-stakes environments. While trustworthiness encompasses a broad spectrum of principles, metrics such as interpretability, robustness, privacy, and fairness consistently emerge as essential components [51, 58]. Benchmarks have been developed for robustness [20, 23, 42, 81], privacy [72, 83], and fairness [3, 40, 95]. Beyond individual trustworthiness benchmarks, some comprehensive trustworthiness benchmarks have been proposed [47, 76]. However, existing benchmarks majorly focus on deep learning models, leaving interpretable models, such as sparse decision trees, largely unexamined. Given their extensive use in healthcare, finance, and criminal justice, evaluating sparse decision trees under a rigorous trustworthiness framework is crucial.

In this work, we develop a framework for interpretable models, assessing robustness, privacy, and fairness while leveraging the Rashomon set as a unifying concept (see Section 3). Our framework enables researchers to explore whether models within the Rashomon set can naturally satisfy multiple trustworthiness criteria without sacrificing accuracy, providing a new perspective on the design and evaluation of trustworthy interpretable models. In Section 4, we provide empirical evidence that the Rashomon set often contains trustworthy models and analyze sparsity, timing, and cross-property behavior using our framework.

3 Background and Evaluation Framework

Our framework provides a systematic approach for evaluating the trustworthiness of models within the Rashomon set of sparse decision trees. It integrates five evaluation components, including robustness, stability to noise, membership inference, machine unlearning, and fairness. For each criterion, we define quantitative metrics, select state-of-the-art baseline algorithms that explicitly optimize for that property, and apply standardized datasets and evaluation protocols for fair comparison. We formally define the Rashomon set of sparse decision trees next and then focus on each trustworthy property.

3.1 The Rashomon set of sparse decision trees

Let $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ be the training dataset, where $\boldsymbol{x}_i \in \{0,1\}^p$ are binary features and $y_i \in \{0,1\}$ as labels. Let $\ell(t, \boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n}\sum_{i=1}^n 1[\hat{y}_i \neq y_i] + \lambda H_t$ be the loss of tree t on the training set, where $\hat{y}_i = t(\boldsymbol{x}_i)$, H_t is the number of leaves in tree t and λ is a regularization parameter. The loss function controls both the misclassification loss and the sparsity of the tree. Follow the definition in Semenova et al. [79], Xin et al. [100] defines the Rashomon set of sparse decision trees as follows: Let t_{ref} be a reference model from $\mathcal{T}$, where $\mathcal{T}$ is a set of binary decision trees. The ϵ-Rashomon set is a set of all trees $t \in \mathcal{T}$ with $\ell(t, \boldsymbol{x}, \boldsymbol{y})$ at most $\ell(t_{\text{ref}}, \boldsymbol{x}, \boldsymbol{y}) + \epsilon$: $R_{\text{set}}(\epsilon, t_{\text{ref}}, \mathcal{T}) := \{t \in \mathcal{T} : \ell(t, \boldsymbol{x}, \boldsymbol{y}) \leq \ell(t_{\text{ref}}, \boldsymbol{x}, \boldsymbol{y}) + \epsilon\}$.

Typically, the reference model is an empirical risk minimizer $t_{\text{ref}} \in \arg\min_{t \in \text{trees}} \ell(t, \boldsymbol{x}, \boldsymbol{y})$. Xin et al. [100] propose the TreeFARMS algorithm, the first method to construct the Rashomon set to find *all* good sparse decision trees. It uses mathematical bounds to prune infeasible spaces, dynamic programming for computation reuse, and the model set representation to extract and store the entire Rashomon set. TreeFARMS can find millions of good sparse trees within a short amount of time (within seconds or minutes, depending on the dataset size, see Section 4.4).

While TreeFARMS can enumerate all good sparse trees, it remains unclear whether models within the Rashomon set inherently satisfy trustworthiness principles. Also, in the presence of thousands or millions of near-optimal trees, it might not be clear which model to choose for deployment. Next, we investigate these questions by benchmarking sparse trees from the Rashomon set and other tree methods across multiple trustworthiness criteria.

3.2 Robustness

Robustness ensures models maintain performance under various conditions such as adversarial perturbations and data noise [31, 64]. Here, we focus on *adversarial robustness* and *stability* to random perturbations and investigate whether robust models are naturally contained in diverse Rashomon sets.

Adversarial robustness measures the ability of a machine learning model to correctly classify inputs that have been intentionally perturbed through white-box or black-box attacks [34]. Given that decision trees are inherently interpretable, meaning their structure is humanly understandable, we primarily consider white-box attacks (attacks with information about the model). Specifically, we consider evasion-style attacks, which aim to minimally perturb an input to cause misclassification. Given a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, and a tree $t \in \mathcal{T}$, Kantchelian et al. [50] propose an algorithm that generates adversarial examples $\boldsymbol{x}'_i$, such that the misclassification error of t is maximized on the dataset $\mathcal{D}' = \{(\boldsymbol{x}'_i, y_i)\}_{i=1}^n$. In other words, if $t(\boldsymbol{x}) = y$, the algorithm outputs a perturbed point $\boldsymbol{x}'$ that results in $t(\boldsymbol{x}') \neq y$. The perturbations are constrained such that $\|\boldsymbol{x}'_i - \boldsymbol{x}_i\|_\infty \leq \theta$, where $\theta \in \mathbb{R}^+$ specifies the strength of the attack. If no such $\boldsymbol{x}'_i$ exists under the constraint, the original input $\boldsymbol{x}_i$ remains unchanged. We create an evaluation set $\mathcal{D}^{\text{adv}} = \{(\boldsymbol{x}_i^{\text{adv}}, y_i)\}_{i=1}^n$, where for each $\boldsymbol{x}_i$, we take the nearest adversarial example $\boldsymbol{x}'_i$, and apply the distance based on θ: $\boldsymbol{x}_i^{\text{adv}} = \boldsymbol{x}_i + \theta \frac{\boldsymbol{x}_i - \boldsymbol{x}'_i}{|\boldsymbol{x}_i - \boldsymbol{x}'_i|}$.

Many prior works focus on improving the adversarial robustness of decision trees against this attack. Common approaches include those that globally optimize over the space of decision trees, such as ROCT-V [89]; those that greedily focus on local optimizations using adversarially modified impurity measures, such as GROOT [88]; and those that construct decision trees with theoretically provable robustness guarantees, such as FPRDT [38]. We select the most recent methods from each category as baselines: ROCT-V, GROOT, and FPRDT. We also include the greedy method CART [9].

ROCT-V [89] finds optimal robust decision trees. It frames robust tree learning as a min-max problem over the 0-1 loss and solves it using mixed-integer programming (MIP).

FPRDT [38] is a greedy recursive approach for constructing robust decision trees. It directly minimizes the adversarial 0-1 loss by making a tradeoff between global and local optimizations over the potential splitting features and thresholds. FPRDT has a computational complexity of $O(n \log n)$, which is the smallest among all provably robust decision trees.

GROOT [88] makes greedy splits according to the adversarial Gini impurity – a splitting criterion that measures the worst-case Gini impurity after an attacker has maximally worsened the split by moving points within a specified perturbation range. Since impurity is concave to the number of modified data points, GROOT uses its analytical solution to compute the function in constant time.

Stability in our context refers to a model's ability to maintain accurate predictions under natural perturbations of the input data. We follow the approach from Justin et al. [49] to evaluate this property. First, for every feature with index $j \in \{0, \dots p\}$, a "confidence level" q_t^j is sampled from a normal distribution: $q_t^j \sim \mathcal{N}(\rho, \sigma)$, where ρ and σ are the normal distribution parameters. The value q^j represents the likelihood that the feature j remains unperturbed. If $q_t^j = 1$, then no perturbation occurs. Next, the noise is sampled as $\xi_i^j \sim S_i^j \cdot (G_i^j - 1),$ where $G_i^j \sim \text{Geom}(q_t^j), S_i^j = 2 \cdot B_i^j - 1$ with $B_i^j \sim \text{Bernoulli}(0.5)$.

Intuitively, G_i^j represents the strength of the noise for feature j of sample i, and S_i^j determines the sign, ensuring equal probability of positive and negative noise. Because features and splits are integer-valued, this symmetric geometric step targets flips/threshold crossing: any nonzero step flips a binary feature, and a split at threshold θ is crossed iff $|\xi_i^j| \geq d(x_i^j, \theta)$ (the integer distance to θ). The new dataset is then $\mathcal{D}^{\text{stab}} = \{(\boldsymbol{x}_i^{\text{stab}}, y_i)\}_{i=1}^n$, where $x_i^{\text{stab},j} = x_i^j + \xi_i^j$.

We use **ROCT-N** [49] as the stability baseline. It finds a globally optimal robust tree using a two-stage robust optimization approach. The first stage determines the tree structure to maximize correctly classified training samples under worst-case perturbation, where confidence levels q_t^j define the uncertainty set for each feature j. The second stage optimizes the classification of training samples after observing the worst-case perturbation. This problem is formulated as a mixed-integer program and solved using a customized Benders decomposition algorithm.

For both robustness and stability, we evaluate CART, FPRDT, GROOT, ROCT-N, ROCT-V, and TreeFARMS across 13 datasets (Table 5) with standard five-fold nested cross-validation. The configuration for each method is in Appendix B.

3.3 Privacy

Protecting privacy is essential for machine learning systems that handle sensitive or personally identifiable data. Among many privacy threats studied in the literature, we focus on membership inference attacks and machine unlearning.

3.3.1 Membership Inference Attack

Membership inference attacks (MIAs) aim to determine whether a particular data point was used to train a given model [80]. We evaluate whether sparse decision trees within the Rashomon set are inherently resistant to MIAs compared to explicitly private trees. To comprehensively evaluate privacy, our setup includes both defense mechanisms and attack mechanisms.

Existing defenses against MIAs fall into two broad categories: theoretical guarantees such as differential privacy [25], and empirical defenses that aim to reduce overfitting or confidence leakage. In this work, we focus on the former and compare TreeFARMS to representative differentially private decision-tree algorithms. Differential privacy offers formal protection against MIAs by injecting randomness during training. Formally, an algorithm $\mathcal{M}$ is (η_1, η_2)-differentially private if for all dataset $\mathcal{D}, \mathcal{D}'$ that differ on a single element, and for any $S \subseteq \text{range}(M)$, $P[\mathcal{M}(\mathcal{D}) \in S] \leq e^{\eta_1} P[\mathcal{M}(\mathcal{D}') \in S] + \eta_2$, where $\eta_1, \eta_2 \geq 0$. The basic idea of differential privacy is to ensure that individual data points cannot be identified while still preserving overall data utility. We include three representative DP tree algorithms as baselines: PRIVA [90], BDPT [36], and DPLDT from DiffPrivLib [43]. These greedy algorithms introduce randomness into split selection and leaf labeling to ensure privacy, though often at the cost of predictive performance.

PRIVA [90] first determines quantile-based bins for numerical features and then uses private histograms to select good splits with minimal privacy budget. It partitions the data recursively until it reaches leaf nodes, which are then labeled using a noise-based majority vote mechanism.

BDPT [36] builds the tree top-down. The best splitting attribute is chosen based on the Gini index via the exponential mechanism, and continuous attributes are discretized using down-sampling. At each leaf node, noisy class counts are calculated (using Laplace noise) to determine the final label.

DPLDT [43] implements the randomized split tree from [30]. Each tree is built by randomly selecting a feature at each node and partitioning the data accordingly. The leaf nodes use the Exponential Mechanism (with smooth sensitivity) to output only the majority label to maintain differential privacy.

To evaluate the privacy of both TreeFARMS and DP tree models described above, we adopt four representative MIA algorithms from the literature, varying in the level of adversarial knowledge and access to the model, ranging from simple prediction-based heuristics to shadow-model training.

- The **baseline attack** [102] infers membership based on the correctness of the model's predictions: a sample is considered a member if the model predicts it correctly. This attack serves as a reference point for other methods, relying purely on the extent of model overfitting.
- The **label-only attack** [59] requires only hard class predictions. It infers the membership by estimating the minimum perturbation needed to flip the predicted label. If this distance exceeds a threshold, the sample is inferred to be a member. We calibrate a membership threshold at the 50th percentile of distance scores computed on a pool of unlabeled points.
- The **label-only supervised attack** [16] is a stronger variant of label-only attack, which leverages partial knowledge of the data distribution to calibrate the threshold. In experiments, the attacker is given 500 reference samples with known membership status to determine the threshold.
- The **shadow model attack** [80] trains auxiliary models to mimic the target model and uses a separate classifier to predict the membership.

We evaluate on a balanced dataset of 1000 samples, where 500 samples are randomly sampled from the training set (members) and 500 from the test set (non-members). For two label-only attacks, we allow up to five perturbation queries per sample. Detailed configurations are in Appendix C.1.

3.3.2 Machine Unlearning

Machine unlearning refers to the process of removing the influence of specific training data from a trained model, ensuring that the model behaves as if those data points had never been used [8, 73]. It has become an important topic, with extensive work in deep learning and ensemble models [11, 78, 97], but little analysis has been conducted on sparse trees. To address this gap, besides TreeFARMS, we consider two unlearning algorithms in our framework: data removal-enabled (DaRE) forests [11] and GBDT unlearning [61].

DaRE [11] is an unlearning algorithm for random forests that leverages randomness and caching to enable efficient unlearning. The trees in the forest can follow a greedy top-down approach, referred to as G-DaRE, or incorporate random layers in the top, where splits are chosen uniformly at random, referred to as R-DaRE.

GBDT unlearning [61] provides an algorithm for gradient boosted trees that uses intermediate data statistics to decide if subtrees need retraining. It uses random split point selection to limit split values and optionally adds random layers to restrict retraining to a subset of subtrees. The standard version without random layers is called G-Boosting, while the version with random layers is R-Boosting.

Unlearning in TreeFARMS requires *no* retraining. Instead, we can directly search within the Rashomon set to find an optimal tree after removing data points. Theorem 5.3 in [100] states that if $\epsilon \geq \frac{2K}{n}$, where K is the number of removed points and n is the original data size, the optimal tree after removal remains in the Rashomon set trained on the full dataset. Thus, with a properly chosen ϵ, the optimal tree can always be found in the Rashomon set, regardless of which or how many (up to K) samples are removed.

We evaluate these methods on 5 datasets. Configurations for each method are in Appendix C.2.

3.4 Fairness

We consider three group fairness metrics: statistical parity, equal opportunity, and equalized odds. Let $Y, \hat{Y}$, A, and $\mathcal{A}$ be random variables for labels, predicted labels, sensitive features, and a set of possible values for A, respectively.

Statistical Parity (sp) ensures that the probability of a positive outcome is the same across all protected groups [26], i.e., $P(\hat{Y} = 1|A = a) = P(\hat{Y} = 1|A = b), \forall a, b \in \mathcal{A}$.

Equal Opportunity (eopp) requires a model to have equal true positive rates (TPR) across different groups, i.e., $P(\hat{Y} = 1|Y = 1, A = a) = P(\hat{Y} = 1|Y = 1, A = b), \forall a, b \in \mathcal{A}$.

Equalized Odds (eodds) requires that the model has equal true positive rates (TPR) and false positive rates (FPR) across all groups. Mathematically, it's defined as: $P(\hat{Y} = 1|A = a, Y = y) = P(\hat{Y} = 1|A = b, Y = y), \forall a, b \in \mathcal{A}, y \in \mathcal{Y}$.

We evaluate the fairness performance of trees within the Rashomon set and trees from optimal fair tree algorithms, DPF [84] and FairOCT [48]. Both are in-processing methods that incorporate fairness constraints directly into the optimization process. We also include the greedy method CART [9] and a post-processing method LinearPost [98, 99] as baselines.

DPF [84] finds an optimal tree with a given depth that minimizes misclassification loss with a statistical parity constraint. This constraint ensures that the difference in positive rates does not exceed a predefined threshold δ. The optimization problem is solved using dynamic programming, and a custom bound is used to prune partial solutions that cannot lead to the optimal fair tree.

FairOCT [48] formulates the optimization problem as a MIP problem, which a mathematical solver then solves. This algorithm can incorporate fairness constraints using the above-mentioned metrics, ensuring that the absolute difference remains within a predefined threshold δ.

LinearPost [98] is a post-processing method to achieve fairness by linearly transforming the predictions of the base classifier. A tolerance α controls the tradeoff between accuracy and fairness. LinearPost is model-agnostic, and we use CART and XGBoost [15] as base predictors, denoted as PostCART and PostXGB.

We consider binary classification with binary sensitive features. The sensitive feature is not used in training but is used for evaluation. Detailed setup is available in Appendix D. Note that DPF is only suitable for statistical parity, whereas other methods can be applied to all three fairness metrics.

4 Experimental Results

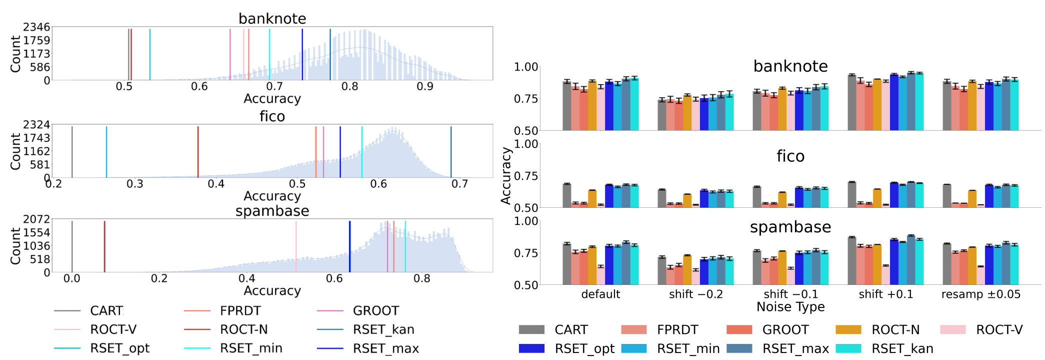
Our evaluation aims to answer the following questions: **Q1**. How do models in the Rashomon set compare to baselines optimized for specific trustworthiness criteria, such as robustness, privacy, and fairness (Section 4.1)? **Q2**. What proportion of models in the Rashomon set outperforms baseline models on key metrics, and how consistently does this occur across datasets (Section 4.2)? **Q3**. What connections, if any, exist between model sparsity and trustworthy properties, and how do these insights inform model selection (Section 4.3)? **Q4**. How does the computational cost of computing the Rashomon set compare to training separate models optimized for individual criteria (Section 4.4)? **Q5**. Are there observable interactions between different trustworthiness criteria (e.g., do fair models also tend to be robust) (Section 4.5)?

To address these questions, we conducted a comprehensive empirical analysis of sparse decision trees in the Rashomon set and compared them to baselines using our evaluation frameworks described in Section 3. We use a total of 21 datasets (see Table 5). We also provide results for other hypothesis spaces in Appendix E.

4.1 Q1. The Rashomon Set of Sparse Decision Trees Contains Trustworthy Models

Adversarial Robustness. Figure 1a compares the adversarial accuracy of trees within the Rashomon set to various baseline models on the test set. The light blue density plot represents the distribution of adversarial accuracy for all trees in the Rashomon set. Vertical lines indicate the accuracy of baseline models, including CART, FPRDT, GROOT, and ROCT, as well as some trees from the Rashomon set, such as the optimal tree, the tree with the fewest leaves, the tree with the most leaves, and RSET_kan, a tree chosen from the held-out selection set. Many trees within the Rashomon set achieve higher accuracy than the baselines, indicating their robustness against adversarial attacks. Also, the selected tree, RSET_kan, can perform comparably to or even better than baselines on the test set.

Stability. Figure 1b compares the stability of tree models when random perturbations are added to the data. Note that only ROCT-N explicitly accounts for these perturbations during training, while all other methods are trained normally or for robustness and evaluated for stability on perturbed test data. As we can see, different representative trees from the Rashomon set (bars colored in blue palette) and CART generally perform better than other methods. Although ROCT-N is designed to optimize for robustness against perturbations, its performance on the test set does not dominate other methods. This suggests that while the Rashomon set is not explicitly designed for stability, it contains trees that perform well under perturbations. More results are in Appendix B.



(a) (b)

Figure 1: (a) Comparison of adversarial accuracy between trees in the Rashomon set (light blue histogram) and baseline models (vertical lines) on the test set. Most Rashomon set trees achieve higher robustness than baselines. (b) Stability comparison of different methods under random perturbations. Trees in the Rashomon set (colored in a blue palette) generally perform better than baselines.

Membership Inference Attacks (MIAs). Table 1 shows the attack success rate when two different MIAs are applied to various methods. A lower value indicates that the method is more resistant to attacks. Overall, these attacks are not successful as the highest observed accuracy remains close to 0.5, meaning the attacker does not significantly outperform random guessing. Compared to 3 differential private tree methods, trees from the Rashomon set with the fewest leaves (RSET_min) achieve comparable or even better resistance against both attacks, despite the fact that the TreeFARMS algorithm does not incorporate any explicit randomness for privacy protection. Results for more attacks are in Appendix C.1.

