DT-sampler: A SAT-based Decision Tree Ensemble

Xiaotian Xue[1] Chao Huang[1*] Koji Tsuda[1,2,3†] Diptesh Das[1‡]

[1] Department of Computational Biology and Medical Sciences, The University of Tokyo, Japan
[2] RIKEN Center for Advanced Intelligence Project, Japan
[3] Center for Basic Research on Materials, National Institute for Materials Science, Japan

ABSTRACT

Interpretable (or explainable) machine learning models, such as decision trees, play a crucial role in the context of trustworthy AI. However, finding optimal decision trees (i.e., minimum size and maximum accuracy trees) is not a simple task and remains an active area of research. While a single decision tree has limited expressivity, using an ensemble of decision trees can effectively capture the complex structures found in many real-world applications. Many existing tree ensemble methods are greedy and suboptimal, and often suffer from randomness in the tree generation process. In this paper, we introduce DT-sampler, a SAT-based decision tree ensemble which allows explicit control over both the size and accuracy of the sampled trees. We developed a novel SAT-based encoding method that utilizes only branch nodes, resulting in a compact representation of decision tree space. Additionally, standard point predictions made using decision tree ensembles do not offer any statistical guarantee over miscoverage rate. We employ conformal prediction (CP), a distribution-free statistical framework which provides a valid finite-sample coverage guarantee, to demonstrate that DT-sampler is statistically more efficient and produces stable results when compared with random forest classifier. We demonstrate the effectiveness of our method through several benchmark and real-world datasets.

1. INTRODUCTION

Interpretable machine learning (ML) models are paramount for their seamless integration in high-stake decision making problems e.g., medical diagnosis [1; 2; 3; 4; 5; 6; 7; 8], criminal justice [9; 10; 11]. In medical diagnosis, especially in computer-assisted diagnosis (CAD), model accuracy is important, but it is equally important for the doctor and the patient to know the features used in CAD modeling [12; 13]. There have been several established feature selection (FS) algorithms in ML literature, namely LASSO [14], marginal screening (MS) [15], orthogonal matching pursuit (OMP) [16], and decision tree (DT) based. Among them, DT-based FS has been widely studied due to its high interpretability [17; 18; 19]. Constructing an accurate and a small size (hence, better interpretable) DT is a challenging problem, and has been an active area of research over the last four decades [20; 21]. Most of the existing methods are ad hoc, and do not have explicit control over the size and accuracy of a DT. For example, there are greedy splitting-based [17; 18; 19], Bayesian-based [22; 23; 24; 25; 26], and branch-and-bound methods [9] for DT construction. A single decision tree is interpretable, but often falls short in modeling complex real-world data, and hence, tree ensemble methods such as random forest (RF) [27], genetic programming based decision tree ensemble [28] have been developed. While these existing ensemble methods have shown improvements in prediction accuracy and mitigating overfitting risk, due to the heuristic algorithms of decision tree generation, they often face challenges such as the preference for larger trees, lack of statistical interpretability, randomness in feature importance measurement. In this paper, to handle those challenges, we proposed DT-sampler, a SAT-based decision tree sampling method where we allow the user (e.g., a domain expert) to explicitly control the size and accuracy of a DT. We leverage a Boolean satisfiability(SAT) encoding [29] and propose a novel encoding of the DT sample space using only branch nodes and perform (uniform) sampling of the SAT space with user-specified accuracy and size. The proposed encoding generates a more compact search space than that of the existing methods [21]. Our method is a tree ensemble method that generates small-size and high-accuracy decision trees, and determines the feature importance based on its emergence probability (i.e., the probability of a feature appearing in the high accuracy space).

In a decision tree classifier, relative frequency is generally used to assign a class probability to a test instance. Relative frequency is defined as the proportion of training instances belonging to a specific class in the leaf where the test instance falls. However, these class probabilities are heuristic notion of uncertainty of the underlying decision tree model and do not ensure any valid statistical guarantee. Therefore, we propose to use conformal prediction (CP) to post-calibrate the sampled trees of DT-sampler. CP is a generic framework to post-calibrate any arbitrary (possibly imperfect) predictor and ensures a valid finite sample statistical guarantee under the weak assumptions of data exchangeability [30; 31]. There have been studies to conformalize a decision tree ensemble using CP, such as conformal genetic programming-based tree ensemble [28] or conformal

*This author contributed to this project when he was affiliated with the Department of Computational Biology and Medical Sciences, the University of Tokyo. His current affiliation is Rakuten Group, Inc., Japan.

†Corresponding author: tsuda@k.u-tokyo.ac.jp
‡Corresponding author: diptesh.das@edu.k.u-tokyo.ac.jp

Table 1: Description of propositional variables used in SAT-based DT encoding. The description indicates that a variable is set to 1 if the condition is true; otherwise, it is set to 0.

Var	Description of variables
vl_i	1 iff branch node i has a left branch child, $i \in [N]$
vr_i	1 iff branch node i has a right branch child, $i \in [N]$
l_{ij}	1 iff node i has node j as the left child, with $j \in Child(i)$
r_{ij}	1 iff node i has node j as the right child, with $j \in Child(i)$
lc_i	1 iff class of the left leaf child of node i is 1, $i \in [N]$
rc_i	1 iff class of the right leaf child of node i is 1, $i \in [N]$
a_{rj}	1 iff feature f_r is assigned to node j, $r \in [K]$, $j \in [N]$
u_{rj}	1 iff feature f_r is being discriminated against by node j, $r \in [K]$, $j \in [N]$

random forest [32]. Although these methods enjoy the statistical validity of CP, unlike SAT-based decision tree sampler, these methods do not have explicit control over accuracy and tree size. Hence, these methods are unable to produce optimal (in size and accuracy) ensemble of decision trees. To the best of our knowledge, this is the first attempt to apply conformal prediction in the context of SAT-based decision tree sampling.Through numerical experiments, we evaluated our proposed method using several benchmark and real-world datasets. We demonstrated that our method is capable of producing comparable accuracy as RF but with small-size DTs. We compared our encoding with existing SAT-based encoding and demonstrated that our encoding scheme generates fewer variables and is computationally more efficient. We also performed stability analysis of DT-sampler against RF both in terms of feature importance measurement as well as prediction tasks. The randomness in tree generation in RF makes it difficult to generate stable feature importance measurement results. Furthermore, the post-calibration using CP framework demonstrates that tree ensemble generated by DT-sampler is statistically more efficient and stable than that of random forest tree ensemble. An open source implementation of DT-Sampler is available at `https://github.com/tsudalab/DT-sampler-CP`.