Machine Unlearning. Table 2 shows the proportion of test data with different predicted labels between the unlearned and retrained models for the Rashomon set, G-DaRE, and G-Boosting after

Table 1: Membership inference attack success rates for baselines. Lower values indicate better resistance to attacks.

	Data	BDPT	DPLDT	PRIVA	RSET_min
Label-only	adult	**Attk Failed**	50.1% ± 1.6%	50.2% ± 1.6%	50.1% ± 1 %
	bank	**Attk Failed**	50.9% ± 0.8%	50.5% ± 1.0%	**Attk Failed**
	fico	49.7% ± 0.4%	49.2% ± 0.6%	50.9% ± 0.6%	**48.5% ± 1.8%**
	mimic	50.2% ± 0.1%	50.3% ± 0.9%	49.8% ± 0.9%	**Attk Failed**
	oulad	**Attk Failed**	50.1% ± 0.8%	50.0% ± 1.1%	**Attk Failed**
Shadow	adult	**49.1% ± 1.2%**	49.3% ± 1.2%	49.2% ± 1.1%	50.0% ± 1.0%
	bank	50.0% ± 0.8%	49.9% ± 1.0%	**49.8% ± 1.1%**	50.6% ± 1.1%
	fico	49.7% ± 0.5%	50.2% ± 0.4%	50.5% ± 0.7%	**49.4% ± 1.3%**
	mimic	49.8% ± 1.0%	49.8% ± 1.1%	**49.6% ± 1.0%**	49.8% ± 1.2%
	oulad	50.4% ± 0.8%	50.4% ± 0.8%	50.4% ± 0.8%	**49.9% ± 1.4%**

Table 2: Proportion of test data with different predicted labels between the unlearned and retrained models (i.e., $\hat{y}_{\text{unlearn}} \neq \hat{y}_{\text{retrain}}$) after 1% of the training data are randomly removed.

Data	RSET	G-DaRE	G-Boosting
carryout	0	1.294% ± 0.465%	5.548% ± 0.741%
restaurant	0	1.902% ± 0.433%	4.746% ± 0.623%
adult	0	0.018% ± 0.013%	0.064% ± 0.027%
compas	0	0.000% ± 0.000%	0.478% ± 0.543%
bank	0	0.607% ± 0.103%	0.888% ± 0.096%

1% of the original training data are randomly removed. Since the optimal tree after data deletion is guaranteed to be within the Rashomon set if ϵ is set appropriately, the RSET column always reports 0. In contrast, such a guarantee does not hold for other methods. More results are in Appendix C.2.

Fairness. Figure 2 compares the test accuracy and fairness of trees within the Rashomon set to baseline models at depth 4, with each row representing a different fairness metric. A higher value on both axes (top-right corner) indicates better performance. The blue density contours represent the distribution of trees in the Rashomon set. The count of trees at each grid is averaged over five folds. Baseline models and a selected Rashomon set tree are shown as dots with error bars, where the error bars indicate standard deviation. The results show that the contours usually overlap with the baseline points and cover the region close to the top-right corner, indicating that the Rashomon set contains trees with performance comparable to the baselines. Since DPF is specifically designed to optimize for statistical parity, it appears only in the top row of the figure. Appendix D displays results on 7 datasets at more depths.

4.2 Q2. Many Near-Optimal Trees are Trustworthy

Figure 1a and 2 have shown that many trees in the Rashomon set have comparable performance in adversarial robustness and fairness to baseline models. Table 3 reports the percentage of trees within the Rashomon set that outperforms baselines. In certain datasets, more than 50% of trees in the Rashomon set achieve higher test adversarial accuracy than FPRDT and ROCT-V. Similarly, the Rashomon set contains trees that have greater fairness than all baselines, except for the adult and compas, where DPF achieves lower disparity in statistical parity at the cost of reduced accuracy.

4.3 Q3. The Importance of Tree Sparsity for Trustworthy Properties

Larger Rashomon sets can contain models with different complexity [79, 100]. In the case of sparse decision trees, we can measure complexity by the number of leaves, and indeed our computed Rashomon sets often contain trees with different numbers of leaves (for example, the Rashomon set of the German Credit dataset includes trees with leaves in the range from 1 to 7, Appendix D.4). Sparsity is important for trustworthy metrics, such as robustness or privacy, as it reduces the amount of information encoded in a model, limiting the risk of revealing sensitive data [60]. Our results also support this. Sparser Rashomon set trees (RSET_min) tend to perform better than more complex counterparts (RSET_max) for adversarial robustness (Figure 1a and Appendix B.1) and under the

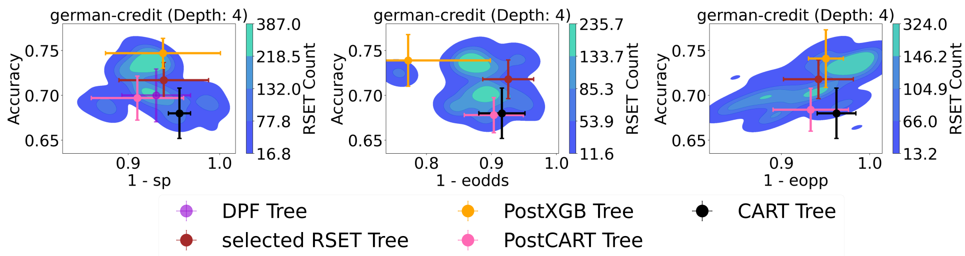


Figure 2: Test accuracy vs. fairness for trees in the Rashomon set (blue density contours) and baselines (dots) at depth 4. Higher values on both axes indicate better performance.

Table 3: Percentage of models in Rashomon set that perform better than baselines for adversarial robustness (left) and outperform *all* baselines for fairness (right) on the test set.

Robustness	FPRDT	ROCT-V	Fairness	SP	EODDS
banknote	94.66 ± 4.48%	95.34 ± 3.43%	adult	0.00% ± 0.00%	15.03% ± 15.99%
blood	4.91 ± 4.79%	5.26 ± 6.64%	bank	65.28% ± 21.96%	39.49% ± 16.71%
breast	56.09 ± 11.98%	75.58 ± 8.36%	german-credit	28.85% ± 39.49%	31.55% ± 36.20%
compas	23.96 ± 10.73%	28.15 ± 8.61%	compas	0.00% ± 0.00%	26.47% ± 35.93%
diabetes	46.59 ± 40.54%	62.34 ± 17.76%	oulad	84.22% ± 8.37%	89.21% ± 6.75%
spambase	44.37 ± 4.00%	78.93 ± 11.49%	student-mat	40.00% ± 48.99%	32.89% ± 33.12%

membership inference attack (Appendix C.1). Notably, we did not observe a connection between the sparsity of our models and fairness (Appendix D.4) and stability (Figure 1b and Appendix B.2).

4.4 Q4. TreeFARMS Training Time is Comparable to Optimal Baselines

TreeFARMS effectively optimizes the search space of trees, allowing us to find thousands or millions of near-optimal trees in seconds or minutes. Table 4 (and Appendices B, D.1) supports this with a summary of the training time compared to the robustness and fairness baselines that optimize for the best model averaged over datasets. Although TreeFARMS might require more time to run for deep depth, it finds significantly more trees (Appendix D.1).

Once the Rashomon set is constructed, model selection becomes a lightweight post-hoc step. Rather than retraining from scratch for each trustworthiness criterion, as required by optimization-based methods (e.g., robust or fair optimal trees), TreeFARMS allows users to evaluate and filter precomputed models according to desired constraints. Evaluating the entire set scales linearly with its size, is trivially parallelizable, and remains far cheaper than repeated retraining. In practice, enumerating and screening Rashomon sets with TreeFARMS is typically the more efficient and flexible option within an ML workflow.

4.5 Q5. Trees in the Rashomon Set Often Satisfy Multiple Trustworthy Properties

For different trustworthy metrics in Section 4.1, we selected trees that performed well based on the selection set. Here, in Figure 3, we further compare how these trees perform on other metrics. First of all, the accuracy of the trees is approximately the same since they all are from the same Rashomon set. All selected trees tend to be stable and perform well under membership inference attack, while the minimum complexity tree (RSET_min) is still preferred sometimes (e.g. bank). Similarly, trees selected specifically for statistical parity (e.g., german credit and bank datasets) seem to be preferable to other models, especially optimal trees. Overall our results highlight the importance of considering multiple trustworthy metrics when selecting models, as no single tree consistently outperforms others across all criteria. Please see Appendix F for more datasets.

5 Conclusion, Limitations and Implications

We introduced a framework for evaluating trustworthy properties of models inside the Rashomon set of sparse decision trees. By benchmarking the Rashomon set against state-of-the-art tree baselines

Table 4: Mean training time, in seconds, for TreeFarms and baselines averaged over thirteen robustness datasets (left) and seven fairness datasets (right) for depth 2.

	ROCT-N	ROCT-V	RSET	DPF	FairOCT	RSET
Mean Fit Time	1725.23	1644.30	252.48	0.06	9099.23	0.46

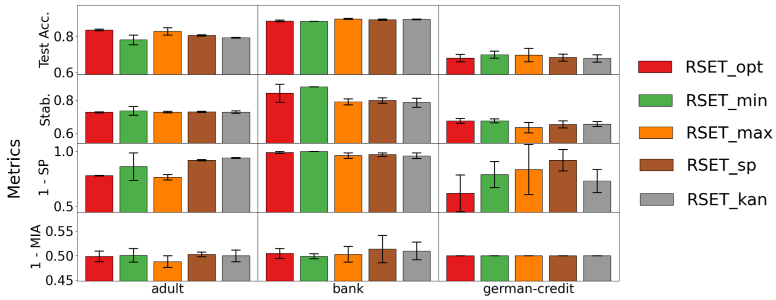


Figure 3: Evaluation of trees selected from the Rashomon set in Section 4.1 on different metrics.

targeted to individual trustworthiness criteria and analyzing fairness, stability, robustness, and privacy, we provide a systematic methodology for understanding and intentionally navigating trade-offs in high-stakes settings. Empirically, we find that Rashomon sets often contain models that are robust, stable, privacy-preserving, and fair even without explicit optimization for these properties. This reframes the Rashomon set as a resource: rather than retraining for every criterion, one can search within the set to identify models that meet desired constraints.

Our results suggest a simple selection protocol that mirrors the experiments: (i) enumerate or approximate the Rashomon set for a target loss tolerance and model class; (ii) evaluate trustworthiness metrics (fairness, stability, robustness, membership-inference privacy, and/or unlearning) for each model; (iii) filter by hard constraints (e.g., max disparity, min stability); (iv) choose model on the empirical Pareto frontier (e.g., fairness-accuracy or privacy-simplicity), or aggregate via a weighted objective when priorities are known. This procedure produces models that satisfy multiple criteria without retraining and exposes transparent trade-offs when criteria conflict.

One limitation of our approach is that we focus on the hypothesis space of decision trees, though our methodology can be extended to other model classes (see Appendix E). Another limitation is that we evaluate membership inference using a limited number of membership inference attack algorithms for trees, as few such methods have been proposed in the literature. Nonetheless, the diversity of near-optimal trees within the Rashomon set already offers benefits: it can be leveraged for robustness via moving-target defenses, for fairness by covering different subpopulations, and for exploring explicit Pareto trade-offs between properties such as privacy and simplicity.

Our findings motivate concrete questions for future work: When does model diversity most improve trustworthiness? Under what data/model conditions do multiple properties co-occur? How should we quantify diversity to predict trust gains? And to what extent do these effects transfer to richer model families? We hope that our evaluation frameworks can be used by data and machine learning scientists, as well as policymakers, to assess model performance across multiple trustworthiness criteria in a systematic and scalable way as well as inspire further research on model selection under larger Rashomon sets.

References

[1] Sina Aghaei, Andrés Gómez, and Phebe Vayanos. Strong optimal classification trees. *Operations Research*, 2024.

[2] Gaël Aglin, Siegfried Nijssen, and Pierre Schaus. Learning optimal decision trees using caching branch-and-bound search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3146–3153, 2020.

[3] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. AI Fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias, October 2018. URL `https://arxiv.org/abs/1810.01943`.

[4] Kristin P Bennett and Jennifer A Blue. Optimal decision trees. *Rensselaer Polytechnic Institute Math Report*, 214:24, 1996.

[5] Dimitris Bertsimas and Jack Dunn. Optimal classification trees. *Machine Learning*, 106(7): 1039–1082, 2017.

[6] Emily Black, Manish Raghavan, and Solon Barocas. Model multiplicity: Opportunities, concerns, and solutions. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 850–863, 2022.

[7] Emily Black, John Logan Koepke, Pauline T Kim, Solon Barocas, and Mingwei Hsu. Less discriminatory algorithms. *Geo. LJ*, 113:53, 2024.

[8] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pages 141–159. IEEE, 2021.

[9] L Breiman, JH Friedman, R Olshen, and CJ Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[10] Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199–231, 2001.

[11] Jonathan Brophy and Daniel Lowd. Machine unlearning for random forests. In *International Conference on Machine Learning*, pages 1092–1104. PMLR, 2021.

[12] Stefano Calzavara, Claudio Lucchese, Gabriele Tolomei, Seyum Assefa Abebe, and Salvatore Orlando. Treant: Training evasion-aware decision trees. *Data Mining and Knowledge Discovery*, 34(5):1390–1420, September 2020. ISSN 1384-5810, 1573-756X. doi: 10.1007/s10618-020-00694-9.

[13] Hongge Chen, Huan Zhang, Duane Boning, and Cho-Jui Hsieh. Robust decision trees against adversarial examples. In *International Conference on Machine Learning*, pages 1122–1131. PMLR, 2019.

[14] Hongge Chen, Huan Zhang, Si Si, Yang Li, Duane Boning, and Cho-Jui Hsieh. Robustness verification of tree-based models. *Advances in Neural Information Processing Systems*, 32, 2019.

[15] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[16] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International Conference on Machine Learning*, pages 1964–1974. PMLR, 2021.

[17] Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance. 2008.

[18] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision support systems*, 47(4):547–553, 2009.

[19] Amanda Coston, Ashesh Rambachan, and Alexandra Chouldechova. Characterizing fairness over the set of good models under selective labels. In *International Conference on Machine Learning*, pages 2144–2155. PMLR, 2021.

[20] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[21] Gordon Dai, Pavan Ravishankar, Rachel Yuan, Daniel B Neill, and Emily Black. Be intentional about fairness!: Fairness, size, and multiplicity in the rashomon set. *arXiv preprint arXiv:2501.15634*, 2025.

[22] Emir Demirović, Anna Lukina, Emmanuel Hebrard, Jeffrey Chan, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao, and Peter J Stuckey. Murtree: optimal classification trees via dynamic programming and search. *Journal of Machine Learning Research*, 23(26):1–47, 2022.

[23] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 321–331, 2020.

[24] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

[25] Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer, 2006.

[26] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[27] Alireza Farhangfar, Russell Greiner, and Martin Zinkevich. A fast way to produce near-optimal fixed-depth decision trees. In *Proceedings of the 10th international symposium on artificial intelligence and mathematics (ISAIM-2008)*, 2008.

[28] FICO, Google, Imperial College London, MIT, University of Oxford, UC Irvine, and UC Berkeley. Explainable Machine Learning Challenge, 2018.

[29] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. 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.

[30] Sam Fletcher and Md Zahidul Islam. Differentially private random decision forests using smooth sensitivity. *Expert systems with applications*, 78:16–31, 2017.

[31] Luciano Floridi. Establishing the rules for building trustworthy ai. *Ethics, Governance, and Policies in Artificial Intelligence*, pages 41–45, 2021.

[32] Prakhar Ganesh, Afaf Taik, and Golnoosh Farnadi. The curious case of arbitrariness in machine learning. *arXiv preprint arXiv:2501.14959*, 2025.

[33] Talia B Gillis, Vitaly Meursault, and Berk Ustun. Operationalizing the search for less discriminatory alternatives in fair lending. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 377–387, 2024.

[34] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[35] R Paul Gorman and Terrence J Sejnowski. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural networks*, 1(1):75–89, 1988.

[36] Zhitao Guan, Xianwen Sun, Lingyun Shi, Longfei Wu, and Xiaojiang Du. A differentially private greedy decision forest classification algorithm with high utility. *Computers & Security*, 96:101930, 2020.

[37] Oktay Günlük, Jayant Kalagnanam, Minhan Li, Matt Menickelly, and Katya Scheinberg. Optimal decision trees for categorical data via integer programming. *Journal of Global Optimization*, pages 1–28, 2021.

[38] Jun-Qi Guo, Ming-Zhuo Teng, Wei Gao, and Zhi-Hua Zhou. Fast provably robust decision trees and boosting. In *International Conference on Machine Learning*, pages 8127–8144. PMLR, 2022.

[39] Shelby J Haberman. Generalized residuals for log-linear models. In *Proceedings of the 9th International Biometrics Conference*, pages 104–122, 1976.

[40] Xiaotian Han, Jianfeng Chi, Yu Chen, Qifan Wang, Han Zhao, Na Zou, and Xia Hu. Ffb: A fair fairness benchmark for in-processing group fairness methods. In *The Twelfth International Conference on Learning Representations*, 2024.

[41] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

[42] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

[43] Naoise Holohan, Stefano Braghin, Pól Mac Aonghusa, and Killian Levacher. Diffprivlib: The IBM differential privacy library. *ArXiv e-prints*, 1907.02444 [cs.CR], July 2019.

[44] Mark Hopkins, Erik Reeber, George Forman, and Jaap Suermondt. Spambase data set. *Hewlett-Packard Labs*, 1(7), 1999.

[45] Hsiang Hsu and Flavio Calmon. Rashomon capacity: A metric for predictive multiplicity in classification. In *Neural Information Processing Systems (NeurIPS)*, volume 35, pages 28988–29000, 2022.

[46] Hsiang Hsu, Guihong Li, Shaohan Hu, et al. Dropout-based rashomon set exploration for efficient predictive multiplicity estimation. *arXiv preprint arXiv:2402.00728*, 2024.

[47] Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, et al. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*, 2024.

[48] Nathanael Jo, Sina Aghaei, Jack Benson, Andres Gomez, and Phebe Vayanos. Learning optimal fair decision trees: Trade-offs between interpretability, fairness, and accuracy. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*, pages 181–192, 2023.

[49] Nathan Justin, Sina Aghaei, Andres Gomez, and Phebe Vayanos. Optimal robust classification trees. In *The AAAI-22 Workshop on Adversarial Machine Learning and Beyond*, 2022.

[50] Alex Kantchelian, J Doug Tygar, and Anthony Joseph. Evasion and hardening of tree ensemble classifiers. In *International conference on machine learning*, pages 2387–2396. PMLR, 2016.

[51] Davinder Kaur, Suleyman Uslu, Kaley J Rittichier, and Arjan Durresi. Trustworthy artificial intelligence: a review. *ACM computing surveys (CSUR)*, 55(2):1–38, 2022.

[52] Lukasz A Kurgan, Krzysztof J Cios, Ryszard Tadeusiewicz, Marek Ogiela, and Lucy S Goodenday. Knowledge discovery approach to automated cardiac SPECT diagnosis. *Artificial intelligence in medicine*, 23(2):149–169, 2001.

[53] Jakub Kuzilek, Martin Hlosta, and Zdenek Zdrahal. Open university learning analytics dataset. *Scientific data*, 4(1):1–8, 2017.

[54] J. Larson, S. Mattu, L. Kirchner, and J. Angwin. How we analyzed the COMPAS recidivism algorithm. *ProPublica*, 2016.

[55] Benjamin Laufer, Manisch Raghavan, and Solon Barocas. Fundamental limits in the search for less discriminatory algorithms–and how to avoid them. *arXiv preprint arXiv:2412.18138*, 2024.

[56] Johann Laux, Sandra Wachter, and Brent Mittelstadt. Trustworthy artificial intelligence and the european union ai act: On the conflation of trustworthiness and acceptability of risk. *Regulation & Governance*, 18(1):3–32, 2024.

[57] Tai Le Quy, Arjun Roy, Vasileios Iosifidis, Wenbin Zhang, and Eirini Ntoutsi. A survey on datasets for fairness-aware machine learning. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12(3):e1452, 2022.

[58] Bo Li, Peng Qi, Bo Liu, Shuai Di, Jingen Liu, Jiquan Pei, Jinfeng Yi, and Bowen Zhou. Trustworthy ai: From principles to practices. *ACM Computing Surveys*, 55(9):1–46, 2023.

[59] Zheng Li and Yang Zhang. Membership leakage in label-only exposures. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 880–895, 2021.

[60] Ningyi Liao, Shufan Wang, Liyao Xiang, Nanyang Ye, Shuo Shao, and Pengzhi Chu. Achieving adversarial robustness via sparsity. *Machine Learning*, pages 1–27, 2022.