2. METHOD

2.1 SAT-based decision tree encoding

Constructing decision trees with high accuracy and small size is an active area of research in the domain of constraint programming, and many Boolean satisfiability (SAT)-based encodings have been proposed in the literature [33; 21; 34; 35]. To reduce the search space and enable fast sampling we proposed an efficient SAT-encoding of DT. Our encoding method is motivated by the method proposed in [21], but developed a new encoding (only branch node encoding) scheme that accelerates the process of SAT sampling significantly. We also introduced additional variables and constraints to make it possible to encode the DTs with any accuracy that you want. Encoding DTs with only branch nodes is non-trivial, the details of which have been provided below.

SAT variables and constraints. We consider the encoding of decision trees with 2N+1 nodes and the training data consists of M samples and K features. Binary decision tree with 2N+1 nodes comprises N branch nodes and N+1 leaf nodes. The base method [21] sets the node ID sequentially as showed in Figure 1.a. Since it cannot distinguish between branch and leaf nodes by node IDs, branch and leaf nodes are assigned equivalent variables and are differentiated by

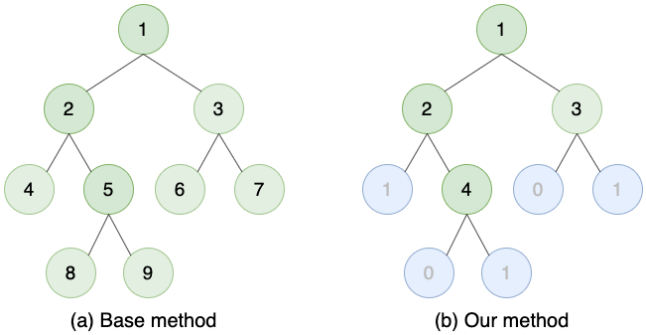

Figure 1: Node ID in SAT-based encoding of DT. (a) Base method cannot distinguish between branch and leaf nodes by node IDs. (b) Our method only uses the branch nodes to encode a DT.

additional constraints. In order to simplify it, we propose a method that only takes the branch nodes into consideration, viewing the leaf nodes as one of the properties of branch nodes as depicted in Figure 1.b. All the variables required to encode a DT are shown in Table 1, the subscript i and j represent the node ID (or index) while the subscript r denotes the feature ID (or index). For any natural number n, we use

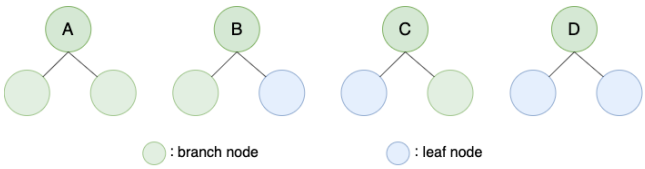

Figure 2: At any level of the DT construction, there can be four types of branch nodes.

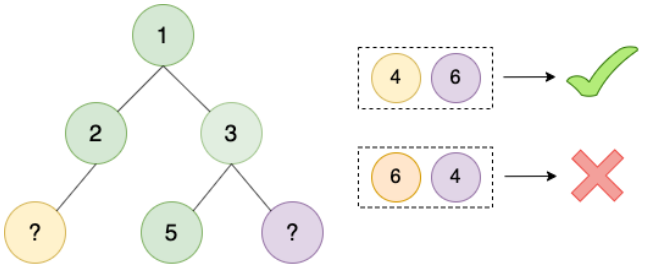

Figure 3: IDs of branch nodes are assigned according to level order of the tree.

$[m:n] = \{m, m+1, \ldots, n-1, n\}$. The function defined as $Child(i) = [i+1 : \min(2i+1, N)]$ can return possible node IDs of the children of the i^{th} node. There are four types of branch nodes as depicted in Figure 2. We use vl_i (resp. vr_i) variable to denote whether the i^{th} node has a left (resp. right) branch child or not. With $i \in [1:N]$ and $C \in \{0,1\}$,

$$vl_i = C \implies \left(\sum_{j \in Child(i)} l_{ij} \right) = C \quad \text{and}$$
$$vr_i = C \implies \left(\sum_{j \in Child(i)} r_{ij} \right) = C. \tag{1}$$

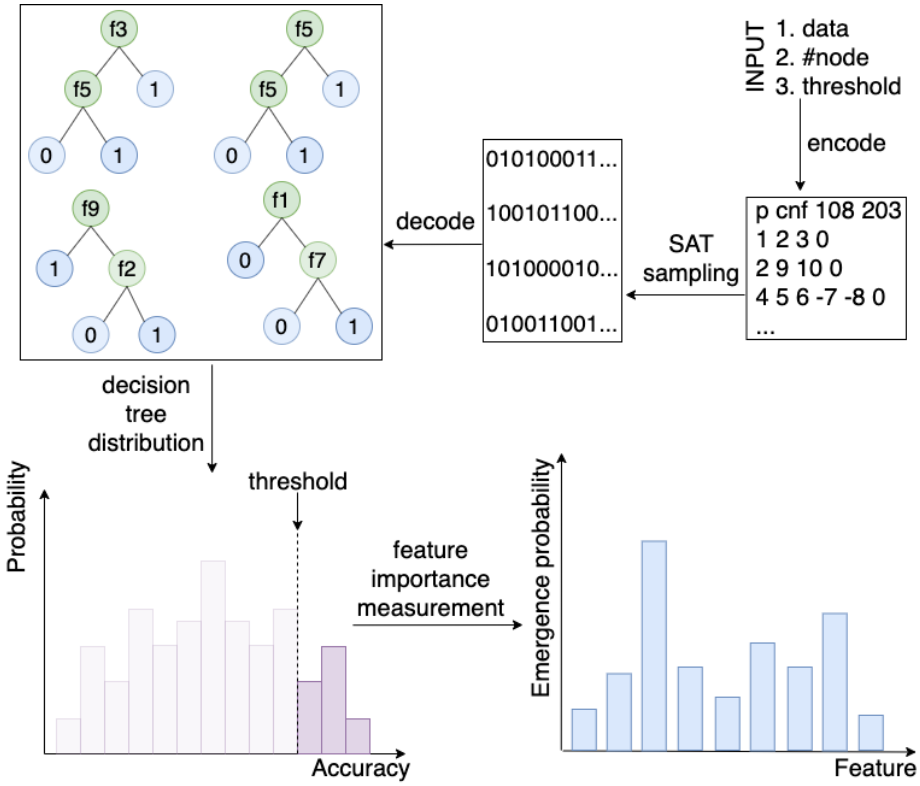

Figure 4: Feature importance measurement based on decision tree sampling by DT-sampler. First, we encode DTs with specific size (#node) and accuracy threshold (τ) as a SAT problem represented in conjunctive normal form (CNF). Once the SAT encoding of DTs is constructed, any solution that satisfies all the constraints in the CNF file can be decoded into a valid decision tree of specific size and accuracy. Then, we utilize SAT sampling to generate multiple decision trees and calculate feature importance (emergence probability) based on the sampling results.

Every branch node (except root) has exactly one parent such that

$$\sum_{i=\lfloor \frac{j}{2} \rfloor}^{j-1} (l_{ij} + r_{ij}) = 1, \quad j \in [2:N]. \tag{2}$$

The IDs of branch nodes are assigned according to level order of the tree. For example, as shown in Figure 3, if $l_{35} = 1$, then l_{26} or r_{26} must be 0, because the 6^{th} node cannot appear in front of the 5^{th} node. With $i \in [1 : N-1]$, $j \in Child(i)$,

$$\begin{aligned} l_{ij} \vee r_{ij} \implies \sum_{h=1}^{i-1} \sum_{k=j}^{k=N} (l_{hk} + r_{hk}) = 0 \quad \text{and} \\ r_{ij} \implies \sum_{k=j}^{k=N} l_{ik} + \sum_{k=j+1}^{k=N} r_{ik} = 0. \end{aligned} \tag{3}$$

At any branch node, exactly one feature is assigned such that

$$\sum_{r=1}^{K} a_{rj} = 1, \quad j \in [1:N]. \tag{4}$$

Variable u_{rj} has the information of whether the r^{th} feature is discriminated at any node on the path from the root to this node. If the r^{th} feature has already been assigned to one of ancestors, then it should not be assigned again. With $r \in [1:K]$, $j \in [1:N]$,

$$\begin{aligned} \bigwedge_{i=\lfloor \frac{j}{2} \rfloor}^{j-1} (u_{ri} \wedge (l_{ij} \vee r_{ij}) \implies \neg a_{rj}) \quad \text{and} \\ u_{rj} \iff (a_{rj} \vee \bigvee_{i=\lfloor \frac{j}{2} \rfloor}^{j-1} (u_{ri} \wedge (l_{ij} \vee r_{ij}))). \end{aligned} \tag{5}$$

The encoding given by Formula (1)-(5) specify a space including all of valid decision trees of a given size but can't learn from the training data. To learn from the training data, we need to track if the r^{th} feature was discriminated positively or negatively along the path from the root to j^{th} node as proposed in [21]. We adopted the same strategy in our method. Furthermore, to constrain the accuracy of the sampled decision trees, we introduce a binary variable w_t for each training example (x_t, y_t). The variable $w_t = 1$ if the sampled decision tree correctly classifies the t^{th} example. Formula (6) enforces that the overall classification accuracy must meet or exceed a specific threshold $\tau \in [0, 1]$. For example, if the parameter $\tau = 0.8$, then the decision trees must correctly classify at least 80% samples such that

$$\frac{1}{M} \sum_{t=1}^{M} w_t \geq \tau, \quad \tau \in [0, 1]. \tag{6}$$

Table 2: Comparison of encoding size. $\#b$: number of training samples, $\#f$: number of features, $\#n$: number of decision tree nodes, and τ : training accuracy threshold (6), used in encoding. The $\#var$ denotes the number of variables used to build the encoding of the decision tree. The $\#var\_cnf$, $\#cls\_cnf$ denote the number of variables and the number of clauses in the Conjunctive Normal Form (CNF) file generated by Tseitin transformation provided in z3-solver [36]. $\#Ave.\ time$ denotes the time to generate 100 samples by unigen. We ran all the experiments (including the base encoding) on Intel(R) Xeon(R) CPU E3-1270 v6 3.80GHz. All results are shown in the order of existing base encoding method and our proposed DT-sampler.

Dataset	$\#b$	#f	$\#n$	τ	$Method$	$\#var$	$\#var\_cnf$	$\#cls\_cnf$	$Ave.\ time\ (s)$
mouse	50	15	13	1.00	Base [21]	896	3599	20586	31.76
					DT-sampler	406	1274	8397	0.25
	50	15	11	0.90	Base [21]	747	3260	22890	1028.02
					DT-sampler	336	1990	24689	567.08
car	40	10	17	1.00	Base [21]	866	3956	21625	260.68
					DT-sampler	390	1406	9495	65.47
	50	15	11	0.90	Base [21]	747	3436	26218	1722.26
					DT-sampler	336	2152	33473	310.99
breast	40	15	17	1.00	Base [21]	1206	5821	32400	96.34
					DT-sampler	550	2041	13663	2.32
	50	12	13	0.90	Base [21]	740	3759	24732	407.32
					DT-sampler	334	2198	27869	125.81
heart	40	19	13	1.00	Base [21]	1104	4572	26063	79.87
					DT-sampler	502	1590	10697	11.62
	50	10	13	0.90	Base [21]	636	3241	21568	1671.18
					DT-sampler	286	2108	25552	327.89

2.2 Decision tree sampling

Sampling method. To obtain samples from the decision tree space, we employ two SAT samplers: QuickSampler [37] and UniGen3 [38]. QuickSampler is a heuristic search algorithm that can generate large amounts of samples quickly. The algorithm starts with a random assignment and iteratively modifies the assignment by flipping the truth values of randomly selected variables. It is very efficient but the uniformity cannot be guaranteed. In contrast, UniGen3 is a more sophisticated algorithm for uniform SAT sampling with solid theoretical guarantees. It requires adding extra clauses to the encoding, which makes the sampling process computationally expensive.