[61] Huawei Lin, Jun Woo Chung, Yingjie Lao, and Weijie Zhao. Machine unlearning in gradient boosting decision trees. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1374–1383, 2023.

[62] Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, and Margo Seltzer. Generalized and scalable optimal sparse decision trees. In *International Conference on Machine Learning*, pages 6150–6160. PMLR, 2020.

[63] Max Little, Patrick Mcsharry, Stephen Roberts, Declan Costello, and Irene Moroz. Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection. *Nature Precedings*, pages 1–1, 2007.

[64] Haochen Liu, Yiqi Wang, Wenqi Fan, Xiaorui Liu, Yaxin Li, Shaili Jain, Yunhao Liu, Anil Jain, and Jiliang Tang. Trustworthy ai: A computational perspective. *ACM Transactions on Intelligent Systems and Technology*, 14(1):1–59, 2022.

[65] Jiachang Liu, Chudi Zhong, Boxuan Li, Margo Seltzer, and Cynthia Rudin. FasterRisk: Fast and accurate interpretable risk scores. In *Neural Information Processing Systems (NeurIPS)*, 2022.

[66] Volker Lohweg. Banknote Authentication. UCI Machine Learning Repository, 2012. DOI: https://doi.org/10.24432/C55P57.

[67] Charles Marx, Flavio Calmon, and Berk Ustun. Predictive multiplicity in classification. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 6765–6774, 2020.

[68] Hayden McTavish, Chudi Zhong, Reto Achermann, Ilias Karimalis, Jacques Chen, Cynthia Rudin, and Margo Seltzer. Fast sparse decision tree optimization via reference ensembles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9604–9613, 2022.

[69] Anna P Meyer, Yea-Seul Kim, Aws Albarghouthi, and Loris D'Antoni. Perceptions of the fairness impacts of multiplicity in machine learning. *arXiv preprint arXiv:2409.12332*, 2024.

[70] Shira Mitchell, Eric Potash, Solon Barocas, Alexander D'Amour, and Kristian Lum. Algorithmic fairness: Choices, assumptions, and definitions. *Annual review of statistics and its application*, 8(1):141–163, 2021.

[71] Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

[72] Sasi Kumar Murakonda and Reza Shokri. Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning. In *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2020.

[73] Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–46, 2025.

[74] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16:1–85, 2022.

[75] Cynthia Rudin, Chudi Zhong, Lesia Semenova, Margo Seltzer, Ronald Parr, Jiachang Liu, Srikar Katta, Jon Donnelly, Harry Chen, and Zachery Boner. Amazing things come from having many good models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.

[76] Jérôme Rutinowski, Simon Klüttermann, Jan Endendyk, Christopher Reining, and Emmanuel Müller. Benchmarking trust: A metric for trustworthy machine learning. In *World Conference on Explainable Artificial Intelligence*, pages 287–307. Springer, 2024.

[77] Mohammed Saeed, Christine Lieu, Greg Raber, and Roger G Mark. Mimic ii: a massive temporal icu patient database to support research in intelligent patient monitoring. In *Computers in Cardiology*, pages 641–644. IEEE, 2002.

[78] Sebastian Schelter, Stefan Grafberger, and Ted Dunning. Hedgecut: Maintaining randomised trees for low-latency machine unlearning. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1545–1557, 2021.

[79] Lesia Semenova, Cynthia Rudin, and Ronald Parr. On the existence of simpler machine learning models. In *ACM Conference on Fairness, Accountability, and Transparency (ACM FAccT)*, 2022.

[80] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[81] Thibault Simonetto, Salah Ghamizi, and Maxime Cordy. Tabularbench: Benchmarking adversarial robustness for tabular deep learning in real-world use-cases. *Advances in Neural Information Processing Systems*, 37:78394–78430, 2024.

[82] Jack W Smith, James E Everhart, WC Dickson, William C Knowler, and Robert Scott Johannes. Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the annual symposium on computer application in medical care*, page 261. American Medical Informatics Association, 1988.

[83] Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2615–2632, 2021.

[84] Jacobus van der Linden, Mathijs de Weerdt, and Emir Demirović. Fair and optimal decision trees: A dynamic programming approach. *Advances in Neural Information Processing Systems*, 35:38899–38911, 2022.

[85] Sicco Verwer and Yingqian Zhang. Learning optimal classification trees using a binary linear program formulation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1625–1632, 2019.

[86] Matheus Guedes Vilas Boas, Haroldo Gambini Santos, Luiz Henrique de Campos Merschmann, and Greet Vanden Berghe. Optimal decision trees for the algorithm selection problem: integer programming based approaches. *International Transactions in Operational Research*, 28(5): 2759–2781, 2021.

[87] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

[88] Daniël Vos and Sicco Verwer. Efficient training of robust decision trees against adversarial examples. In *International conference on machine learning*, pages 10586–10595. PMLR, 2021.

[89] Daniël Vos and Sicco Verwer. Robust optimal classification trees against adversarial examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8520–8528, 2022.

[90] Daniël Vos, Jelle Vos, Tianyu Li, Zekeriya Erkin, and Sicco Verwer. Differentially-private decision trees and provable robustness to data poisoning. *arXiv preprint arXiv:2305.15394*, 2023.

[91] Tong Wang, Cynthia Rudin, Finale Doshi-Velez, Yimin Liu, Erica Klampfl, and Perry MacNeille. A Bayesian framework for learning rule sets for interpretable classification. *Journal of Machine Learning Research*, 18(70):1–37, 2017.

[92] Haider J Warraich, Troy Tazbaz, and Robert M Califf. Fda perspective on the regulation of artificial intelligence in health care and biomedicine. *JAMA*, 333(3):241–247, 2025.

[93] Jamelle Watson-Daniels, David C Parkes, and Berk Ustun. Predictive multiplicity in probabilistic classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 10306–10314, 2023.

[94] Jamelle Watson-Daniels, Flavio du Pin Calmon, Alexander D'Amour, Carol Long, David C Parkes, and Berk Ustun. Predictive churn with the set of good models. *arXiv preprint arXiv:2402.07745*, 2024.

[95] Hilde Weerts, Miroslav Dudík, Richard Edgar, Adrin Jalali, Roman Lutz, and Michael Madaio. Fairlearn: Assessing and improving fairness of ai systems, 2023. URL `http://jmlr.org/papers/v24/23-0389.html`.

[96] William H Wolberg and Olvi L Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the national academy of sciences*, 87(23):9193–9196, 1990.

[97] Zhaomin Wu, Junhui Zhu, Qinbin Li, and Bingsheng He. Deltaboost: Gradient boosting decision trees with efficient machine unlearning. *Proceedings of the ACM on Management of Data*, 1(2):1–26, 2023.

[98] Ruicheng Xian and Han Zhao. A Unified Post-Processing Framework for Group Fairness, 2024.

[99] Ruicheng Xian, Lang Yin, and Han Zhao. Fair and Optimal Classification via Post-Processing. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[100] Rui Xin, Chudi Zhong, Zhi Chen, Takuya Takagi, Margo Seltzer, and Cynthia Rudin. Exploring the whole rashomon set of sparse decision trees. *Advances in neural information processing systems*, 35:14071–14084, 2022.

[101] I-Cheng Yeh, King-Jang Yang, and Tao-Ming Ting. Knowledge discovery on RFM model using Bernoulli sequence. *Expert Systems with applications*, 36(3):5866–5871, 2009.

[102] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.

[103] Chudi Zhong, Zhi Chen, Jiachang Liu, Margo Seltzer, and Cynthia Rudin. Exploring and interacting with the set of good sparse generalized additive models. In *Neural Information Processing Systems (NeurIPS)*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are consistent with the paper's scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, we have discussed the limitation of the work in the conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper is a benchmark study and does not include theoretical results or formal proofs. The focus is on empirical evaluation.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Experimental setups are detailed in Appendix B-F. Code is available in supplement.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Datasets used in this paper are publicly available. Code is available in supplement. We have detailed experimental setup in Appendix B-F.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Please see Appendix C-F.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report mean and standard deviation over multiple runs for evaluation metrics. We explain the computation of reported statistics.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute details are available in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our work adheres to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have mentioned the social impact in the conclusion.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper doesn't release models that have the potential to cause harm.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use open access datasets and baselines and cite the sources of all the datasets and baselines we used in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the code for this paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We used LLM only for editing and improving the clarity of wording.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix

Contents

A Datasets

Table 5 lists all the datasets we considered in this paper. We report the number of samples, features, class imbalance, and also for which trustworthy metrics we used the datasets. Some datasets we collected based on the baseline papers (e.g., oulad, german credit). In other cases (e.g., fico and mimic), we select important datasets from high-stakes domains where trustworthiness is highly valued.

Table 5: Summary of datasets. In the metrics column, R represents Robustness, P represents Privacy (Membership inference attack), U represents Machine unlearning, and F represents Fairness. * indicates the dataset has been binarized according to [57].

Dataset	# Inst.	# Feat.	% Pos.	Metrics
adult* [24]	45222	17	24.78%	P/U/F
bank* [71]	45211	46	11.70%	P/U/F
banknote [66]	1372	4	44.46%	R
blood [101]	748	4	23.80%	R
breast [96]	683	9	34.99%	R
carryout [91]	2280	22	73.77%	P/U
compas-orig [54]	6907	7	46.27%	R/F
compas* [54]	6907	9	46.27%	P/U/F
diabetes [82]	768	8	34.90%	R
fico [28]	10459	23	52.19%	R
haberman [39]	306	3	26.47%	R
german-credit* [24]	1000	69	69.97 %	P/F
mimic [77]	24508	17	12.25%	R/P
oulad* [53]	21562	45	67.97%	P/F
parkinsons [63]	195	22	75.38%	R
restaurant [91]	2653	22	70.90 %	P/U
sonar [35]	208	60	53.37%	R
spectf [52]	267	44	79.40%	R
spambase [44]	4601	57	39.40%	R
stud-mat* [17]	649	55	32.89%	F
stud-por* [17]	649	55	15.41%	F
wine-q [18]	6497	11	63.31%	R

B Experiments: Robustness

For both robustness and stability, we evaluate CART, FPRDT, GROOT, ROCT-N, ROCT-V, and TreeFARMS across thirteen datasets (Table 5) with standard five-fold nested cross-validation. We configure each method as follows:

- **TreeFARMS**: We tune the depth=$\{1, 2, 3, 4\}$ and $\lambda = \{0.02, 0.015, 0.01, 0.005\}$. Since it is infeasible to target every model in the Rashomon set at inference time, we use the optimal tree as a reference to generate adversarial examples. We then select a tree within the Rashomon set that exhibits robustness against adversarial attacks on the held-out selection set. We denote this tree as RSET_kan. We also report the performance of the optimal tree in the Rashomon set (RSET_opt), a tree with the minimum and maximum number of leaves (RSET_min and RSET_max, respectively). If multiple such trees exist, we choose one randomly.
- **GROOT**: We tune the depth= $\{1, 2, 3, 4\}$, fix the minimum number of samples required to split an internal node (min sample split) to 10 and the minimum number of samples required to be at a leaf node (min sample leaf) to 5.
- **FPRDT** We tune the depth= $\{1, 2, 3, 4\}$, fix min sample split to 10 and min sample leaf to 5.
- **ROCT-V**: We set fixed depth 2 and 1800 seconds time limit.
- **ROCT-N**: We set fixed depth 2 and 1800 seconds time limit.
- **CART**: We tune the depth= $\{1, 2, 3, 4\}$, fix min_sample_split to 10 and min_sample_leaf to 5.

For adversarial robustness attack, we set the attack strength θ to 0.1, adjusting it for some datasets to reproduce results in [88] (see more in Appendix B.1). For stability perturbations, we set the normal distribution parameters that determine the confidence level to $\rho = 0.9$ and $\sigma = 0.1$. During the evaluation, we consider the regular setting as described above, apply a shift of $[-0.2, -0.1, 0.1]$ to q_t^j, and resample q_t^j uniformly in the range of ± 0.05. By evaluating based on the stronger perturbations, we are testing the out-of-distribution performance of the different methods. For each of the five settings, 5000 trials are performed to compute meaningful statistics.

B.1 Adversarial Robustness

Setup: We ran adversarial robustness experiment on 13 datasets: banknote, blood(-transfusion), breast(-cancer), compas(-orig), diabetes, fico, haberman, mimic, parkinsons, sonar, spambase, spectf, wine-q(uality), as shown in table 5. Both TreeFARMS and ROCT-N solve NP-hard optimization problems, requiring preprocessing (binarization) for real-valued features. We applied GOSDT threshold guessing with n_estimator=30, max_depth=2, learning_rate=0.1, backselect=True [68]. The resulting tree is converted into a standard tree structure with appropriate features and thresholds by replacing binary splits with their corresponding threshold values. This step is necessary for adversarial attacks to generate meaningful perturbation.

Result: Table 6 shows the test accuracy of all models on 13 datasets we used in adversarial robustness experiments. Trees within the Rashomon set (e.g., RSET_opt and RSET_max) perform competitively and often achieve the highest test accuracy (bolded). In contrast, robust tree models such as GROOT, ROCT-N, and ROCT-V generally have lower accuracy, suggesting that robustness-focused optimization may come at the cost of predictive performance.

Table 7 shows accuracy on adversarial samples generated from the test dataset. The highest adversarial accuracy for each dataset is in bold. In our setup, the attack targets the optimal tree in the Rashomon set, so it often has lower adversarial accuracy. However, many other trees within the Rashomon set remain robust. RSET_kan achieves the highest accuracy on most datasets, indicating that the selected trees can provide both robustness and accurate predictions.

We also investigate how different representative trees within the Rashomon set perform when the attack targets a randomly selected tree rather than the optimal tree. Table 8 shows that attacking different random trees from the Rashomon set leads to different robustness performances of representative trees. In some cases, RSET_kan, the model chosen from the selection set, achieves the highest accuracy. However, in most cases, RSET_min, the tree with the fewest leaves, outperforms the others. This finding suggests that sparser trees tend to be more robust against adversarial attacks compared to their more complex counterparts, such as RSET_max, as we discussed in Section 4.3.

Figure 4 compares the adversarial accuracy of trees within the Rashomon set to various baseline models on the test set. This figure expands on the discussion in Section 4.1. The light blue density plot represents the distribution of adversarial accuracy for all trees in the Rashomon set or a random subsample of 100,000 trees of the Rashomon set, whichever is smaller. Vertical lines indicate the accuracy of baseline models, including CART, FPRDT, GROOT, and ROCT, as well as specific Rashomon set trees such as the optimal tree (RSET_opt), the sparsest tree (RSET_min), the most complex tree (RSET_max), and RSET_kan, a tree selected from the held-out validation set. As we can see, many trees within the Rashomon set achieve higher accuracy than the baselines, indicating their robustness against adversarial attacks. This suggests that careful selection within the Rashomon set can yield robust models for critical tasks facing adversarial attacks, as attackers can not possibly generate 100,000 distinct adversarial examples. Additionally, these density plots highlight areas for improving our selection metric. For instance, while the selected Rashomon set tree performs best on certain datasets, it does not always achieve the best performance in others (e.g., breast cancer). This finding might inspire future research on developing better selection criteria.

Table 9 extends Table 3 in Section 4.2, reporting the percentage of trees within the Rashomon set that outperforms the baseline models on the test set. Across all datasets, the Rashomon set includes trees that achieve higher adversarial accuracy than the baselines. In certain cases, such as banknote and fico datasets, nearly 80% of trees in the Rashomon set achieve higher test adversarial accuracy than FPRDT and ROCT-V.

Table 6: Comparison of test accuracy across all methods. The highest accuracy for each dataset is bolded. Trees within the Rashomon set consistently achieve competitive performance.

Dataset	CART	FPRDT	GROOT	ROCT-N	ROCT-V	RSET_opt	RSET_min	RSET_max	RSET_sel
banknote	0.953 ± 0.017	0.908 ± 0.009	0.873 ± 0.050	0.901 ± 0.017	0.904 ± 0.013	0.962 ± 0.015	0.935 ± 0.010	**0.972 ± 0.015**	0.960 ± 0.012
blood	**0.783 ± 0.025**	0.763 ± 0.006	0.762 ± 0.003	0.762 ± 0.003	0.767 ± 0.006	0.782 ± 0.016	0.762 ± 0.003	0.777 ± 0.010	0.770 ± 0.016
breast	0.944 ± 0.012	0.94 ± 0.009	0.938 ± 0.012	0.928 ± 0.037	0.939 ± 0.020	0.952 ± 0.012	0.918 ± 0.027	**0.953 ± 0.012**	0.931 ± 0.033
compas	0.665 ± 0.013	0.564 ± 0.011	0.559 ± 0.009	0.647 ± 0.026	0.540 ± 0.006	0.673 ± 0.010	0.645 ± 0.016	**0.677 ± 0.019**	0.653 ± 0.017
diabetes	0.738 ± 0.025	0.737 ± 0.025	0.737 ± 0.025	**0.749 ± 0.028**	0.73 ± 0.026	**0.749 ± 0.037**	0.738 ± 0.027	0.738 ± 0.041	0.743 ± 0.032
fico	0.703 ± 0.012	0.538 ± 0.006	0.535 ± 0.007	0.646 ± 0.047	0.522 ± 0.000	0.704 ± 0.007	0.698 ± 0.008	**0.711 ± 0.010**	0.698 ± 0.008
haberman	**0.739 ± 0.038**	0.732 ± 0.008	0.725 ± 0.014	0.732 ± 0.041	0.732 ± 0.008	0.732 ± 0.043	0.735 ± 0.005	0.748 ± 0.035	0.732 ± 0.028
mimic	0.880 ± 0.001	0.878 ± 0.000	0.878 ± 0.001	0.878 ± 0.000	0.878 ± 0.000	0.878 ± 0.000	0.878 ± 0.000	**0.883 ± 0.002**	0.881 ± 0.002
parkinsons	0.846 ± 0.036	0.810 ± 0.023	0.800 ± 0.038	0.851 ± 0.049	0.821 ± 0.031	0.867 ± 0.033	**0.872 ± 0.044**	0.851 ± 0.049	0.867 ± 0.046
sonar	0.716 ± 0.064	0.707 ± 0.064	0.683 ± 0.055	0.745 ± 0.02	0.650 ± 0.067	**0.765 ± 0.053**	0.707 ± 0.058	0.702 ± 0.047	0.726 ± 0.069
spambase	0.894 ± 0.006	0.815 ± 0.019	0.805 ± 0.011	0.815 ± 0.011	0.651 ± 0.049	0.902 ± 0.007	0.850 ± 0.004	**0.908 ± 0.004**	0.892 ± 0.015
spectf	0.772 ± 0.032	0.794 ± 0.002	0.794 ± 0.002	**0.809 ± 0.037**	0.794 ± 0.002	0.753 ± 0.042	0.801 ± 0.032	0.749 ± 0.034	0.738 ± 0.038
wine-q	0.729 ± 0.009	0.635 ± 0.001	0.633 ± 0.001	0.672 ± 0.042	0.634 ± 0.002	0.733 ± 0.008	0.693 ± 0.011	**0.745 ± 0.006**	0.725 ± 0.016

Table 7: Comparison of test accuracy on adversarial samples. The highest accuracy for each dataset is in bold. The optimal tree RSET_opt in the Rashomon set is attacked and RSET_sel is the model within the Rashomon set selected from the selection set.