Sampling set. Unigen3 and Quicksampler allow users to assign a subset of all the variables as sampling set. If the sampling set contains Y variables, the size of the solution search space will be 2^Y. The samplers provide uniformity within the sampling set and increasing Y may adversely affect the sampling efficiency. Only part of variables will be in the sampling set. For example, u_{rj} is used to ensure that there are no repeated assigned features in any decision path but we do not need it during the decoding process. In addition, either the set $\{vl_i, vr_i\}$ or the set $\{l_{ij}, r_{ij}\}$ contains all the information needed to decode the tree structure, we only need to add one of them in the sampling set. Therefore, the smallest sampling set is $\{vl_i, vr_i, lc_i, rc_i, a_{rj}\}$.

2.3 Feature importance measurement

In [39], the authors measured the importance of elements in sequences based on the distribution under a qualification threshold. Inspired by this concept, we define feature importance as the contribution of each feature to a high accuracy space. Specifically, within a space consisting of decision trees surpassing a given threshold, the contributions can be evaluated based on the probability of each feature appearing in this space (we name it as "emergence probability"). Since we sample decision trees from uniform distribution, we can estimate the probability by just counting how many times each feature appears. The flow diagram of our method is shown in Figure 4. Random forest often uses feature permutation or mean decrease in impunity to calculate feature importance. It is also possible to apply these approaches to our framework.

2.4 Calibration using conformal prediction

We use conformal prediction to conformalize the sampled trees by a DT-sampler, thus generating an ensemble of conformal trees. Unlike the standard decision tree, a conformal decision tree produces a prediction set $C^\alpha(x)$ that contains the (unknown) true label y of a test example x with probability at least $1-\alpha$, for any error rate $\alpha \in [0,1]$ such that

$$\mathbb{P}\Big(y \in C^\alpha(x)\Big) \geq 1-\alpha. \tag{7}$$

In a binary classification with two possible categories (1: positive class and 0: negative class), we define the set $\mathcal{Y} = \big\{\{0\},\{1\},\{0,1\},\emptyset\big\}$. A conformal decision tree produces a prediction set $C^\alpha \subset \mathcal{Y}$ that may contain a single class, multiple classes, or be empty. Theoretically, any conformal decision tree is a statistically valid tree and it conforms to the coverage guarantee stated in (7). Therefore, we are mainly interested in finding statistically efficient conformal trees where efficiency is determined by the proportion of singleton (one class) prediction against multi-class or empty predictions it makes. A conformal tree is statistically efficient if it makes a large number of singleton predictions and a small number of multi-class or empty predictions.The statistical efficiency of a conformal tree depends on many factors including the accuracy of the underlying prediction model. The size (node counts) of a decision tree is a key factor that trades off between accuracy and interpretability of a decision tree. Existing conformal tree ensemble methods [32] suggest that unpruned tree is essential for generating accurate prediction model and improved statisti-

Algorithm 1: Inductive (or split) conformal prediction

Input:

Data $(X_i, Y_i) \in \mathbb{R}^p \times \mathcal{Y},\ 1 \leq i \leq n$;
$\mathcal{Y} = \{1, \ldots, m\}$ is a finite set of m classes;
Test data $X_{\text{test}} \in \mathbb{R}^p$;
Miscoverage level $\alpha \in (0, 1)$;
Prediction algorithm A used in "Fit model" below.

Process:

- Randomly split $\{1, \ldots, n\}$ into disjoint sets $\mathcal{I}_{\text{prop}}$ and $\mathcal{I}_{\text{cal}}$;
- Fit model: $P(Y = y|X = x) \leftarrow A(\{(X_i, Y_i),\ \forall i \in \mathcal{I}_{\text{prop}}\})$;
- Compute non-conformity scores $s_i,\ \forall i \in \mathcal{I}_{\text{cal}}$;
- Compute $\hat{q}_{1-\alpha}$, the $\frac{\lceil(|\mathcal{I}_{\text{cal}}|+1)(1-\alpha)\rceil}{|\mathcal{I}_{\text{cal}}|}$ empirical quantile of $\{s_i : i \in \mathcal{I}_{\text{cal}}\}$.

Output:

Prediction set $\mathcal{C}^{\alpha}(X_{\text{test}}) = \{y : s(X_{\text{test}}, y) \leq \hat{q}_{1-\alpha}\}$.

cal efficiency. However, this may lead to generation of large and complex decision trees (long chain of "if-then-else" rules), thus compromising interpretabilty, one of the key benefits of using decision tree in the context of trustworthy AI. Unlike existing tree ensemble frameworks (e.g., random forest, genetic algorithm-based tree ensemble), we propose a SAT-based decision tree sampler (DT-sampler) where we have explicit control over both the size and accuracy of the generated trees. Hence, although we generate an ensemble of trees, the generated trees are of similar size and accuracy, and provides better interpretability than the existing tree ensemble methods. See results for empirical evaluation.

Inductive conformal prediction. We use inductive conformal prediction (ICP) framework [40; 30] as it is computationally efficient. Given a labelled dataset $\mathcal{D}_\ell = \{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$, we randomly split the indices $\{1, \ldots, \ell\}$ into two disjoint sets $\mathcal{I}_{tr}$ and $\mathcal{I}_{te}$. The training set indices $\mathcal{I}_{tr}$ is further splitted into another two disjoint sets $\mathcal{I}_{prop}$ and $\mathcal{I}_{cal}$. A prediction model (e.g. DT-sampler) is trained only once with the examples in the proper training set. The trained prediction model is then used to generate the non-conformity scores ($s \in \mathbb{R}$) for the examples in calibration set as well as for the test instances, where a large score implies bad agreement between x and y. We then compute the $\hat{q}_{1-\alpha}$ as the $\frac{\lceil(|\mathcal{I}_{cal}|+1)(1-\alpha)\rceil}{|\mathcal{I}_{cal}|}$ quantile of the calibration scores such that

$$\hat{q}_{1-\alpha} = Quantile\left(s_1, \ldots, s_n; \frac{\lceil(|\mathcal{I}_{cal}| + 1)(1-\alpha)\rceil}{|\mathcal{I}_{cal}|}\right). \quad (8)$$

Therefore, an inductive conformal prediction set $C^{\alpha}(X_{test})$ for any test example X_{test} and any miscoverage level $\alpha \in (0, 1)$ can be defined such that

$$C^{\alpha}(X_{test}) = \{y \in \mathcal{Y} : s(X_{test}, y) \leq \hat{q}_{1-\alpha}\}. \quad (9)$$

To compute this prediction set (9), we require a prediction model A and an associated score function S. A common choice of non-conformity score function in classification settings is the *"1 – class probability"*, defined as

$$s(x, y) = 1 - p_x^y,$$

or, alternatively, the *"1 – class margin"*, defined as

$$s(x, y) = 1 - \big(p_x^y - \max_{y' \in \mathcal{Y} \setminus \{y\}} p_x^{y'}\big),$$

where $p_x^y = \mathbb{P}(Y = y \mid X = x)$ denotes the conditional class probability, and $\mathcal{Y} = \{1, \ldots, m\}$ is the finite set of m class labels. In the ICP framework, the training of a prediction model and the generation of the non-conformity scores of the calibration set examples are executed only once, and both the learned model as well as the calibration scores are stored for repeated use, thus making it computationally efficient. Now, we formally state the coverage guarantee of ICP framework next.

THEOREM 1 (ICP COVERAGE GUARANTEE [40]). *If $\{(X_i, Y_i)\}_{i \in \mathcal{I}_{cal}}$ and (X_{test}, Y_{test}) are exchangeable random variables and we define $\hat{q}_{1-\alpha}$ (8) and $C^{\alpha}(X_{test})$ (9), then for any miscoverage level $\alpha \in (0, 1)$,*

$$\mathbb{P}\Big(Y_{test} \in C^{\alpha}(X_{test})\Big) \geq (1 - \alpha).$$

Note that the above coverage is called "marginal coverage" where the probability is marginal (averaged) over the randomness of the calibration set and the test data point. For the proof of Theorem 1, please see [40]. An algorithm to compute the prediction set is given in Algorithm 1.

3. RESULTS

Dataset. We compared DT-sampler with RF using several real-world benchmark datasets [41]. In addition to that we also used real world biological and criminal justice dataset. We considered Entacmaea quadricolor fluorescent protein eqFP611, two variant of which namely one bright deep-red (mKate2, $\lambda_{ex} = 590$nm, $\lambda_{em} = 635$nm) and one bright blue (mTagBFP2, $\lambda_{ex} = 405$nm, $\lambda_{em} = 460$nm) are seperated by thirteen mutations [42]. From biological perspective it is important to identify the crucial mutations and their pattern of epistasis (high-order interactions among mutations) that relate to the phenotypes (e.g., brightness). We also evaluated our method using ProPublica's COMPAS recidivism dataset [43] which contains seven categorical and integer-valued features and binary class labels. The equivalent 14 binary features and binary class labels are download from the Github repository of CORELS [9]. Model interpretabilty is crucial for the analysis of such high-stake decision making problems where an algorithm derived predictions are associated with the life of a human being or critical biological analysis.

Comparison with existing SAT encodings. Our new encoding of tree structure reduces a large part of variables and constraints compared to the (base) encoding method in [21]. The results on several benchmark datasets proved the acceleration in the process of SAT sampling as shown in Table 2.

Prediction stability analysis using conformal prediction. Here, we present the calibration results obtained using the Conformal Prediction (CP) framework to demonstrate the statistical efficiency and stability of the DT-sampler compared with the Random Forest (RF) classifier. In an