Dataset	CART	FPRDT	GROOT	ROCT-N	ROCT-V	RSET_opt	RSET_min	RSET_max	RSET_sel
banknote	0.506 ± 0.033	0.665 ± 0.023	0.641 ± 0.053	0.509 ± 0.107	0.659 ± 0.018	0.534 ± 0.052	0.693 ± 0.131	0.737 ± 0.039	**0.774 ± 0.094**
blood	0.378 ± 0.047	0.761 ± 0.007	0.762 ± 0.003	0.762 ± 0.003	**0.763 ± 0.007**	0.411 ± 0.084	0.762 ± 0.003	0.558 ± 0.095	0.522 ± 0.137
breast	0.228 ± 0.071	**0.873 ± 0.033**	0.843 ± 0.026	0.198 ± 0.071	0.813 ± 0.038	0.224 ± 0.026	0.628 ± 0.359	0.488 ± 0.36	0.53 ± 0.353
compas	0.171 ± 0.054	0.549 ± 0.013	**0.557 ± 0.01**	0.213 ± 0.138	0.537 ± 0.002	0.193 ± 0.016	0.487 ± 0.025	0.461 ± 0.081	0.481 ± 0.112
diabetes	0.594 ± 0.067	0.682 ± 0.04	0.676 ± 0.038	0.613 ± 0.065	0.67 ± 0.032	0.581 ± 0.047	**0.728 ± 0.03**	0.626 ± 0.028	0.698 ± 0.034
fico	0.223 ± 0.033	0.523 ± 0.003	0.532 ± 0.007	0.378 ± 0.062	0.522 ± 0.0	0.265 ± 0.008	0.579 ± 0.023	0.553 ± 0.074	**0.689 ± 0.006**
haberman	0.447 ± 0.327	0.732 ± 0.008	0.703 ± 0.043	0.562 ± 0.187	0.729 ± 0.014	0.558 ± 0.252	**0.735 ± 0.005**	0.538 ± 0.139	0.598 ± 0.162
mimic	0.466 ± 0.375	0.878 ± 0.000	0.878 ± 0.001	0.878 ± 0.000	0.878 ± 0.000	0.878 ± 0.000	0.878 ± 0.000	0.883 ± 0.002	**0.881 ± 0.002**
parkinsons	0.518 ± 0.108	**0.785 ± 0.023**	0.728 ± 0.034	0.544 ± 0.056	0.769 ± 0.063	0.523 ± 0.102	0.774 ± 0.064	0.631 ± 0.126	0.749 ± 0.105
sonar	0.407 ± 0.124	0.548 ± 0.037	0.582 ± 0.04	0.408 ± 0.077	0.5 ± 0.089	0.236 ± 0.063	0.596 ± 0.093	0.501 ± 0.145	**0.626 ± 0.173**
spambase	0.000 ± 0.001	**0.735 ± 0.016**	0.721 ± 0.020	0.075 ± 0.065	0.512 ± 0.238	0.074 ± 0.042	0.762 ± 0.064	0.635 ± 0.118	0.636 ± 0.155
SPECTF	0.637 ± 0.217	**0.794 ± 0.002**	**0.794 ± 0.002**	0.529 ± 0.159	**0.794 ± 0.002**	0.262 ± 0.055	0.783 ± 0.054	0.521 ± 0.154	0.618 ± 0.057
wine-q	0.203 ± 0.081	0.632 ± 0.002	0.632 ± 0.002	0.459 ± 0.176	**0.633 ± 0.001**	0.317 ± 0.058	0.611 ± 0.046	0.578 ± 0.077	0.631 ± 0.090

Table 8: Adversarial accuracy of different Rashomon set trees when the attack targets a randomly selected tree instead of the optimal tree.

Dataset	**RSET_opt**	**RSET_min**	**RSET_max**	**RSET_kan**
banknote	0.772 ± 0.066	0.767 ± 0.076	0.794 ± 0.08	**0.812 ± 0.046**
blood	0.71 ± 0.089	**0.762 ± 0.003**	0.485 ± 0.076	0.639 ± 0.085
breast	0.726 ± 0.266	0.918 ± 0.027	**0.946 ± 0.018**	0.911 ± 0.034
compas	**0.529 ± 0.081**	0.389 ± 0.152	0.503 ± 0.052	0.472 ± 0.087
diabetes	0.71 ± 0.037	**0.716 ± 0.054**	0.693 ± 0.063	0.706 ± 0.061
fico	0.564 ± 0.054	0.547 ± 0.044	0.576 ± 0.079	**0.64 ± 0.043**
haberman	0.601 ± 0.192	**0.735 ± 0.005**	0.503 ± 0.117	0.578 ± 0.15
mimic	**0.878 ± 0.0**	**0.878 ± 0.0**	0.849 ± 0.039	0.827 ± 0.107
parkinsons	0.831 ± 0.029	**0.856 ± 0.064**	0.703 ± 0.142	0.8 ± 0.103
sonar	0.616 ± 0.084	0.625 ± 0.039	0.606 ± 0.066	**0.65 ± 0.101**
spambase	0.764 ± 0.119	**0.853 ± 0.013**	0.708 ± 0.112	0.665 ± 0.121
SPECTF	0.636 ± 0.109	**0.772 ± 0.047**	0.697 ± 0.109	0.618 ± 0.105
wine-q	0.604 ± 0.059	**0.609 ± 0.056**	0.583 ± 0.078	0.57 ± 0.072

Table 9: Percentage of trees in Rashomon set that have better test adversarial accuracy than the baselines.

Dataset	FPRDT	ROCT-V
banknote	**94.66 ± 4.48%**	**95.34 ± 3.43%**
blood	4.91 ± 4.79%	5.26 ± 6.64%
breast	56.09 ± 11.98%	**75.58 ± 8.36%**
compas	23.96 ± 10.73%	28.15 ± 8.61%
diabetes	46.59 ± 40.54%	**62.34 ± 17.76%**
fico	**79.35 ± 7.06%**	**79.58 ± 6.93%**
haberman	18.61 ± 27.94%	18.13 ± 28.02%
mimic	**68.16 ± 8.51%**	**68.42 ± 6.45%**
parkinsons	45.19 ± 18.39%	55.87 ± 34.06%
sonar	**70.06 ± 15.72%**	**81.93 ± 14.15%**
spambase	44.37 ± 4.00%	**78.93 ± 11.49%**
SPECTF	3.16 ± 3.18%	3.16 ± 3.18%
wine-q	24.65 ± 3.57%	23.79 ± 2.79%

B.2 Stability

Setup: To ensure stability, we evaluate our models on the same 13 datasets used in the adversarial robustness experiments. We consider five different noise perturbations, briefly introduced in Section 3.2. Additionally, threshold guessing is applied to ROCT-N and TreeFARMS using the same parameters described in Section B.1.

For baselines that require an attack strength specification, we set the budget to 10% by default. This choice is justified by the expected perturbation value for noise, which is approximately 11%. This value is derived from the expectation of a geometric distribution, $\frac{1}{q_t^j}$, where the expected value of q_t^j corresponds to the mean of our normal distribution, which is 0.9. Thus, using a 10% budget ensures consistency with the setup used in other sections.

For each perturbation, we conduct 5000 repeated trials, resampling the noise in each iteration while maintaining the fixed confidence level. We compute the average and worst-case scores and record the standard deviation of the results. This process is repeated across five folds.

Results: Figure 5 visualizes the stability performance of trees across 13 datasets under five different types of noise perturbations. Note that only ROCT-N explicitly accounts for these perturbations during training, while all other methods are trained for accuracy or for accuracy and robustness and evaluated for stability on perturbed test data. As shown in the figure, different representative trees from the Rashomon set (bars colored in blue palette) are generally comparable to ROCT-N, indicating that while the Rashomon set is not explicitly designed for stability, it contains trees that perform well under perturbations. Interestingly, CART also performs well for stability.

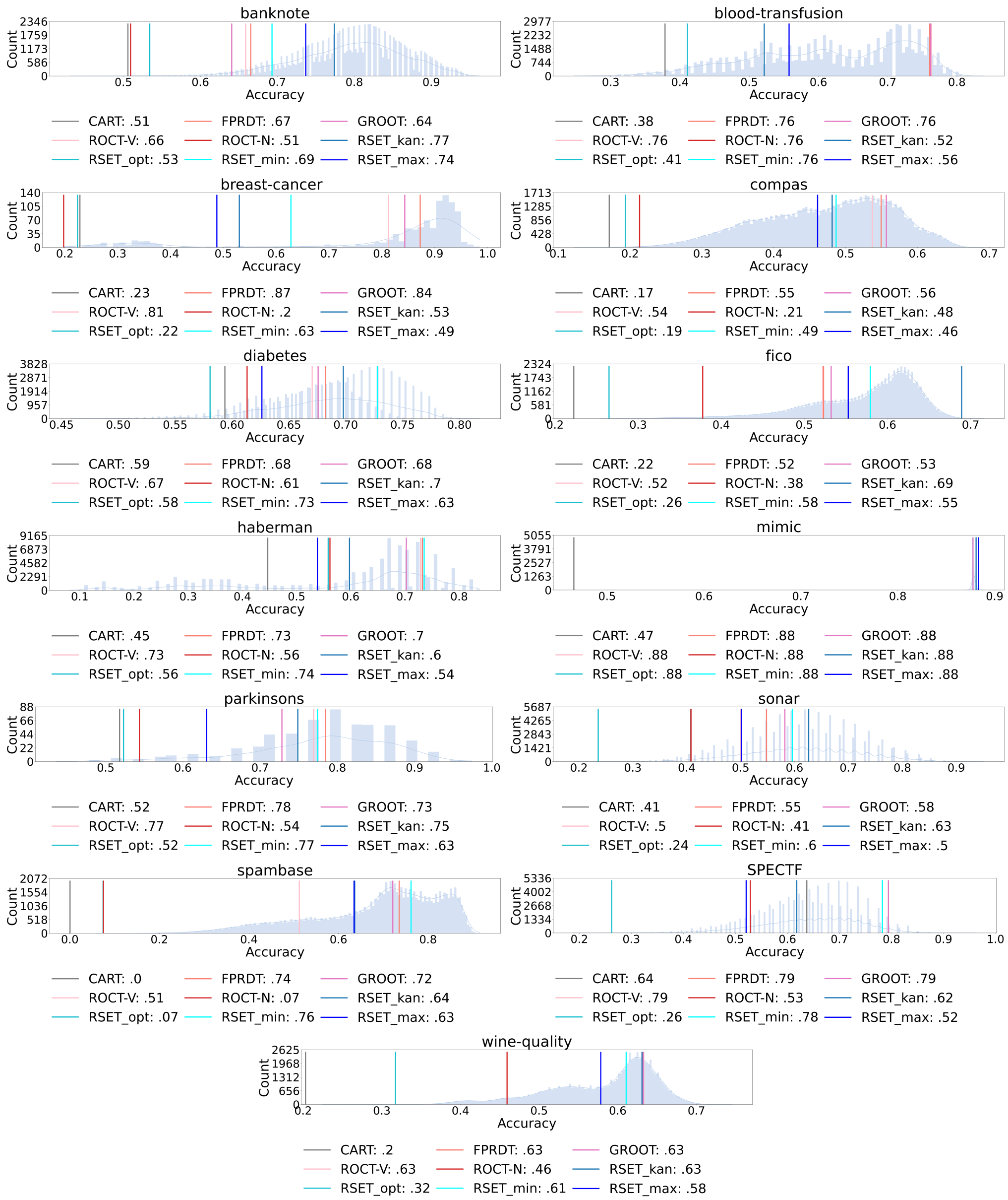


Figure 4: Comparison of adversarial accuracy between trees in the Rashomon set (light blue histogram) and baseline models (vertical lines) on the test set. Most Rashomon set trees achieve higher robustness than baselines.

Training Time: Time consumption is an important metric to consider when evaluating model performance. We investigate whether TreeFARMS' consistently strong results come at the cost of significantly longer training times. Table 10 reports the training time (in seconds) for each method. Greedy methods such as CART, FPRDT, and GROOT complete training quickly due to their heuristic-based construction. In contrast, ROCT-N, ROCT-V, and TreeFARMS aim to find globally optimal solutions, which are NP-hard problems. As a result, ROCT-N and ROCT-V often reach or nearly approach the 1,800-second time limit. TreeFARMS (RSET) usually completes training within a reasonable time frame. Note that the table shows training latency – for example, ROCT-N continues processing beyond the time limit before terminating.

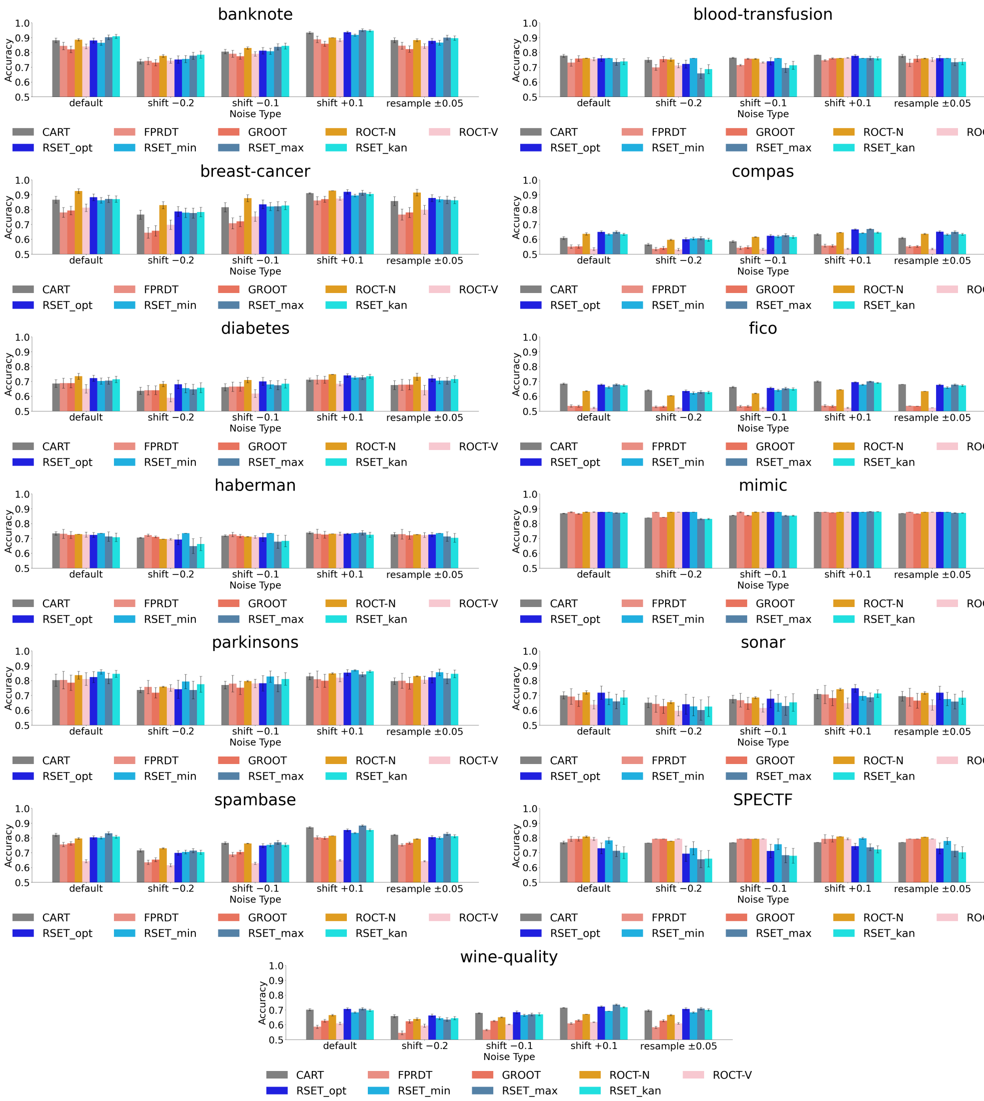


Figure 5: Comparison of stability accuracy across different methods under 5 perturbation types. Error bars represent the average standard deviation across 5000 trials over five folds.

C Experiments: Privacy

C.1 Membership Inference Attack

Configuration: We perform five-fold cross-validation for all methods with a fixed tree depth of four and further configure our methods as follows:

- **TreeFARMS**: We set $\lambda = 0.01$. We evaluate trees with the minimum and maximum number of leaves within the Rashomon set (RSET_min and RSET_max respectively) and also evaluate the optimal model (RSET_opt).
- **PRIVA**: We set min sample split to 10 and min sample leaf to 5. We further set the privacy threshold η_1 to 0.1.
- **BDPT**: We set min sample split to 10 and min sample leaf to 5. We further set the privacy threshold η_1 to 0.1.
- **DPLDT**: We set the privacy η_1 threshold to 0.1.

Both DPLDT and BDPT were designed as base estimators for Random Forest, but we evaluate whether a single estimator can provide sufficient privacy against membership inference attacks.

Table 10: Training time (in seconds) for each method across different datasets in the robustness evaluation. Greedy methods (CART, FPRDT, and GROOT) complete training quickly, while ROCT-N, ROCT-V, and TreeFARMS (RSET) aim to find globally optimal solutions.

Dataset	**CART**	**FPRDT**	**GROOT**	**ROCT-N**	**ROCT-V**	**RSET**
banknote	0.002563	0.022296	0.005495	1595.718707	1801.358524	12.700638
blood-transfusion	0.001143	0.004410	0.001120	1818.296185	1611.371671	5.452538
breast-cancer	0.001125	0.010057	0.003129	1808.178998	462.495866	0.034354
compas	0.002892	0.069576	0.010519	1831.967454	1803.826331	26.162563
diabetes	0.001703	0.005114	0.001367	1815.053041	1800.516992	2.381064
fico	0.018382	0.106858	0.054123	1829.859036	1809.198182	287.172908
haberman	0.000886	0.001329	0.001058	1261.664404	1250.347395	2.144632
mimic	0.032952	0.104270	0.100324	1913.298006	1821.218696	546.053602
parkinsons	0.001485	0.004239	0.002995	1199.133712	1800.299358	0.035118
sonar	0.002356	0.015405	0.008226	1812.106851	1800.470384	784.626750
spambase	0.015867	0.154811	0.054068	1871.833115	1809.663264	213.106217
spectf	0.001426	0.002304	0.005296	1781.597735	1800.519040	1186.967646
wine-quality	0.008854	0.108675	0.012904	1889.285067	1804.638895	215.480192

Table 11: Comparison of test accuracy between differentially private tree models and trees within the Rashomon set across seven datasets.

Dataset	**CART**	**BDPT**	**DPLDT**	**PRIVA**	**RSET_opt**	**RSET_min**	**RSET_max**
adult	0.836 ± 0.005	0.752 ± 0.000	0.752 ± 0.001	0.752 ± 0.000	0.836 ± 0.005	0.793 ± 0.002	**0.839 ± 0.006**
bank	**0.897 ± 0.002**	0.883 ± 0.000	0.883 ± 0.001	0.883 ± 0.000	0.884 ± 0.003	0.883 ± 0.000	0.895 ± 0.002
compas	**0.664 ± 0.009**	0.545 ± 0.000	0.641 ± 0.011	0.600 ± 0.010	0.663 ± 0.006	0.630 ± 0.011	0.663 ± 0.009
fico	0.707 ± 0.003	0.510 ± 0.006	0.521 ± 0.001	0.629 ± 0.088	0.698 ± 0.008	0.698 ± 0.008	**0.713 ± 0.012**
german	0.684 ± 0.019	0.700 ± 0.000	0.683 ± 0.014	0.711 ± 0.032	0.694 ± 0.024	**0.719 ± 0.017**	0.707 ± 0.026
mimic	0.880 ± 0.001	0.876 ± 0.000	0.876 ± 0.001	0.877 ± 0.001	0.878 ± 0.000	0.878 ± 0.000	**0.882 ± 0.002**
oulad	**0.686 ± 0.002**	0.680 ± 0.000	0.680 ± 0.000	0.680 ± 0.000	0.680 ± 0.000	0.680 ± 0.000	**0.686 ± 0.002**

Setup: We evaluate the membership inference attack on seven datasets: adult, bank, compas, fico, german-credit, mimic, and oulad. Among these, adult, bank, compas, german-credit (german), and oulad are binarized datasets. As before, we apply GOSDT threshold guessing for TreeFARMS on the fico and mimic datasets using `n_estimator`=30, `max_depth`=2, and `learning_rate`=0.1.

Result: Table 11 presents the standard test accuracy of different methods, including greedy tree CART, differentially private trees BDPT, DPLDT, and PRIVA, and representative trees within the Rashomon set. The highest accuracy for each dataset is bolded. The results show that trees within the Rashomon set (RSET_min and RSET_max) usually achieve higher test accuracy compared to all baselines. Differentially private trees often underperform. This is likely due to their reliance on randomized splitting or noise injection during training, which introduces additional variability and reduces accuracy.

Tables 12- 15 report the accuracy of four different attacks, ordered from weakest to strongest: baseline attack, label-only inference attacks, label-sup inference attack, and shadow model attack. A lower value indicates that the method is more resistant to attacks.

Overall, these attacks are not successful as the highest observed accuracy remains close to 0.5 in all four tables, meaning the attacker does not significantly outperform random guessing. Compared to 3 differential private tree methods, trees from the Rashomon set with the fewest leaves (RSET_min) achieve comparable or even better resistance against both attacks, despite the fact that the TreeFARMS algorithm does not incorporate any explicit randomness for privacy protection.

Table 12: Baseline membership inference attack success rates for different methods. Lower values indicate better resistance to attacks.

Dataset	**CART**	**BDPT**	**DPLDT**	**PRIVA**	**RSET_opt**	**RSET_min**	**RSET_max**
adult	0.497 ± 0.011	0.497 ± 0.017	**0.496 ± 0.016**	0.497 ± 0.017	0.497 ± 0.011	0.501 ± 0.01	0.501 ± 0.009
bank	0.503 ± 0.006	0.504 ± 0.01	0.504 ± 0.008	0.503 ± 0.008	0.503 ± 0.01	0.507 ± 0.01	**0.502 ± 0.012**
compas	0.484 ± 0.016	0.491 ± 0.019	**0.48 ± 0.025**	0.489 ± 0.014	0.484 ± 0.014	0.515 ± 0.032	0.516 ± 0.025
fico	0.505 ± 0.015	0.493 ± 0.006	0.492 ± 0.005	0.501 ± 0.006	0.496 ± 0.009	**0.485 ± 0.018**	0.508 ± 0.009
german	0.590 ± 0.066	**0.503 ± 0.03**	0.505 ± 0.033	0.528 ± 0.048	0.52 ± 0.047	0.542 ± 0.026	0.532 ± 0.058
mimic	0.503 ± 0.013	**0.502 ± 0.011**	0.502 ± 0.012	**0.502 ± 0.011**	**0.502 ± 0.011**	0.506 ± 0.011	0.504 ± 0.007
oulad	0.504 ± 0.014	0.500 ± 0.01	0.500 ± 0.01	0.500 ± 0.01	0.500 ± 0.01	**0.497 ± 0.013**	0.509 ± 0.014

Table 13: Label-only unsupervised MIA success rates for different methods. * means only one fold has successful attacks.