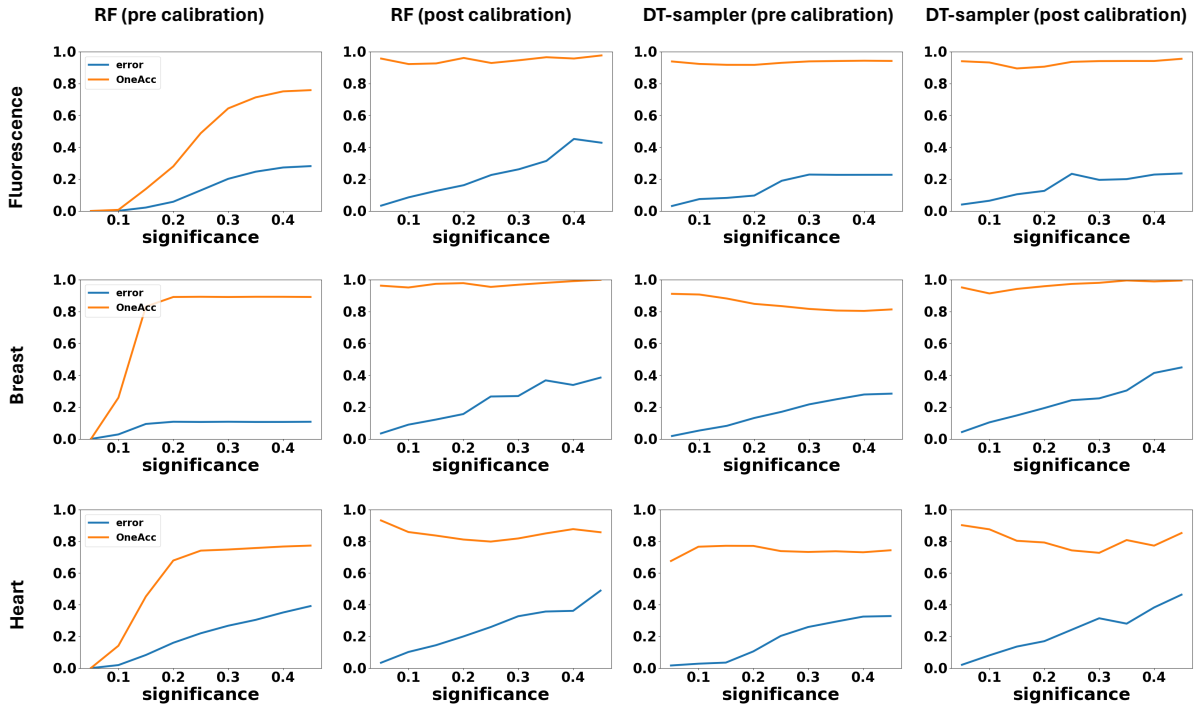

Figure 5: Pre and post calibration results using Fluorescence, Breast and Heart data at different significance levels. Here, error indicates the average miscoverage rate and OneAcc indicates the average one class accuracy. In pre-calibration, a single decision tree predictions are calibrated, whereas in post-calibration average predictions of all decision trees in an ensemble are calibrated.

RF, predictions are derived by aggregating the outputs of multiple randomly constructed decision trees, without explicit control over individual tree accuracy or size. Analysis of individual tree performance typically shows that many constituent trees have limited predictive accuracy. Nevertheless, the ensemble averaging process enables the RF to achieve strong overall predictive performance, albeit at the expense of interpretability. In contrast, the DT-sampler encodes both accuracy and tree size as satisfiability (SAT) constraints, enabling the generation of trees from regions characterized by high accuracy and user-specified complexity. This approach improves interpretability and prevents the formation of excessively large trees that may overfit the training data and generalize poorly to validation or test datasets. To demonstrate this effect we performed CP-based pre-calibration and post-calibration. In pre-calibration, individual tree predictions generated by both RF and DT-sampler are calibrated, whereas in post-calibration, tree ensemble predictions (average predictions) of both RF and DT-sampler are calibrated and the results are shown in Figure 5. We performed experiments on three datasets for different significance levels and for each significance level we repeated experiments for five times and reported the average results. We plotted the average one class accuracy (OneAcc) and average mis-coverage rate (error) for each significance level. Owing to the coverage guarantee of the CP framework, it can be observed that the mis-coverage rates are well controlled at every significance level for all experiments. However, the main differentiating factor is the OneAcc which is an indicator of statistical efficiency of the underlying prediction model. Precisely, a statistically efficient predictor is the one which makes maximum number of single class predictions and minimum number of multi-class or empty predictions. It can be clearly observed that the pre-calibrated OneAcc values of RF are bad at smaller significance (high coverage) levels for all the experiments, indicating that RF generates many random bad (statistically inefficient) predictors (trees) and averages them. On the other hand, both pre and post calibrated OneAcc values of DT-sampler are quite stable for all significance levels, indicating that DT-sampler is statistically more efficient at all significance levels.

Comparing accuracy and tree size between DT-sampler and RF. We compared our method with RF on several real-world benchmark datasets [41]. As shown in Table 3, our method can provide similar accuracy compared with random forest even if we sample decision trees in a small space. Relying on heuristic rules to build decision trees, random forest tends to generate larger decision trees, whereas DT-sampler have explicit control over tree size and can generate similar accuracy with smaller size trees.

Stable feature importance measurement. We define feature importance as its emergence probability in the high accuracy space as mentioned in §2.3. Parameter τ is used to describe what a high accuracy space means and its value depends on specific real-world scenarios and the desired level of strictness regarding accuracy requirements. To demonstrate our method, we utilize decision tree sampling on a subset of the breast-cancer dataset, which consists of 150 samples and 15 selected features. Initially, we set the accu-

Table 3: Comparison of tree sizes and accuracy. Grid search on parameters *max_leaf_nodes* is utilized to run random forest (RF). We reported mean ± standard deviation results of training and test accuracies of three experiments on different subsets of the corresponding datasets, shown in the order of RF/DT-sampler. $\#b$: number of training samples, $\#f$: number of features, $\tau(\%)$: training accuracy threshold (6) in percentage.

Dataset	$\#b$	$\#f$	τ (%)	*Method*	*#node*	*Training Acc.* (%)	*Test Acc.* (%)
mouse	50	15	92.0	RF	6.90	97.50 ± 2.50	93.33 ± 11.55
				DT-sampler	7.00	97.50 ± 2.50	93.33 ± 5.77
car	100	15	92.0	RF	16.93	92.92 ± 3.15	78.33 ± 5.77
				DT-sampler	11.00	94.00 ± 3.46	90.00 ± 0.00
breast	150	15	81.6	RF	19.00	96.94 ± 1.73	97.78 ± 1.92
				DT-sampler	11.00	98.00 ± 1.00	96.00 ± 3.46
heart	170	19	81.0	RF	23.00	93.33 ± 2.08	80.95 ± 2.18
				DT-sampler	15.00	85.67 ± 1.53	78.57 ± 6.23
diabetes	442	10	70.0	RF	11.00	79.37 ± 2.95	72.26 ± 5.78
				DT-sampler	9.00	76.83 ± 2.57	72.18 ± 4.01
penguins	214	6	90.0	RF	9.64	97.69 ± 1.10	96.85 ± 2.48
				DT-sampler	6.00	93.00 ± 1.73	91.23 ± 1.52
sonar	214	10	75.0	RF	5.00	78.61 ± 0.48	71.21 ± 5.12
				DT-sampler	5.00	78.00 ± 3.00	66.67 ± 0.93
compas	721	14	60.0	RF	14.64	70.89 ± 1.06	65.02 ± 5.57
				DT-sampler	7.00	68.56 ± 0.96	68.96 ± 2.19
fluorescence-1	100	13	90.0	RF	22.52	94.47 ± 5.07	90.60 ± 4.71
				DT-sampler	7.00	92.33 ± 2.08	90.73 ± 0.73
fluorescence-2	384	91	90.0	RF	33.48	95.00 ± 4.36	91.78 ± 1.94
				DT-sampler	11.00	91.33 ± 2.31	86.03 ± 3.18

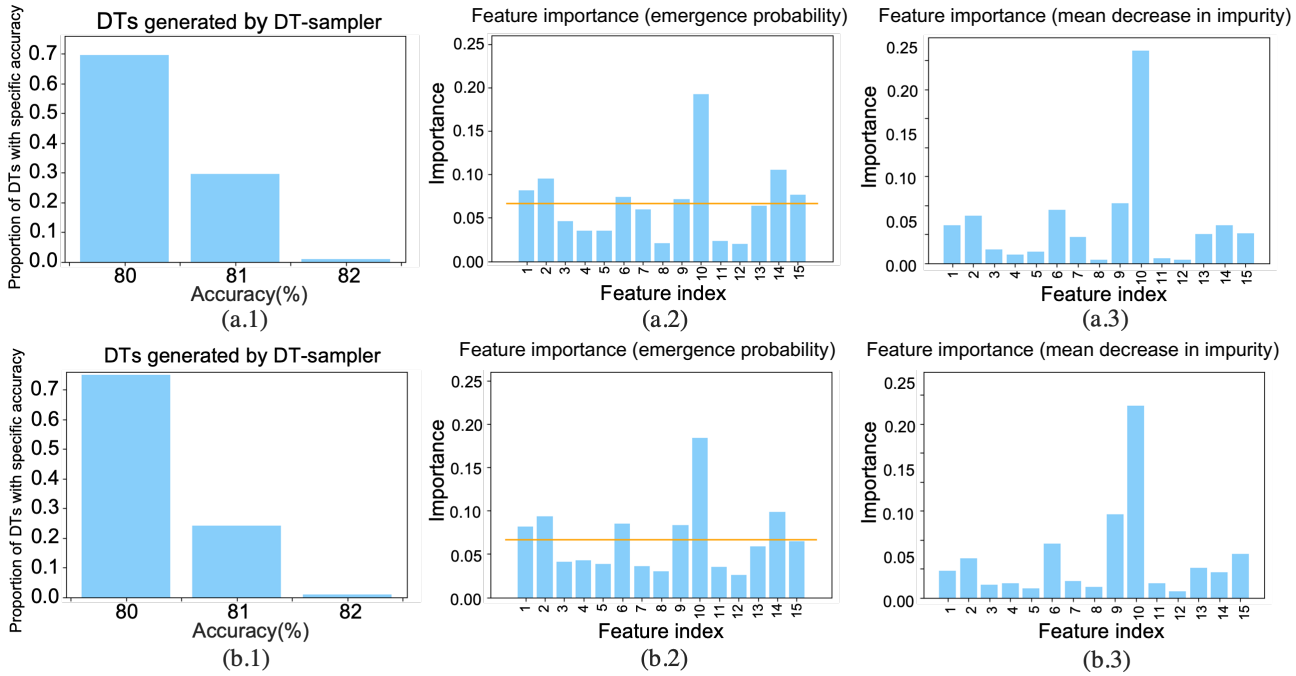

Figure 6: Stability of decision tree sampling. The two rows of figures show the results of two experiments on breast dataset using different random seeds during decision tree sampling. The first column shows the training accuracy distribution of the sampling results. The second and third columns show the feature importance measured by emergence probability and mean decrease in impurity, respectively.

racy threshold (τ) to 0, allowing for the random sampling of any decision tree. In this case, each feature is assigned to any branch node with equal probability, resulting in an emergence probability of $\frac{1}{15}$ for each feature. However, as we increase the threshold (τ), the emergence probabilities of the features differ. Features with an emergence probability