Dataset	**CART**	**BDPT**	**DPLDT**	**PRIVA**	**RSET_opt**	**RSET_min**	**RSET_max**
adult	0.497 ± 0.011	**Attack Failed**	0.501 ± 0.016	0.502 ± 0.016	0.497 ± 0.011	0.501 ± 0.01	0.501 ± 0.009
bank	0.503 ± 0.006	**Attack Failed**	0.509 ± 0.008	0.505 ± 0.01	0.5 (1/5)*	**Attack Failed**	0.502 ± 0.012
compas	0.484 ± 0.016	**Attack Failed**	0.479 ± 0.023	0.484 ± 0.016	0.484 ± 0.014	0.515 ± 0.032	0.516 ± 0.025
fico	0.5 ± 0.011	0.497 ± 0.004	0.492 ± 0.006	0.509 ± 0.006	0.496 ± 0.009	**0.485 ± 0.018**	0.508 ± 0.009
german	0.59 ± 0.066	**Attack Failed**	0.495 ± 0.042	0.528 ± 0.048	0.52 ± 0.047	0.542 ± 0.026	0.532 ± 0.058
mimic	0.506 ± 0.014	0.502 ± 0.001	0.503 ± 0.009	0.498 ± 0.009	**Attack Failed**	**Attack Failed**	0.504 ± 0.007
oulad	0.504 ± 0.014	**Attack Failed**	0.501 ± 0.008	0.5 ± 0.011	**Attack Failed**	**Attack Failed**	0.509 ± 0.014

Table 14: Label-only supervised MIA success rates for different methods.

Dataset	**CART**	**BDPT**	**DPLDT**	**PRIVA**	**RSET_opt**	**RSET_min**	**RSET_max**
adult	**0.488 ± 0.012**	0.5 ± 0.0	0.496 ± 0.006	0.492 ± 0.012	0.494 ± 0.007	0.501 ± 0.01	0.506 ± 0.009
bank	0.498 ± 0.007	0.5 ± 0.0	0.508 ± 0.017	**0.488 ± 0.013**	0.5 ± 0.0	0.5 ± 0.0	0.502 ± 0.02
compas	0.484 ± 0.016	0.5 ± 0.0	**0.481 ± 0.022**	0.489 ± 0.014	0.484 ± 0.014	0.515 ± 0.032	0.515 ± 0.026
fico	0.499 ± 0.024	0.498 ± 0.003	0.497 ± 0.01	0.501 ± 0.009	0.496 ± 0.009	**0.485 ± 0.018**	0.508 ± 0.006
german	0.59 ± 0.066	0.5 ± 0.0	**0.495 ± 0.043**	0.528 ± 0.048	0.52 ± 0.047	0.542 ± 0.026	0.532 ± 0.058
mimic	0.503 ± 0.011	0.503 ± 0.008	0.507 ± 0.004	0.508 ± 0.005	**0.5 ± 0.0**	**0.5 ± 0.0**	0.504 ± 0.012
oulad	0.5 ± 0.003	0.5 ± 0.0	0.499 ± 0.008	**0.496 ± 0.005**	0.5 ± 0.0	0.5 ± 0.0	0.504 ± 0.018

C.2 Machine Unlearning

Configuration: We evaluate TreeFARMS, DaRE, and GBDT unlearning on 5 datasets. Five-fold cross validation is used to tune hyperparameters:

- **TreeFARMS**: We tune depth=[2,3,4] and $\lambda = [0.01, 0.005, 0.001]$.
- **DaRE**: We first fix the number of random layers to 0 and tune G-DaRE, considering the maximum tree depth [1,3,5,10,20], the number of trees [10,25,50,100,250], and the number of threshold values per attribute [5,10,25,50]. After identifying the best configuration for G-DaRE, we tune the number of random layers from 1 to 10, stopping when the cross-validation score exceeds a 0.5% tolerance compared to the greedy model for R-DaRE.
- **GBDT unlearning**: Similar to DaRE, we first ignore random layers and tune the maximum number of leaves [5,10,15,20] and feature sampling rate [0.05, 0.1, 0.5, 1] for G-Bossting. We then tune the number of random layers from 1 to 4.

We randomly remove 0.5%, 1%, and 2% of training samples 10 times and compare the results of unlearning with those of retraining for each method. Note that we only need to construct one Rashomon set from TreeFARMS by setting $\epsilon = 0.04$.

Setup: We ran machine unlearning experiments on 5 datasets: carryout, restaurant, adult, compas, and bank. Details are shown in Table 5. Note that adult, compas, and bank datasets are binarized, while carryout and restaurant are real-valued. The datasets are split into training and test sets using an 80-20 split. All methods are fitted on the training set after hyperparameters have been tuned based on the configurations described in Section 3.3.2. Note that constructing the Rashomon set is NP-hard, so we apply GOSDT threshold guessing with n_estimator=30, max_depth=2, learning_rate=0.1, backselect=True [68] when fitting TreeFARMS on carryout and restaurant datasets.

Once models and the Rashomon set are constructed, we randomly remove 0.5%, 1%, and 2% of the original training data and repeat the process 10 times to evaluate the test performance of the unlearned and retrained model.

Results: Table 16 is a complete version of Table 2. It shows the proportion of test data with different predicted labels between the unlearned and retrained models for the Rashomon set, G-DaRE, R-DaRE,

Table 15: Shadow model MIA success rates for different methods.

Dataset	**CART**	**BDPT**	**DPLDT**	**PRIVA**	**RSET_opt**	**RSET_min**	**RSET_max**
adult	0.496 ± 0.015	**0.491 ± 0.012**	0.493 ± 0.012	0.492 ± 0.012	0.496 ± 0.017	0.5 ± 0.01	0.499 ± 0.014
bank	0.498 ± 0.007	0.5 ± 0.008	0.499 ± 0.01	0.498 ± 0.011	0.499 ± 0.007	0.506 ± 0.011	**0.495 ± 0.008**
compas	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0
fico	0.506 ± 0.016	0.497 ± 0.005	0.502 ± 0.004	0.505 ± 0.007	0.5 ± 0.005	**0.494 ± 0.013**	0.499 ± 0.012
german	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0	0.5 ± 0.0
mimic	**0.492 ± 0.008**	0.498 ± 0.01	0.498 ± 0.011	0.496 ± 0.01	0.497 ± 0.007	0.498 ± 0.012	0.497 ± 0.007
oulad	0.502 ± 0.008	0.504 ± 0.008	0.504 ± 0.008	0.504 ± 0.008	0.502 ± 0.01	**0.499 ± 0.014**	0.508 ± 0.016

Table 16: Proportion of test data with different predicted labels between the unlearned and retrained models (i.e., $\hat{y}_{\text{unlearn}} \neq \hat{y}_{\text{retrain}}$) after a subset of the original data is randomly removed. The Rashomon set achieves the lowest mismatch loss, as it is guaranteed to contain the optimal tree trained after a certain proportion of samples are removed.

Dataset	unlearn size	RSET	G-DARE	R-DARE	G-Boosting	R-Boosting
carryout	0.5%	0	1.272% ± 0.611%	1.601% ± 0.529%	5.680% ± 0.385%	2.961% ± 0.660%
	1%	0	1.294% ± 0.465%	1.601% ± 0.510%	5.548% ± 0.741%	3.224% ± 0.844%
	2%	0	1.469% ± 0.501%	1.689% ± 0.393%	4.978% ± 1.334%	3.662% ± 1.233%
restaurant	0.5%	0	2.015% ± 0.267%	1.620% ± 0.388%	4.708% ± 0.834%	4.087% ± 0.513%
	1%	0	1.902% ± 0.433%	1.808% ± 0.493%	4.746% ± 0.623%	3.691% ± 0.847%
	2%	0	2.147% ± 0.349%	1.846% ± 0.582%	4.501% ± 0.755%	3.974% ± 0.865%
adult	0.5%	0	0.008% ± 0.007%	0.066% ± 0.037%	0.052% ± 0.019%	0.170% ± 0.170%
	1%	0	0.018% ± 0.013%	0.067% ± 0.047%	0.064% ± 0.027%	0.175% ± 0.165%
	2%	0	0.010% ± 0.014%	0.060% ± 0.036%	0.051% ± 0.024%	0.238% ± 0.166%
compas	0.5%	0	0.000% ± 0.000%	0.000% ± 0.000%	0.470% ± 0.198%	1.741% ± 1.599%
	1%	0	0.000% ± 0.000%	0.000% ± 0.000%	0.478% ± 0.543%	1.895% ± 1.422%
	2%	0	0.000% ± 0.000%	0.000% ± 0.000%	0.810% ± 0.871%	1.879% ± 1.608%
bank	0.5%	0	0.587% ± 0.117%	0.201% ± 0.040%	0.911% ± 0.099%	1.081% ± 0.157%
	1%	0	0.607% ± 0.103%	0.248% ± 0.033%	0.888% ± 0.096%	1.006% ± 0.105%
	2%	0	0.631% ± 0.077%	0.289% ± 0.055%	0.850% ± 0.139%	1.066% ± 0.124%

G-Boosting, and R-Boosting. Since the Rashomon set has a theoretical guarantee that the optimal tree for the reduced dataset remains within the set after a subset of the training data is removed, the RSET column always reports 0. However, other methods do not have this guarantee.

Another important metric in machine unlearning experiments is time consumption. Table 17 compares the unlearning time and retraining time for each method after different proportions of the training data are removed in seconds. In general, unlearning is faster than retraining for all methods, and the timing is fairly comparable across different methods. Additionally, removing fewer samples results in faster unlearning times. In some cases, retraining may be faster than unlearning for the Rashomon set. This is because unlearning in the Rashomon set is actually a search process. When the Rashomon set is large, and the proportion of removed data is high, more models within the set may need to be evaluated to identify the optimal one, leading to a longer learning time.

D Experiments: Fairness

Setup: We ran fairness experiments on 7 datasets: adult, bank, compas, german-credit, oulad, student-mat, and student-por. Details are shown in Table 5. They are all binarized datasets. The datasets are split into training, selection, and test sets using a 70-10-20 split. One note is that we follow the convention of dropping the sensitive feature before training on TreeFARMS. All methods are fitted on the training set based on the configurations described in Section 3.4. The selection set is used to (1) find the best tree within the Rashomon set that minimizes the misclassification loss with a penalty on fairness disparity controlled by α and (2) apply the post-processing step for PostCART and PostXGB.

Note that TreeFARMS, FairOCT, PostCART, and PostXGB can handle all three fairness metrics, while DPF works only with statistical parity. We ran experiments 5 times and reported the mean and standard deviation for every method and dataset.

Following the setting in [84], we configure each method as follows:

- **TreeFARMS**: We construct the Rashomon set with depth=[2,3,4], $\lambda = 0.01, \epsilon = 0.05$ and select the best tree that minimizes 0-1 loss plus α times fairness disparity on the selection set. We set $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 1\}$.
- **DPF**: We train trees with depth=[2,3,4] and set $\delta = 0.01$.
- **FairOCT**: We train trees with depth=[2,3,4] and set $\delta = 0.01$.
- **PostCART**: We train CART with depth=[2,3,4] and the minimum sample per leaf 1% of the training sample size. We set $\alpha = 0.03$ for post-processing.
- **PostXGB**: We use default setting of XGBoost and set $\alpha = 0.03$.

Results: For trees computed by baselines and selected from the Rashomon set, we show the disparity in statistical parity and accuracy on the test set in Table 18 for depth 2 and Table 19 for depth 3 and 4. Here, the better results have lower disparity and higher accuracy.

Table 17: Comparison of unlearning time and retraining time (in seconds). Unlearning is usually faster than retraining across all methods.

Dataset	unlearn size	RSET		G-DARE		R-DARE		G-Boosting		R-Boosting	
		unlearn time	retrain time	unlearn time	retrain time	unlearn time	retrain time	unlearn time	retrain time	unlearn time	retrain time
carryout	0.5%	0.093 ± 0.029	0.087 ± 0.001	0.187 ± 0.029	0.784 ± 0.01	0.133 ± 0.016	0.738 ± 0.013	0.679 ± 0.066	0.548 ± 0.004	0.605 ± 0.054	0.533 ± 0.007
	1%	0.849 ± 0.098	0.088 ± 0.002	0.277 ± 0.022	0.785 ± 0.004	0.208 ± 0.017	0.742 ± 0.012	0.661 ± 0.007	0.550 ± 0.006	0.611 ± 0.054	0.532 ± 0.007
	2%	22.021 ± 7.462	0.090 ± 0.001	0.363 ± 0.028	0.760 ± 0.004	0.289 ± 0.017	0.717 ± 0.010	0.690 ± 0.063	0.543 ± 0.004	0.616 ± 0.056	0.528 ± 0.004
restaurant	0.5%	0.018 ± 0.011	0.156 ± 0.001	0.064 ± 0.011	0.346 ± 0.006	0.044 ± 0.006	0.303 ± 0.004	1.143 ± 0.126	0.887 ± 0.005	1.104 ± 0.114	0.901 ± 0.006
	1%	0.176 ± 0.059	0.156 ± 0.002	0.094 ± 0.018	0.346 ± 0.001	0.069 ± 0.010	0.303 ± 0.001	1.125 ± 0.033	0.890 ± 0.007	1.080 ± 0.006	0.905 ± 0.013
	2%	8.699 ± 6.793	0.158 ± 0.002	0.127 ± 0.013	0.335 ± 0.001	0.098 ± 0.008	0.293 ± 0.002	1.156 ± 0.107	0.888 ± 0.009	1.140 ± 0.120	0.901 ± 0.005
adult	0.5%	0.006 ± 0.001	1.342 ± 0.027	0.273 ± 0.023	7.306 ± 0.517	0.088 ± 0.005	2.090 ± 0.067	4.425 ± 0.436	3.580 ± 0.014	4.150 ± 0.457	3.724 ± 0.008
	1%	0.017 ± 0.001	1.329 ± 0.029	0.412 ± 0.076	7.840 ± 0.630	0.129 ± 0.006	2.128 ± 0.075	4.315 ± 0.030	3.603 ± 0.090	4.041 ± 0.012	3.700 ± 0.014
	2%	0.218 ± 0.018	1.313 ± 0.038	0.613 ± 0.075	7.959 ± 0.593	0.193 ± 0.007	2.051 ± 0.063	4.566 ± 0.527	3.580 ± 0.030	4.246 ± 0.370	3.691 ± 0.015
compas	0.5%	0.012 ± 0.004	0.045 ± 0.002	0.001 ± 0.000	0.011 ± 0.000	0.001 ± 0.000	0.009 ± 0.000	0.581 ± 0.057	0.465 ± 0.005	0.559 ± 0.052	0.474 ± 0.019
	1%	0.067 ± 0.005	0.045 ± 0.002	0.001 ± 0.000	0.011 ± 0.000	0.001 ± 0.000	0.009 ± 0.000	0.567 ± 0.007	0.462 ± 0.005	0.541 ± 0.005	0.464 ± 0.004
	2%	0.296 ± 0.027	0.044 ± 0.002	0.001 ± 0.000	0.010 ± 0.000	0.001 ± 0.000	0.009 ± 0.000	0.588 ± 0.056	0.470 ± 0.032	0.563 ± 0.048	0.462 ± 0.005
bank	0.5%	0.006 ± 0.001	1.082 ± 0.023	0.165 ± 0.056	2.796 ± 0.059	0.079 ± 0.022	1.491 ± 0.009	5.721 ± 0.515	4.606 ± 0.032	5.577 ± 0.483	4.648 ± 0.010
	1%	0.032 ± 0.000	1.076 ± 0.017	0.246 ± 0.035	2.803 ± 0.062	0.129 ± 0.029	1.480 ± 0.009	5.687 ± 0.019	4.579 ± 0.012	5.623 ± 0.066	4.618 ± 0.005
	2%	0.129 ± 0.000	1.063 ± 0.021	0.441 ± 0.062	3.007 ± 0.036	0.214 ± 0.027	1.507 ± 0.008	5.980 ± 0.414	4.551 ± 0.011	5.971 ± 0.466	4.591 ± 0.010

Table 18: Test disparity in statistical parity (SP) and accuracy for the sparse decision trees with depth limit 2.

	DPF		**FairOCT**		**POST_XGB**		**POST_CART**		**CART**		**RSET**	
Dataset	sp	Acc	sp	Acc	sp	Acc	sp	Acc	sp	Acc	sp	Acc
adult	0.015 ± 0.005	0.763 ± 0.004	$\mathbf{0.000 \pm 0.000}$	0.749 ± 0.003	0.081 ± 0.040	0.813 ± 0.012	0.037 ± 0.039	0.765 ± 0.012	0.214 ± 0.031	0.809 ± 0.002	0.055 ± 0.011	0.801 ± 0.002
bank	0.007 ± 0.004	0.895 ± 0.002	$\mathbf{0.000 \pm 0.000}$	0.882 ± 0.002	0.007 ± 0.003	0.899 ± 0.003	0.014 ± 0.005	0.893 ± 0.002	0.012 ± 0.004	0.893 ± 0.002	0.007 ± 0.003	0.893 ± 0.001
compas	0.023 ± 0.017	0.565 ± 0.013	$\mathbf{0.022 \pm 0.015}$	0.553 ± 0.003	0.150 ± 0.021	0.656 ± 0.011	0.165 ± 0.032	0.638 ± 0.012	0.165 ± 0.032	0.638 ± 0.012	0.150 ± 0.031	0.646 ± 0.018
german	0.083 ± 0.056	0.688 ± 0.022	0.058 ± 0.037	0.698 ± 0.016	0.060 ± 0.037	0.732 ± 0.015	0.002 ± 0.003	0.681 ± 0.023	$\mathbf{0.002 \pm 0.003}$	0.681 ± 0.023	0.042 ± 0.026	0.687 ± 0.016
oulad	0.007 ± 0.004	0.682 ± 0.002	$\mathbf{0.001 \pm 0.001}$	0.674 ± 0.004	0.013 ± 0.004	0.689 ± 0.005	0.008 ± 0.006	0.683 ± 0.003	0.010 ± 0.003	0.683 ± 0.003	0.009 ± 0.003	0.683 ± 0.003
stud-mat	0.116 ± 0.095	0.846 ± 0.071	0.116 ± 0.095	0.846 ± 0.071	0.099 ± 0.053	0.878 ± 0.026	0.110 ± 0.069	0.909 ± 0.020	0.095 ± 0.061	0.916 ± 0.030	$\mathbf{0.089 \pm 0.062}$	0.901 ± 0.032
stud-por	0.061 ± 0.035	0.886 ± 0.020	0.061 ± 0.035	0.886 ± 0.020	0.032 ± 0.022	0.891 ± 0.013	0.049 ± 0.035	0.912 ± 0.019	0.045 ± 0.032	0.914 ± 0.018	$\mathbf{0.045 \pm 0.032}$	0.914 ± 0.018

Across all depths, the selected tree from the Rashomon set often achieves fairness and accuracy comparable to that of baseline methods. At depth 3, for instance, the Rashomon tree outperforms all baselines on the bank and stud-por datasets (Table 19) by achieving fairness scores of 0.006 and 0.028, respectively (which are lower than those of the baselines), while still maintaining high accuracy. In cases where the tree selected from the Rashomon set does not outperform the baselines, its fairness score remains within ± 0.1 of the best score. This is also observed at depths 2 (Table 18) and 4 (Table 19), where the Rashomon Set representative matches baseline performance in statistical parity, even if it does not always lead on every dataset.

It is also worth noting that when the Rashomon tree exhibits a slightly higher (i.e., less fair) disparity in statistical parity score compared to another baseline, it usually compensates with improved accuracy. For example, at depth 4, the RSET representative has a statistical parity score of 0.135 compared to DPF's 0.011, yet its accuracy is higher (0.825 versus 0.787). This pattern often occurs across datasets and suggests that our selection method could be further refined to better balance the trade-off between fairness and accuracy. In Section 3.4, we introduced the α parameter to control such trade-off. When $\alpha = 1$, the fairness is valued as much as accuracy. However, this value of α usually leads to a significant reduction in accuracy as $1 - fairness$ generally has a higher magnitude than accuracy. During our experiment, we arbitrarily chose $\alpha = 0.3$ to demonstrate that a representative from the Rashomon Set can compete with baselines that optimize fairness, but we don't claim $\alpha = 0.3$ to be the optimal value. The results of adjusting α are shown in Section D.3.

Similar trends can be found when the models are optimized and evaluated for equalized odds and equal opportunity metrics. Tables 20, 21, 22, 23 have similar results to those that discussed for statistical parity above. At depth = 2, while FairOCT beats RSET in terms of fairness in most datasets, it comes at a cost to accuracy. For example, for compas dataset, while FairOCT's tree has a disparity in Equalized Odds score of 0.008, which is lower than RSET's 0.18, its accuracy of 0.551 is much worse than RSET's 0.643. This again shows that for these datasets at depth 2, $\alpha = 0.3$ might not be capturing the most competitive tree from the RSET to compare with the baseline that optimizes over fairness.

As we increase the depth of decision tree classifiers, FairOCT becomes incredibly inefficient, timing out for all datasets. The training time of RSET and baselines can be found in Appendix D.1. Within the efficient methods, the Rashomon set begins to outperform other baseline methods at depth 4. As shown in Table 21, for compas, german, and oulad, the RSET found the fairest tree with $\alpha = 0.3$, while maintaining high levels of accuracy. Another important note here is that at depth 4, DPF timed out on both the bank and oulad datasets, showing potential inefficiencies when growing deep trees.

Table 24 displays the percentage of trees in the Rashomon set that outperforms all baselines on the test data. We notice that the majority of trees within the Rashomon sets outperform all baselines across multiple datasets at depth 3. From datasets bank, german credit, and oulad, 78%, 56%, and 77% of trees from their respective Rashomon sets perform better than baselines in statistical parity. We also bolded instances in the table where more than 30% of the trees from the Rashomon Set outperform the baselines. We observe that as we increase the depth of trees, the Rashomon set is able to generate superior trees more consistently, with 15/21 evaluations having 30% of the Rashomon set being superior. However note that some of these performances have high variance, therefore limiting statistical significance. Nevertheless, for many datasets the Rashomon set contains trees that are fairer than baselines.

Density plots: For each dataset, depth, and fairness metric, we display the performance of the entire Rashomon set and baseline trees evaluated on the test set (Figures 6-8). On the x-axis, we plot $1 -$ fairness disparity, and on the y-axis, we plot accuracy. The blue regions represent the density of the Rashomon set, averaged over 5 folds, with the number of trees shown on the color bar. The most optimal and fair tree should be located towards the top right of the graph, where we have high accuracy and low disparity.

While for the majority of the density plots, the Rashomon set often encapsulates trees generated by baseline methods, we observe that sometimes the Rashomon set cannot capture the DPF or FairOCT tree. For example, in Figure 6, both adult and compas datasets have DPF and FairOCT trees outside of the blue region.However, these trees also exhibit sacrifice in accuracy, as they perform significantly worse than other baselines. The FairOCT tree on compas dataset has an accuracy of only 0.55, while other trees and the Rashomon set have an accuracy of around 0.65. However, as we mentioned, the

Table 19: Test disparity in statistical parity (SP) and accuracy for the sparse decision trees with the depth limits 3 and 4. FairOCT is omitted as it exceeds the time limit.

	DPF		**POST_XGB**		**POST_CART**		**CART**		**RSET**	
Dataset	sp	Acc	sp	Acc	sp	Acc	sp	Acc	sp	Acc
Depth 3										
adult	**0.008 ± 0.004**	0.784 ± 0.003	0.095 ± 0.055	0.816 ± 0.016	0.039 ± 0.042	0.784 ± 0.012	0.225 ± 0.029	0.817 ± 0.003	0.075 ± 0.045	0.809 ± 0.010
bank	**0.007 ± 0.002**	0.896 ± 0.002	0.007 ± 0.003	0.899 ± 0.003	0.011 ± 0.005	0.893 ± 0.003	0.014 ± 0.005	0.896 ± 0.002	**0.006 ± 0.002**	0.894 ± 0.000
compas	**0.038 ± 0.014**	0.576 ± 0.013	0.156 ± 0.021	0.660 ± 0.005	0.154 ± 0.016	0.661 ± 0.007	0.154 ± 0.016	0.661 ± 0.007	0.161 ± 0.029	0.650 ± 0.010
german	0.082 ± 0.043	0.694 ± 0.023	0.047 ± 0.038	0.733 ± 0.019	0.050 ± 0.032	0.701 ± 0.046	**0.059 ± 0.035**	0.703 ± 0.040	0.084 ± 0.055	0.724 ± 0.012
oulad	0.010 ± 0.004	0.684 ± 0.004	0.012 ± 0.004	0.690 ± 0.005	**0.005 ± 0.002**	0.683 ± 0.003	0.015 ± 0.009	0.682 ± 0.003	0.006 ± 0.005	0.684 ± 0.005
stud-mat	0.125 ± 0.011	0.858 ± 0.038	**0.094 ± 0.054**	0.881 ± 0.026	0.098 ± 0.054	0.899 ± 0.033	0.110 ± 0.063	0.894 ± 0.021	0.121 ± 0.073	0.899 ± 0.035
stud-por	0.048 ± 0.037	0.886 ± 0.019	0.029 ± 0.023	0.889 ± 0.014	0.052 ± 0.017	0.900 ± 0.036	0.046 ± 0.024	0.911 ± 0.024	**0.028 ± 0.021**	0.909 ± 0.024
Depth 4										
adult	**0.011 ± 0.007**	0.787 ± 0.003	0.096 ± 0.055	0.817 ± 0.016	0.138 ± 0.061	0.821 ± 0.018	0.166 ± 0.008	0.833 ± 0.004	0.135 ± 0.046	0.825 ± 0.011
bank	Timeout	Timeout	**0.007 ± 0.003**	0.899 ± 0.003	0.015 ± 0.003	0.895 ± 0.002	0.014 ± 0.002	0.896 ± 0.002	0.008 ± 0.006	0.895 ± 0.003
compas	**0.037 ± 0.010**	0.580 ± 0.014	0.150 ± 0.022	0.657 ± 0.009	0.141 ± 0.016	0.656 ± 0.009	0.151 ± 0.016	0.660 ± 0.006	0.158 ± 0.025	0.646 ± 0.008
german	0.069 ± 0.038	0.700 ± 0.029	**0.056 ± 0.043**	0.734 ± 0.020	0.067 ± 0.034	0.703 ± 0.025	0.069 ± 0.021	0.700 ± 0.027	0.059 ± 0.051	0.698 ± 0.023
oulad	Timeout	Timeout	0.013 ± 0.004	0.689 ± 0.004	**0.004 ± 0.004**	0.683 ± 0.004	0.015 ± 0.010	0.683 ± 0.004	0.012 ± 0.005	0.684 ± 0.003
stud-mat	0.106 ± 0.067	0.851 ± 0.022	**0.094 ± 0.054**	0.881 ± 0.026	0.127 ± 0.043	0.884 ± 0.043	0.118 ± 0.056	0.873 ± 0.033	0.138 ± 0.045	0.914 ± 0.037
stud-por	0.044 ± 0.041	0.889 ± 0.026	**0.022 ± 0.020**	0.889 ± 0.014	0.058 ± 0.024	0.895 ± 0.034	0.047 ± 0.018	0.897 ± 0.027	0.040 ± 0.026	0.925 ± 0.021

Table 20: Test disparity in equalized odds (EODDS) and accuracy for the sparse decision trees with the depth limit 2.

	FAIROCT		**POST_XGB**		**POST_CART**		**CART**		**RSET**	
Dataset	eodds	Accuracy	eodds	Accuracy	eodds	Accuracy	eodds	Accuracy	eodds	Accuracy
adult	**0.004 ± 0.002**	0.749 ± 0.003	0.071 ± 0.038	0.831 ± 0.013	0.091 ± 0.046	0.791 ± 0.023	0.130 ± 0.024	0.809 ± 0.002	0.044 ± 0.015	0.800 ± 0.003
bank	**0.000 ± 0.000**	0.882 ± 0.002	0.022 ± 0.005	0.898 ± 0.004	0.035 ± 0.016	0.893 ± 0.002	0.037 ± 0.014	0.893 ± 0.002	0.026 ± 0.012	0.893 ± 0.004
compas	**0.008 ± 0.002**	0.551 ± 0.013	0.165 ± 0.076	0.641 ± 0.029	0.131 ± 0.054	0.617 ± 0.014	0.191 ± 0.027	0.638 ± 0.012	0.180 ± 0.042	0.643 ± 0.015
german	0.008 ± 0.004	0.688 ± 0.026	0.105 ± 0.073	0.725 ± 0.031	**0.000 ± 0.000**	0.692 ± 0.024	0.010 ± 0.012	0.681 ± 0.023	0.057 ± 0.034	0.689 ± 0.021
oulad	**0.000 ± 0.000**	0.675 ± 0.004	0.026 ± 0.014	0.690 ± 0.003	**0.000 ± 0.000**	0.675 ± 0.004	0.017 ± 0.005	0.683 ± 0.003	0.020 ± 0.007	0.680 ± 0.004
stud-mat	**0.007 ± 0.001**	0.896 ± 0.052	0.132 ± 0.087	0.871 ± 0.017	0.090 ± 0.062	0.914 ± 0.029	0.083 ± 0.062	0.916 ± 0.030	0.140 ± 0.101	0.906 ± 0.034
stud-por	**0.006 ± 0.003**	0.878 ± 0.025	0.277 ± 0.156	0.889 ± 0.014	0.201 ± 0.173	0.885 ± 0.044	0.223 ± 0.161	0.914 ± 0.018	0.154 ± 0.150	0.902 ± 0.015

Table 21: Test disparity in equalized odds (EODDS) and accuracy for the sparse decision trees with the depth limits 3 and 4. FairOCT is omitted as it exceeds the time limit.

	POST_XGB		POST_CART		CART		RSET	
Dataset	eodds	Accuracy	eodds	Accuracy	eodds	Accuracy	eodds	Accuracy
Depth 3								
adult	0.074 ± 0.034	0.831 ± 0.013	**0.072 ± 0.055**	0.799 ± 0.014	0.132 ± 0.025	0.817 ± 0.003	0.073 ± 0.008	0.825 ± 0.010
bank	**0.021 ± 0.006**	0.898 ± 0.004	0.024 ± 0.014	0.892 ± 0.002	0.038 ± 0.019	0.896 ± 0.002	0.032 ± 0.017	0.895 ± 0.002
compas-recid	0.192 ± 0.028	0.657 ± 0.010	**0.169 ± 0.046**	0.642 ± 0.015	0.199 ± 0.034	0.661 ± 0.007	0.193 ± 0.043	0.649 ± 0.012
german-credit	0.116 ± 0.068	0.722 ± 0.029	**0.030 ± 0.037**	0.692 ± 0.024	0.116 ± 0.052	0.703 ± 0.040	0.122 ± 0.085	0.724 ± 0.005
oulad	0.024 ± 0.014	0.689 ± 0.004	**0.000 ± 0.000**	0.675 ± 0.004	0.028 ± 0.016	0.682 ± 0.003	0.018 ± 0.010	0.685 ± 0.003
student-mat	0.137 ± 0.059	0.881 ± 0.031	**0.109 ± 0.075**	0.906 ± 0.036	0.147 ± 0.073	0.894 ± 0.021	0.160 ± 0.065	0.899 ± 0.036
student-por	0.252 ± 0.134	0.883 ± 0.017	**0.204 ± 0.148**	0.908 ± 0.026	0.246 ± 0.132	0.911 ± 0.024	0.250 ± 0.153	0.915 ± 0.030
Depth 4								
adult	**0.073 ± 0.035**	0.831 ± 0.013	0.092 ± 0.039	0.829 ± 0.002	0.074 ± 0.008	0.833 ± 0.004	0.075 ± 0.007	0.831 ± 0.004
bank	0.021 ± 0.006	0.898 ± 0.004	**0.013 ± 0.009**	0.891 ± 0.004	0.024 ± 0.011	0.896 ± 0.002	0.030 ± 0.010	0.894 ± 0.001
compas-recid	0.192 ± 0.028	0.656 ± 0.010	**0.183 ± 0.037**	0.656 ± 0.014	0.193 ± 0.026	0.660 ± 0.006	**0.183 ± 0.037**	0.649 ± 0.011
german-credit	0.228 ± 0.126	0.739 ± 0.029	0.096 ± 0.045	0.678 ± 0.020	0.084 ± 0.035	0.680 ± 0.028	**0.074 ± 0.039**	0.718 ± 0.022
oulad	0.025 ± 0.014	0.689 ± 0.004	0.029 ± 0.014	0.673 ± 0.006	0.032 ± 0.018	0.683 ± 0.004	**0.019 ± 0.011**	0.684 ± 0.002
student-mat	0.132 ± 0.059	0.894 ± 0.017	**0.081 ± 0.037**	0.881 ± 0.058	0.161 ± 0.058	0.861 ± 0.018	0.118 ± 0.055	0.906 ± 0.033
student-por	0.286 ± 0.111	0.889 ± 0.030	0.171 ± 0.096	0.892 ± 0.011	**0.111 ± 0.088**	0.886 ± 0.023	0.169 ± 0.146	0.909 ± 0.018

Table 22: Test disparity in equalized opportunities (EOPP) and accuracy for the sparse decision trees with the depth limit 2.

	FAIROCT		POST_XGB		POST_CART		CART		RSET	
Dataset	eopp	Accuracy	eopp	Accuracy	eopp	Accuracy	eopp	Accuracy	eopp	Accuracy
adult	**0.003 ± 0.002**	0.749 ± 0.003	0.033 ± 0.015	0.833 ± 0.004	0.076 ± 0.039	0.791 ± 0.023	0.060 ± 0.058	0.809 ± 0.002	0.042 ± 0.031	0.810 ± 0.006
bank	**0.000 ± 0.000**	0.882 ± 0.002	0.018 ± 0.009	0.900 ± 0.002	0.035 ± 0.016	0.893 ± 0.002	0.037 ± 0.014	0.893 ± 0.002	0.019 ± 0.014	0.890 ± 0.003
compas	**0.017 ± 0.007**	0.604 ± 0.023	0.074 ± 0.027	0.657 ± 0.009	0.074 ± 0.032	0.627 ± 0.009	0.107 ± 0.042	0.638 ± 0.012	0.091 ± 0.035	0.651 ± 0.013
german	0.102 ± 0.046	0.690 ± 0.027	0.052 ± 0.042	0.731 ± 0.031	**0.000 ± 0.000**	0.692 ± 0.024	0.002 ± 0.003	0.681 ± 0.023	0.042 ± 0.034	0.683 ± 0.022
oulad	**0.000 ± 0.000**	0.675 ± 0.005	0.006 ± 0.005	0.690 ± 0.003	**0.000 ± 0.000**	0.675 ± 0.004	0.010 ± 0.004	0.683 ± 0.003	0.009 ± 0.005	0.682 ± 0.005
stud-mat	0.125 ± 0.051	0.886 ± 0.049	0.081 ± 0.044	0.876 ± 0.026	0.108 ± 0.050	0.904 ± 0.021	**0.073 ± 0.066**	0.916 ± 0.030	0.079 ± 0.069	0.906 ± 0.034
stud-por	0.205 ± 0.096	0.902 ± 0.034	0.045 ± 0.007	0.878 ± 0.015	0.029 ± 0.025	0.900 ± 0.032	**0.027 ± 0.022**	0.914 ± 0.018	**0.027 ± 0.022**	0.914 ± 0.018

Rashomon set contains fair models for the majority of the datasets. For example, in Figure 7, across all three metrics, all baselines are within the Rashomon set. Similar can be observed in Figure 8.

D.1 Training Time

Table 25 shows the average training time of TreeFARMS (REST), DPF, and FairOCT. We discover that TreeFARMS training time is comparable to that of DPF and significantly shorter than FairOCT. At deeper depths, training time varies significantly across datasets: sometimes DPF times out, while other times TreeFARMS takes longer to train. In our experiment, we set a time limit of one hour, and therefore any training that exceeds this limit is considered as timeout. Therefore, the Rashomon set for german credit should be considered as a "time out" since the average training time is 5000 seconds. This may be due to system latency or variations in computational load. FairOCT consistently shows extremely long training times across all datasets. For example, on the adult dataset, FairOCT takes over 12,500 seconds, whereas both DPF and TreeFARMS require less than a second. This huge gap suggests that, even at shallow depths, FairOCT cannot find fair trees efficiently.

D.2 Pareto Frontier Visualization

Another way to evaluate the Rashomon set against baselines is to compare the set of optimal solutions. We generated the Pareto frontiers of the Rashomon set on each training dataset we used in fairness evaluations and compared them to those of DPF (see Figures 9-10). In some cases, the two Pareto frontiers differ. When this happens, the RSET Pareto frontier usually includes solutions that achieve high accuracy but lower fairness. For example, on the adult dataset, the RSET Pareto frontier reaches an accuracy of 0.82, compared to DPF's 0.77. But, in terms of statistical parity, DPF's Pareto frontier includes a perfectly fair solution (score of 1), while RSET's Pareto frontier reaches a score of 0.975.

Table 23: Test disparity in equalized opportunities (EOPP) and accuracy for the sparse decision trees with the depth limits 3 and 4. FairOCT is omitted as it exceeds the time limit.

	POST_XGB		POST_CART		CART		RSET	
Dataset	eopp	Accuracy	eopp	Accuracy	eopp	Accuracy	eopp	Accuracy
Depth 3								
adult	**0.032 ± 0.015**	0.834 ± 0.004	0.076 ± 0.039	0.809 ± 0.013	0.079 ± 0.049	0.817 ± 0.003	0.052 ± 0.023	0.826 ± 0.004
bank	**0.018 ± 0.009**	0.900 ± 0.002	0.020 ± 0.016	0.892 ± 0.002	0.038 ± 0.019	0.896 ± 0.002	0.038 ± 0.008	0.895 ± 0.002
compas-recid	**0.074 ± 0.028**	0.656 ± 0.011	0.110 ± 0.046	0.646 ± 0.009	0.075 ± 0.024	0.661 ± 0.007	0.092 ± 0.036	0.652 ± 0.012
german-credit	0.056 ± 0.041	0.731 ± 0.032	**0.015 ± 0.016**	0.690 ± 0.029	0.056 ± 0.051	0.703 ± 0.040	0.054 ± 0.038	0.721 ± 0.010
oulad	0.006 ± 0.005	0.690 ± 0.003	**0.003 ± 0.004**	0.675 ± 0.005	0.012 ± 0.004	0.682 ± 0.003	0.008 ± 0.008	0.686 ± 0.003
student-mat	**0.081 ± 0.044**	0.886 ± 0.031	0.088 ± 0.062	0.906 ± 0.036	0.094 ± 0.070	0.894 ± 0.021	0.092 ± 0.073	0.904 ± 0.028
student-por	0.044 ± 0.011	0.889 ± 0.023	**0.027 ± 0.022**	0.914 ± 0.018	0.029 ± 0.027	0.911 ± 0.024	0.034 ± 0.021	0.909 ± 0.023
Depth 4								
adult	**0.032 ± 0.015**	0.833 ± 0.004	0.074 ± 0.053	0.828 ± 0.003	0.053 ± 0.025	0.833 ± 0.004	0.036 ± 0.030	0.825 ± 0.005
bank	0.018 ± 0.008	0.900 ± 0.002	**0.012 ± 0.010**	0.893 ± 0.002	0.023 ± 0.012	0.896 ± 0.002	0.026 ± 0.008	0.894 ± 0.001
compas-recid	0.075 ± 0.027	0.658 ± 0.008	0.075 ± 0.027	0.662 ± 0.006	**0.074 ± 0.022**	0.660 ± 0.006	0.095 ± 0.033	0.653 ± 0.012
german-credit	0.050 ± 0.020	0.741 ± 0.032	0.067 ± 0.043	0.684 ± 0.024	**0.038 ± 0.022**	0.680 ± 0.028	0.058 ± 0.040	0.718 ± 0.022
oulad	**0.005 ± 0.005**	0.690 ± 0.003	0.010 ± 0.008	0.675 ± 0.005	0.011 ± 0.007	0.683 ± 0.004	0.010 ± 0.007	0.684 ± 0.002
student-mat	0.114 ± 0.072	0.861 ± 0.021	0.091 ± 0.061	0.894 ± 0.046	0.096 ± 0.040	0.861 ± 0.018	**0.090 ± 0.061**	0.906 ± 0.033
student-por	0.042 ± 0.026	0.889 ± 0.022	0.035 ± 0.022	0.898 ± 0.012	0.047 ± 0.040	0.886 ± 0.023	**0.027 ± 0.022**	0.906 ± 0.017

Table 24: Percentage of trees in Rashomon set that have better test fairness than the best baselines.

Depth 2			
Dataset	SP	EOPP	EODDS
adult	0.00% ± 0.00%	3.43% ± 3.33%	0.57% ± 0.70%
bank	0.77% ± 0.30%	1.66% ± 0.68%	0.77% ± 0.30%
compas-recid	0.00% ± 0.00%	0.00% ± 0.00%	0.00% ± 0.00%
german-credit	9.85% ± 2.45%	7.48% ± 2.22%	3.45% ± 2.93%
oulad	16.68% ± 6.13%	5.05% ± 2.67%	3.91% ± 2.47%
student-mat	**41.92% ± 45.54%**	**58.01% ± 42.70%**	0.00% ± 0.00%
student-por	11.13% ± 15.65%	25.45% ± 37.87%	0.00% ± 0.00%
Depth 3			
Dataset	SP	EOPP	EODDS
adult	0.00% ± 0.00%	14.27% ± 5.51%	10.17% ± 8.38%
bank	**78.37% ± 15.83%**	**68.41% ± 15.79%**	**71.63% ± 15.83%**
compas-recid	0.00% ± 0.00%	**39.26% ± 35.97%**	21.29% ± 31.05%
german-credit	**55.56% ± 25.08%**	23.30% ± 10.61%	13.40% ± 7.32%
oulad	**77.07% ± 8.37%**	**59.74% ± 6.82%**	7.40% ± 1.23%
student-mat	**43.28% ± 36.47%**	**56.92% ± 36.49%**	**44.98% ± 22.40%**
student-por	**35.42% ± 35.27%**	**53.74% ± 35.54%**	**57.12% ± 39.21%**
Depth 4			
Dataset	SP	EOPP	EODDS
adult	0.00% ± 0.00%	13.13% ± 10.67%	15.03% ± 15.99%
bank	**65.28% ± 21.96%**	**38.31% ± 16.34%**	**39.49% ± 16.71%**
compas-recid	0.00% ± 0.00%	**36.95% ± 37.95%**	26.47% ± 35.93%
german-credit	28.85% ± 39.49%	**30.79% ± 30.80%**	**31.55% ± 36.20%**
oulad	**84.22% ± 8.37%**	**75.80% ± 9.33%**	**89.21% ± 6.75%**
student-mat	**40.00% ± 48.99%**	**60.00% ± 48.99%**	**32.89% ± 33.12%**
student-por	**43.70% ± 41.99%**	**58.95% ± 37.77%**	**41.23% ± 42.31%**

D.3 Fairness Selection Criteria for the Trees in the Rashomon Set

As discussed previously in Section 3.4, we introduce a parameter α as a way to control the trade-off between accuracy and fairness, as a way to aid in selecting a tree from the Rashomon set. Depending on the value of this parameter α, practitioners can end up with very different trees in terms of fairness and potentially accuracy. In Figures 11-12, we report the performance of selected trees from the Rashomon set of sparse decision trees of depth 4 for different values of $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 1\}$. Overall, for some datasets we observe a negative correlation trend between accuracy and $1-$fairness score when varying the α parameter, indicating the existence of a trade-off between these two metrics.

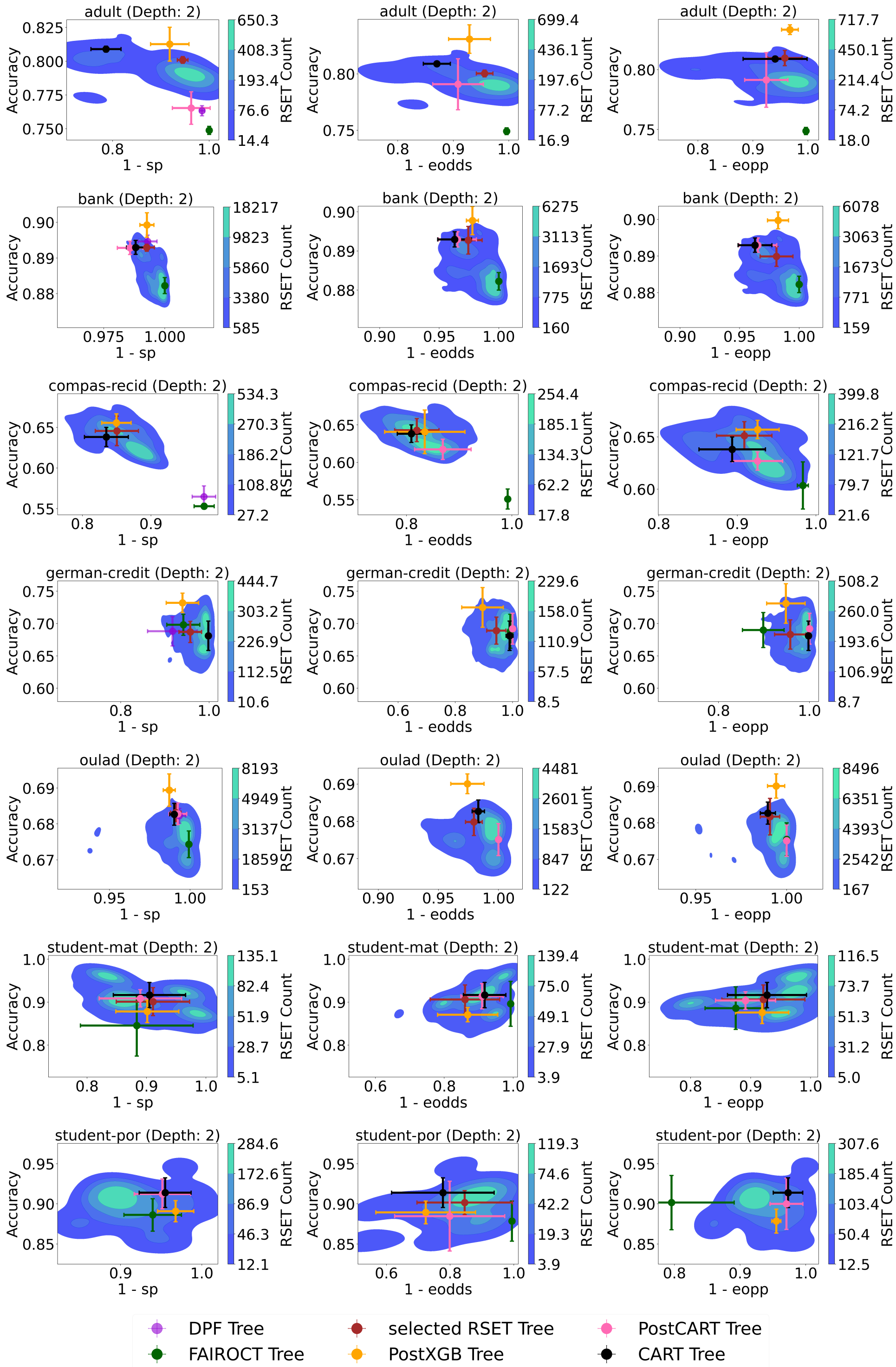


Figure 6: Comparison of test accuracy and fairness between trees in the Rashomon set (blue density contours) and baselines (dots with error bars) at depth 2.

This is the most evident for adult dataset evaluated for statistical parity or for compas dataset when evaluated on equal opportunity.

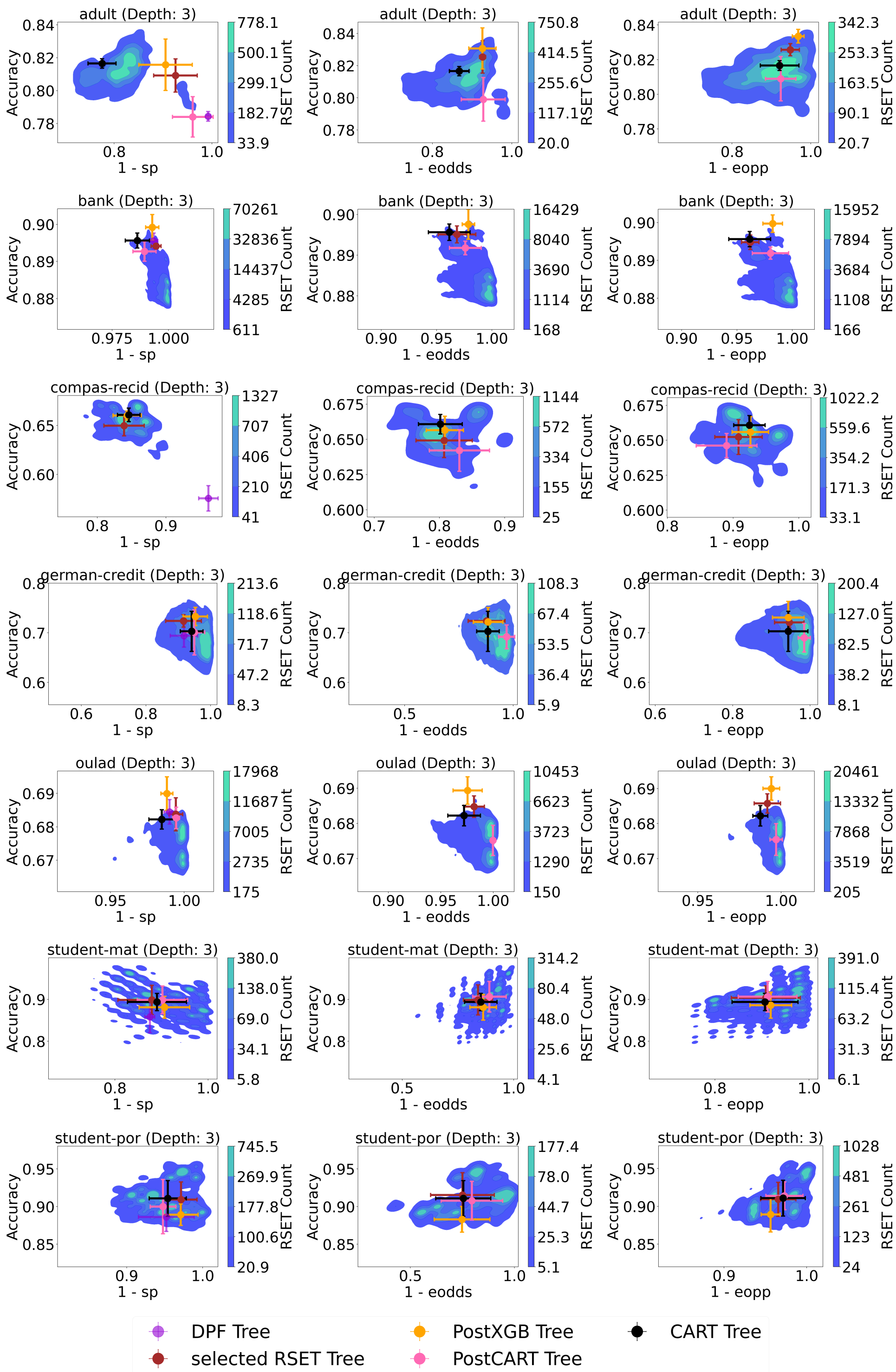


Figure 7: Comparison of test accuracy and fairness between trees in the Rashomon set (blue density contours) and baselines (dots with error bars) at depth 3.

For other datasets, this trade-off is not present. For example, in Figure 12, the value $\alpha = 0.4$ clearly selects the most accurate and fair tree for stud-por dataset. This shows that the structure of the dataset could potentially influence the selection method of a fair and optimal representative from

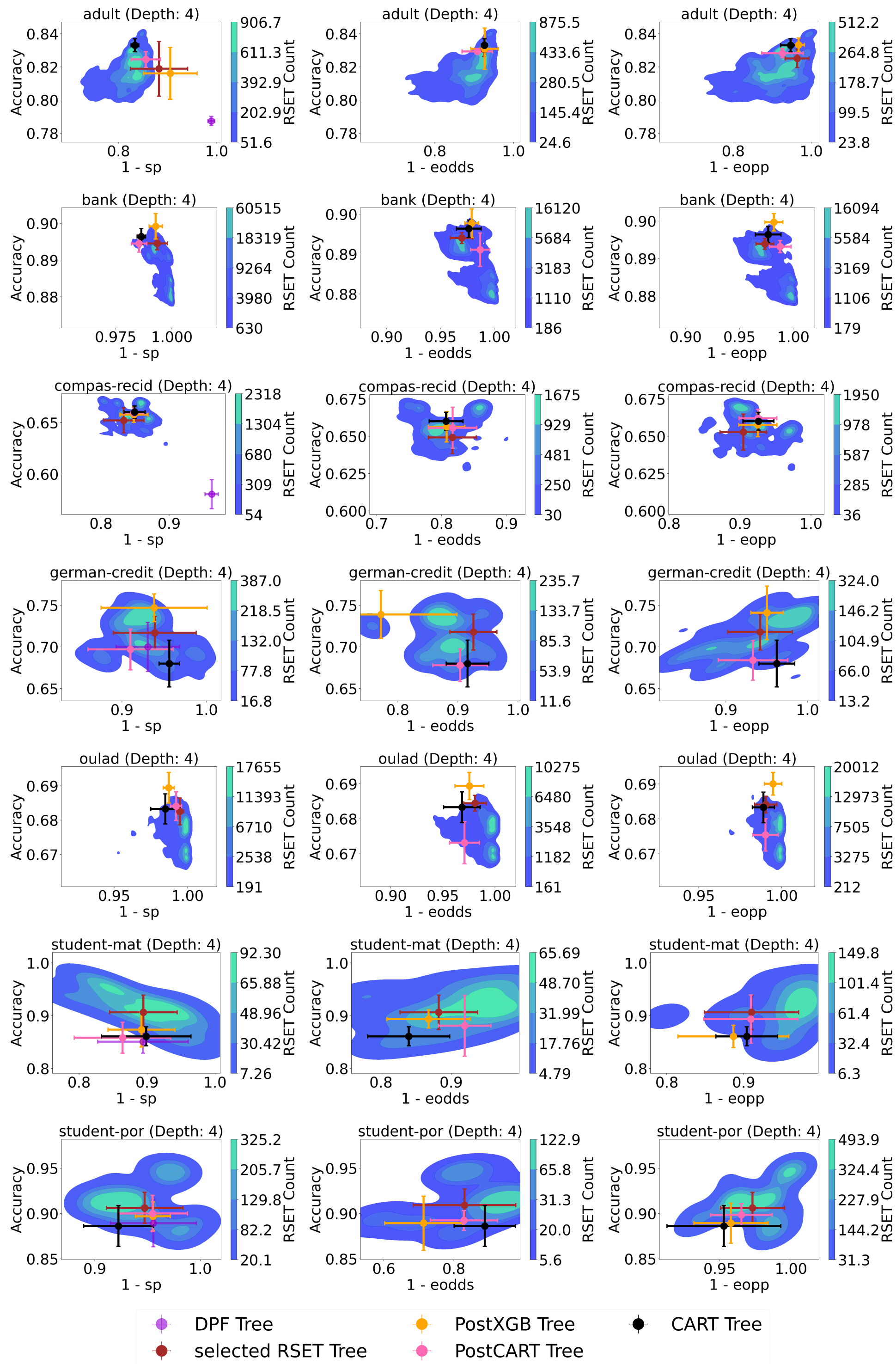


Figure 8: Comparison of test accuracy and fairness between trees in the Rashomon set (blue density contours) and baselines (dots with error bars) at depth 4.

the Rashomon set. As a reminder, in Section 4.1, we presented results when $\alpha = 0.3$, but this value might not be optimal for all datasets, which we leave for future studies.

Table 25: Training time comparison (in seconds) across datasets at different depths.

Dataset	DPF Training Time	RSET Training Time	FairOCT Training Time
Depth 2			
adult	0.058 ± 0.001	0.434 ± 0.011	12542.740 ± 18.165
bank	0.129 ± 0.010	1.573 ± 0.015	12473.707 ± 159.017
compas-recid	0.016 ± 0.011	0.037 ± 0.001	10736.319 ± 201.201
german-credit	0.035 ± 0.001	0.274 ± 0.031	10823.109 ± 3.372
oulad	0.150 ± 0.012	0.767 ± 0.005	11640.883 ± 33.762
student-mat	0.014 ± 0.002	0.059 ± 0.005	1433.153 ± 417.020
student-por	0.016 ± 0.010	0.082 ± 0.005	4044.723 ± 971.958
Depth 3			
Dataset	DPF Training Time	RSET Training Time	
adult	0.231 ± 0.009	1.496 ± 0.038	
bank	1.413 ± 0.008	35.131 ± 1.656	
compas-recid	0.018 ± 0.000	0.065 ± 0.000	
german-credit	1.271 ± 0.258	40.188 ± 4.500	
oulad	1.539 ± 0.018	16.117 ± 0.195	
student-mat	0.205 ± 0.018	17.597 ± 3.811	
student-por	0.250 ± 0.005	9.302 ± 0.776	
Depth 4			
Dataset	DPF Training Time	RSET Training Time	
adult	653.557 ± 15.103	11.328 ± 0.079	
bank	Time out	918.890 ± 12.836	
compas-recid	3.324 ± 0.979	0.192 ± 0.015	
german-credit	234.161 ± 109.967	5328.995 ± 365.490	
oulad	Time out	409.511 ± 11.556	
student-mat	10.646 ± 0.667	2428.415 ± 609.305	
student-por	14.715 ± 0.993	1464.381 ± 185.139	

D.4 Sparsity Plots

In order to check if there is a connection between sparsity and fairness, in Figure 13, we visualize the number of leaves (sparsity) in the trees of the Rashomon set (x-axis) and their corresponding fairness value (y-axis). As the number of leaves increases, so does the span of possible fairness values; however, other than that, we didn't observe any significant trend across datasets and fairness metrics. For example, if we consider the bank dataset evaluated on statistical parity, trees with both fewer leaves and more leaves can achieve similar fairness performance.

E Additional Hypothesis Spaces: Random Forest, FasterRisk

To support our findings with other hypothesis spaces, we conducted additional experiments on random forests for both fairness and robustness. To approximate the Rashomon set in this setting, we trained 100 random forests with different random seeds. As baselines, we used PostRF [98] for fairness (a post-hoc editing method) and GROOT-RF [88] for robustness (also a post-hoc method). We observed that this approximate Rashomon set tends to contain models that outperform the baselines in terms of both fairness and robustness.

Tables 26 and 27 show fairness and robustness comparisons for the random forest model class, respectively. "X Win Rate" refers to the proportion of models in the Rashomon set that outperform the baseline in metric X (e.g., accuracy, fairness, or robustness). "Joint Win Rate" denotes the proportion of models that outperform the baseline in both accuracy and the trustworthiness metric. Results are reported over 5 folds, with mean and standard deviation.

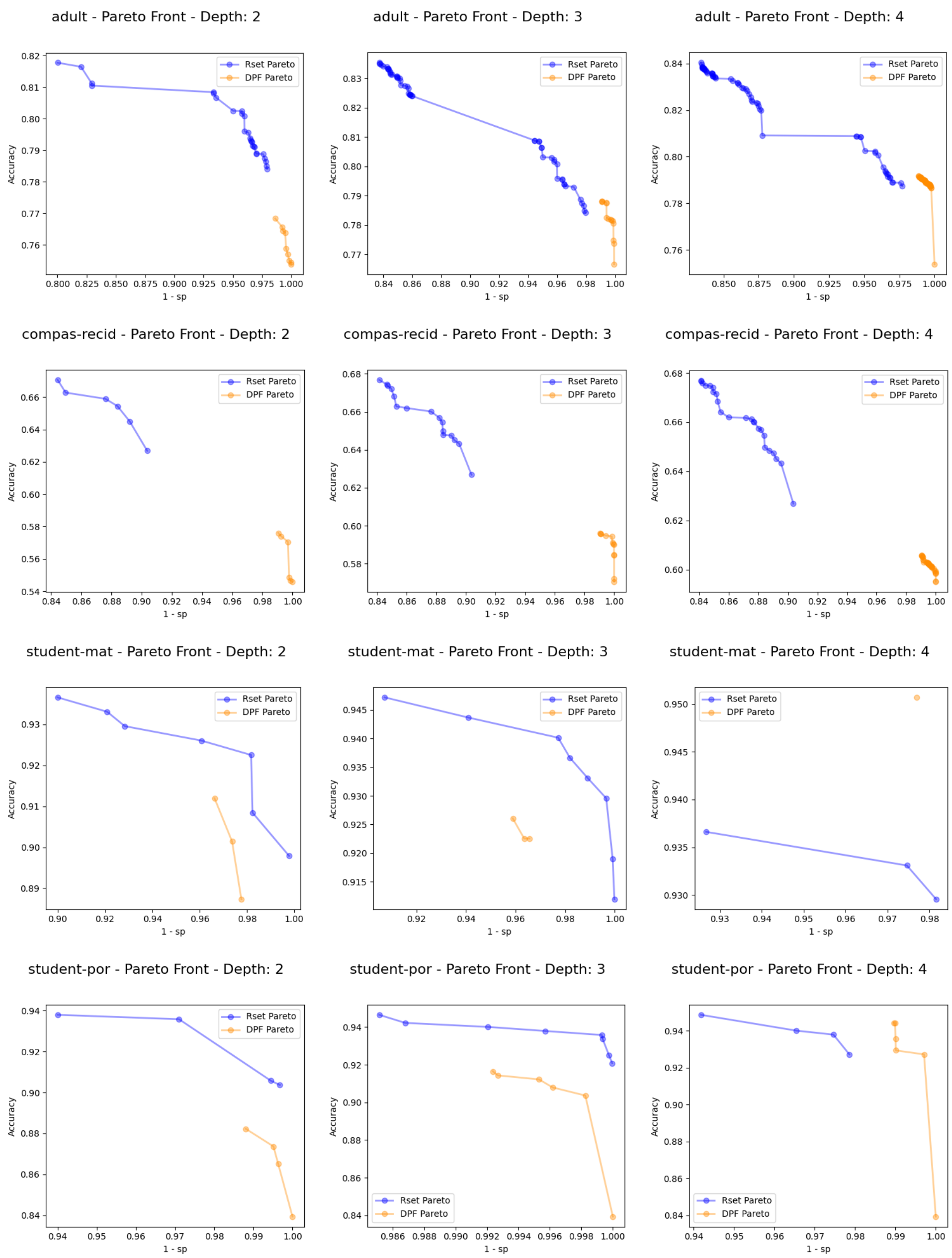


Figure 9: Pareto front: DPF vs. RSET for adult, compas, student-mat and student-por.

We note that in the california-houses dataset, we were unable to find a fair model in the approximated Rashomon set. At this point, we cannot conclude whether this is due to limitations in our approximation method or because the baseline (PostRF) is a post-hoc method that may not correspond to any near-optimal model in the Rashomon set. If this experiment had been conducted in the sparse decision tree setting, we would have had a definitive answer due to the ability to enumerate the entire Rashomon set and the availability of fairness-optimal algorithms.

We can also extend our framework to sparse linear models. However, to the best of our knowledge, there is no method that can exactly enumerate the Rashomon set for linear classifiers; only approximators are available. For example, we use the FasterRisk algorithm [65] for sparse linear models and report the fairness results in Table 28.

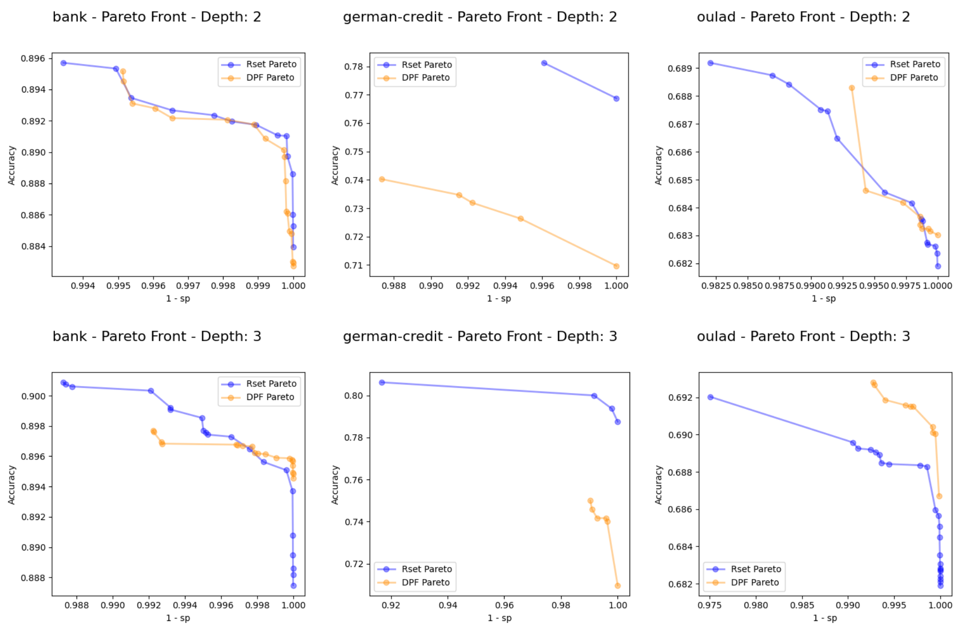


Figure 10: Pareto front: DPF vs. RSET for bank, german-credit, and oulad.

Table 26: Fairness comparison between the Random Forest Rashomon set constructed with different random seeds and PostRF. (Avg ± Std)

Dataset	Joint Win Rate	Accuracy Win Rate	SP Win Rate
adult	0.08 ± 0.11	0.60 ± 0.21	0.36 ± 0.33
california-houses	0.00 ± 0.00	0.99 ± 0.02	0.00 ± 0.00
default-credit	0.19 ± 0.15	0.43 ± 0.09	0.46 ± 0.28
diabetes-130US	0.44 ± 0.24	0.68 ± 0.18	0.59 ± 0.29

We also observe that for california-houses we couldn't outperform the baseline. The limitation of the approximated Rashomon set makes empirical analysis less conclusive than in our current setting with decision trees, where the Rashomon set can be exactly enumerated.

F Evaluation of Selected Trees Across Multiple Criteria

Setup: As mentioned in Section 4.5, we are interested in evaluating the performance of decision trees across various metrics. To achieve this, we consider five datasets: adult, bank, compas, german-credit, and oulad. These datasets are binarized following the procedure described in Section 3.4. We train TreeFARMS after removing the sensitive feature, as required for the evaluation of fairness metrics. For robustness and privacy considerations, these omitted features do not interact with the framework.

We highlight a few discrepancies when evaluating robustness. Specifically, for adversarial robustness, we observe that the attack mechanism does not perform effectively on binary datasets. Since the attack distance is computed using the infinity norm, this can only take values of 0 or 1 in the context of binary datasets. Allowing a meaningful attack strength will result in a 0% accuracy as it can perturb every feature freely, so we remove adversarial robustness from comparison. To assess stability, we apply a metric with a stronger mean of 0.7, which translates to an expected perturbation rate of $\frac{1}{0.7} - 1 \approx 43\%$. This allows certain perturbations to perturb binary values from 0 to 0.5, consequently affecting the model's predictions.

Result: Figure 14 presents a comprehensive comparison of six trees within the Rashomon set across five datasets, evaluated using six different metrics discussed in this paper. For fairness and privacy metrics, we compute $1 -$ score to provide an intuitive visualization, where taller bars indicate better metric performance. Since all trees belong to the Rashomon set, they achieve comparable

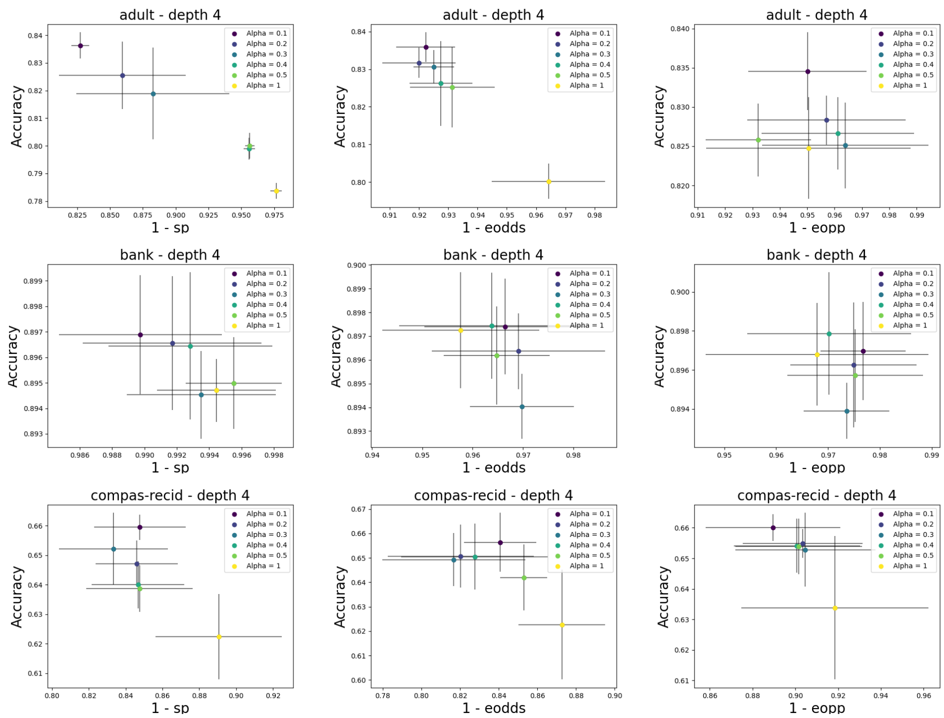


Figure 11: Comparison of test accuracy and fairness between selected trees in the Rashomon set at depth 4 with different alpha values for adult, bank, and compas.

Table 27: Robustness comparison between the Random Forest Rashomon set and baseline. (Avg ± Std)

Dataset	Joint Win Rate	Accuracy Win Rate	Adv. Acc. Win Rate
adult	0.16 ± 0.10	1.00 ± 0.00	0.16 ± 0.10
california-houses	0.63 ± 0.24	1.00 ± 0.00	0.63 ± 0.24
default-credit	0.49 ± 0.24	0.75 ± 0.33	0.51 ± 0.22
diabetes-130US	0.01 ± 0.02	1.00 ± 0.00	0.01 ± 0.02

performance based on the test accuracy. However, we observe a sparsity-accuracy trade-off within the trees with minimum and maximum number leaves on datasets such as adult and compas. In terms of stability, trees with fewer leaves tend to be the most stable models, as expected due to their sparsity.

For statistical (demographic) parity, we find that the selected fair tree performs well on multiple datasets (for example, see german-credit). Interestingly, the tree selected for equalized odds (EODD) also shows strong performance on statistical parity, the opposite case is also sometime true. We also observe that the tree with minimum leaves (RSET_min) can be fair, but this observation is not consistent across all datasets. Finally, regarding privacy metrics, we observe that most trees perform similarly around 50%. As discussed in Section 4.1, these attacks are generally ineffective against tree-based models.

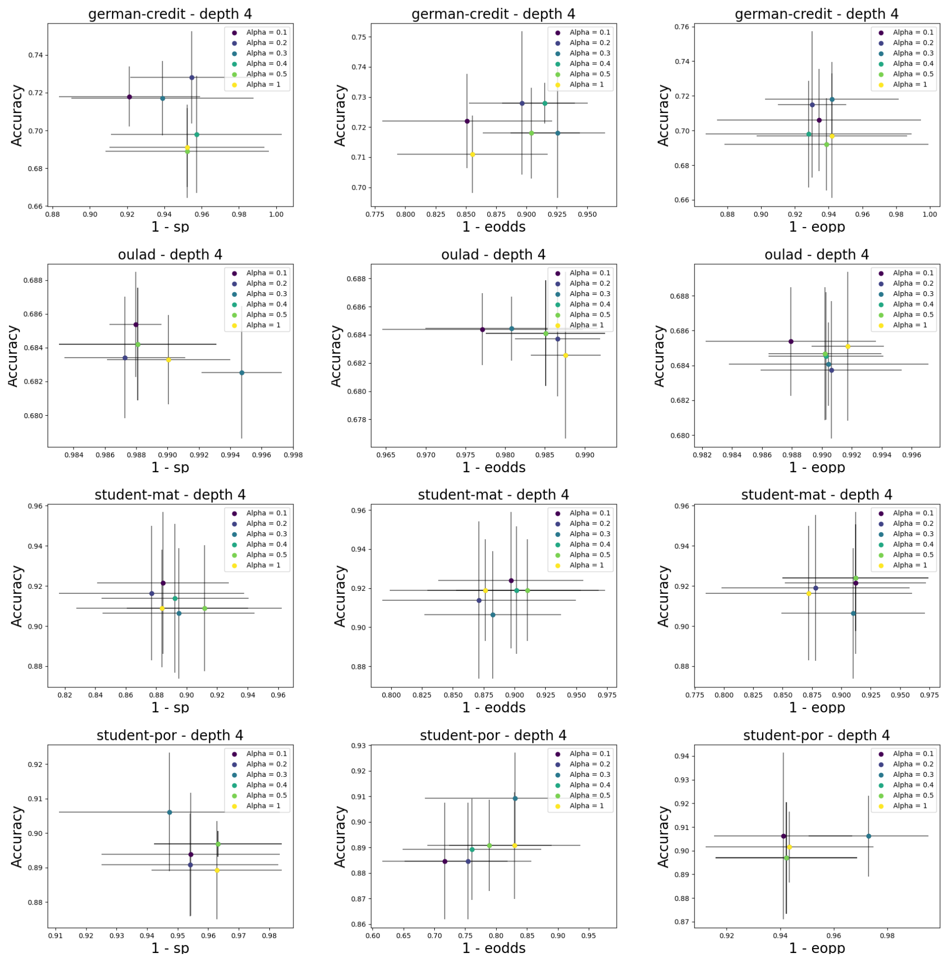


Figure 12: Comparison of test accuracy and fairness between selected trees in the Rashomon set at depth 4 with different alpha values for german-credit, oulad, student-mat, and student-por.

Table 28: Fairness results using the FasterRisk Algorithm. (Avg ± Std)

Dataset	**Joint Win Rate**	**Accuracy Win Rate**	**SP Win Rate**
california-houses	0.00 ± 0.00	0.75 ± 0.00	0.00 ± 0.00
default-credit	0.07 ± 0.12	0.12 ± 0.18	0.75 ± 0.26
diabetes-130US	0.08 ± 0.16	0.13 ± 0.15	0.48 ± 0.27
german-credit	0.01 ± 0.02	0.04 ± 0.06	0.51 ± 0.38

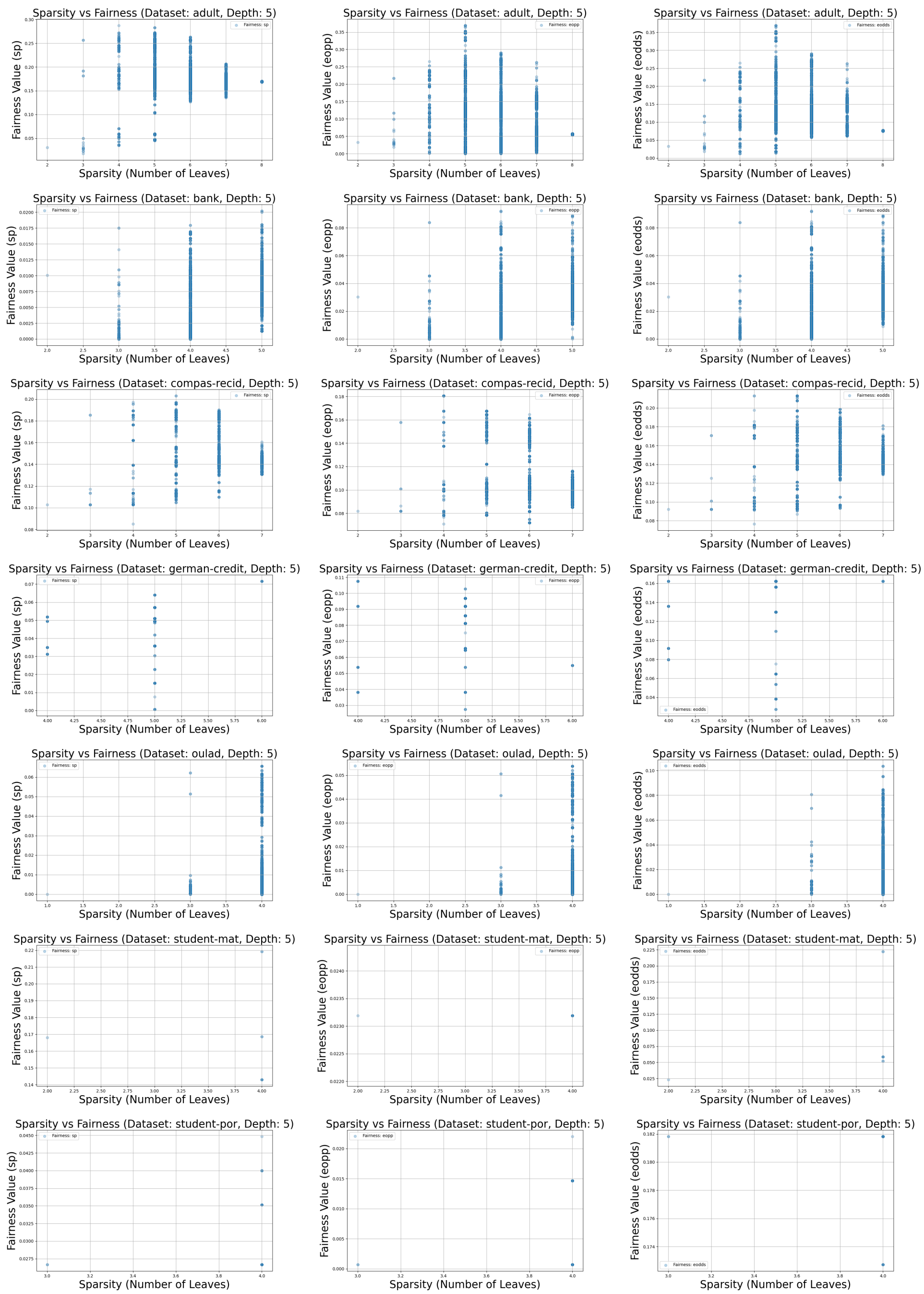


Figure 13: Sparsity plots for sparse decision trees with depth limit 4 evaluated on different fairness metrics.

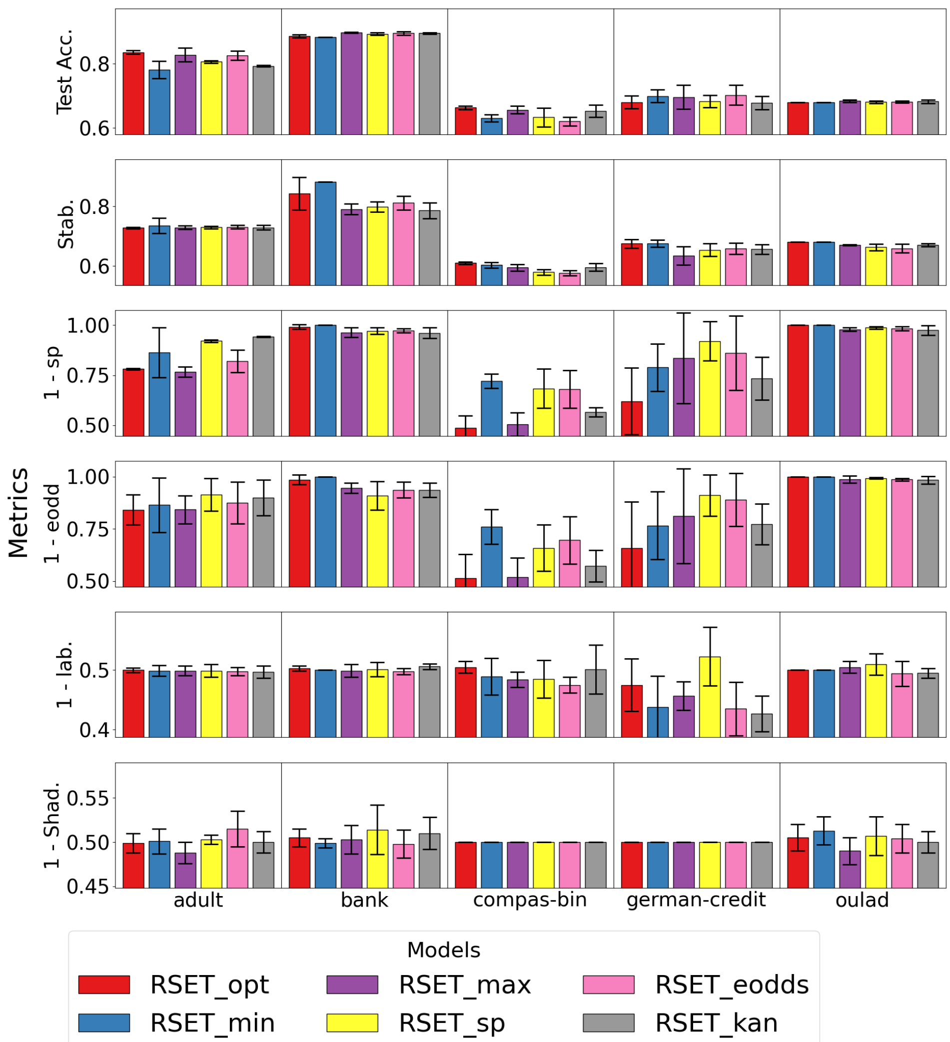


Figure 14: Bar plots of metric accuracy on multiple dataset. Metrics from top to bottom are Test Accuracy. Stability, 1 - Statistical Parity, 1 - equalized odds, 1 - label only MI attack, and 1 - shadow model MI attack.