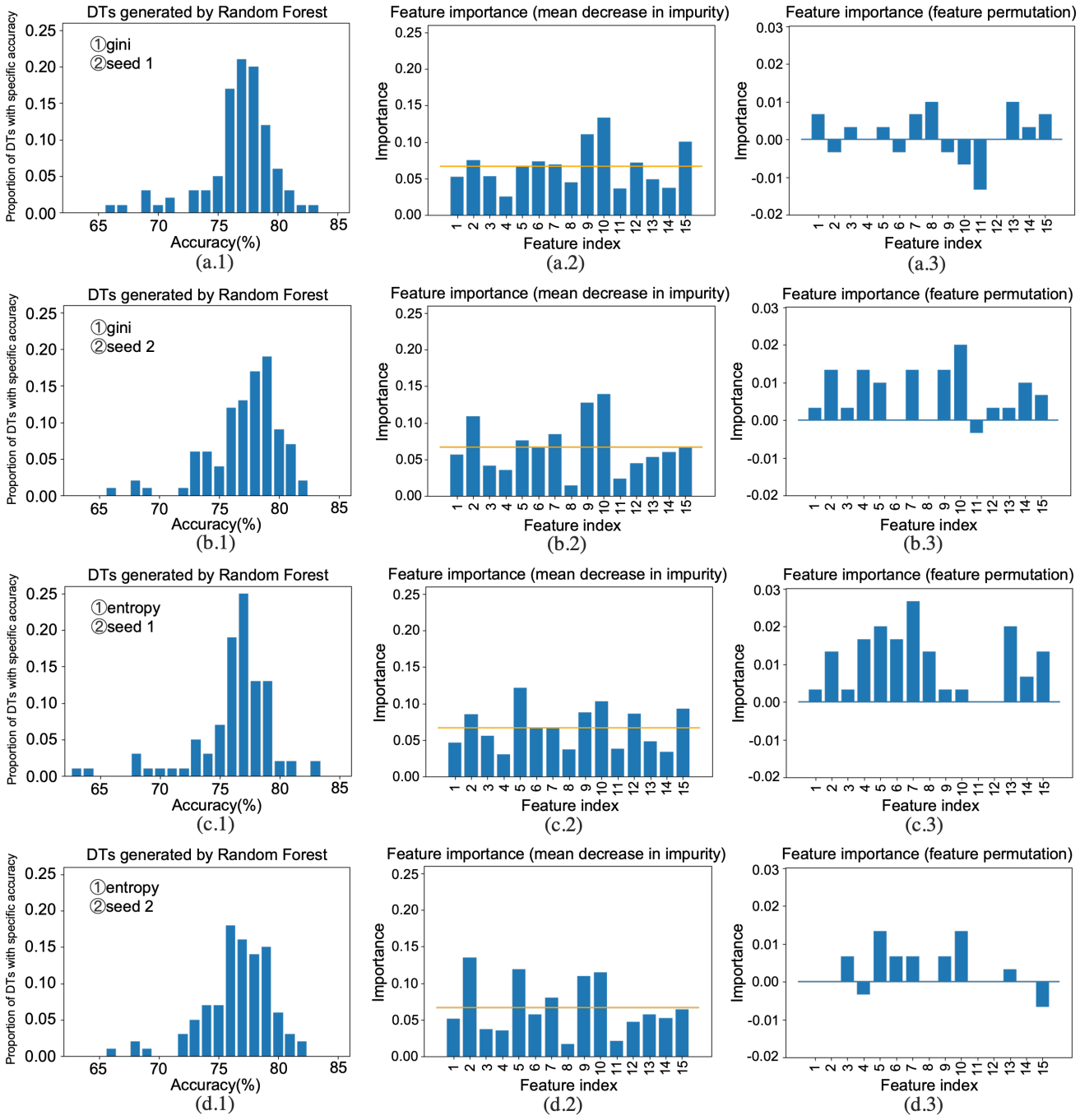

Figure 7: Drawbacks of random forest. The rows show the results of four experiments on breast dataset using different random seeds (seed 1, seed 2) and splitting criterion (gini, entropy) as random forest parameters. The first column shows the training accuracy distribution of the decision trees generated by random forest. The second and third columns show the feature importance measured by mean decrease in impurity and feature permutation, respectively.

$\geq \frac{1}{15}$ are considered important (see Figure 6). In random forest, the randomness of tree generation makes it difficult to generate stable results for feature importance measurement (see Figure 7 in appendix). DT-sampler shows superior stability compared to random forest. In Figure 7, we observe that when different random seeds or parameters are used, the distribution of decision trees generated by random forest consistently changes. This variability in tree generation directly impacts the feature importance results, leading to significant differences. Furthermore, random forest tends to generate a large number of trees with low accuracy, making it unreliable to measure feature importance for real-world problems. In contrast, DT-sampler calculates feature importance based on decision trees sampled exclusively from a

high accuracy space, which ensures the stability and interpretability of our results as depicted in Figure 6.

4. CONCLUSION

We proposed an SAT-based decision tree ensemble method and compared it with random forest using several benchmark and real-world datasets. We demonstrated that due to the randomness in tree generation and over-dependence on many parameters, random forest-based predictions and feature selections are unstable and unreliable. Our method provides a principled framework to measure feature importance based on sampling results from a high-accuracy space with a clear threshold, which offers stable analysis results for real-world problems. Using the conformal prediction framework, we demonstrated that the proposed method is statistically more efficient and produces stable predictions compared to random forest. Potential future research directions include developing a statistical hypothesis testing framework based on the proposed DT sampling method to assess the reliability of feature selection, and designing a fast SAT solver using quantum annealing or other QUBO-based approaches to enhance sampling efficiency.

5. AUTHOR CONTRIBUTIONS

Chao Huang, Koji Tsuda, and Diptesh Das conceived the idea and developed the methodology. Chao Huang, Xiaotian Xue, and Diptesh Das implemented the method. Diptesh Das designed the experiments; Chao Huang and Xiaotian Xue conducted them and generated results. All authors analyzed the results. Diptesh Das and Chao Huang wrote the manuscript. Diptesh Das and Koji Tsuda supervised the project. All authors reviewed and approved the final manuscript.

6. ACKNOWLEDGEMENTS

Koji Tsuda is supported by JST MIRAI, JST ERATO JPMJER1903 and JST CREST JPMJCR21O2. Diptesh Das is supported by JSPS KAKENHI 23K16942.

7. REFERENCES

[1] Himabindu Lakkaraju, Stephen H Bach, and Jure Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1675–1684, 2016.

[2] Shamim Nemati, Andre Holder, Fereshteh Razmi, Matthew D Stanley, Gari D Clifford, and Timothy G Buchman. An interpretable machine learning model for accurate prediction of sepsis in the icu. *Critical care medicine*, 46:547–553, 2018.

[3] Muhammad Aurangzeb Ahmad, Carly Eckert, and Ankur Teredesai. Interpretable machine learning in healthcare. In *Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics*, pages 559–560, 2018.

[4] Diptesh Das, Junichi Ito, Tadashi Kadowaki, and Koji Tsuda. An interpretable machine learning model for diagnosis of alzheimer's disease. *PeerJ*, 7:e6543, 2019.

[5] Diptesh. Das. *Interpretable Machine Learning Models for Medical Data*. Ph.D. diss., Department of Computational Biology and Medical Sciences, The University of Tokyo, Kashiwa, Japan, 2019.

[6] Diptesh Das, Vo Nguyen Le Duy, Hiroyuki Hanada, Koji Tsuda, and Ichiro Takeuchi. Fast and more powerful selective inference for sparse high-order interaction model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9999–10007, 2022.

[7] Diptesh Das, Eugene Ndiaye, and Ichiro Takeuchi. A confidence machine for sparse high-order interaction model. *Stat*, 13:e633, 2024.

[8] Amina Adadi and Mohammed Berrada. Explainable ai for healthcare: from black box to interpretable models. In *Embedded Systems and Artificial Intelligence: Proceedings of ESAI 2019, Fez, Morocco*, pages 327–337. Springer, 2020.

[9] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 35–44, 2017.

[10] Caroline Wang, Bin Han, Bhrij Patel, and Cynthia Rudin. In pursuit of interpretable, fair and accurate machine learning for criminal recidivism prediction. *Journal of Quantitative Criminology*, 39:519–581, 2023.

[11] Jiachang Liu, Chudi Zhong, Boxuan Li, Margo Seltzer, and Cynthia Rudin. Fasterrisk: fast and accurate interpretable risk scores. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pages 17760–17773, 2022.

[12] Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38:50–57, 2017.

[13] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[14] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[15] Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70:849–911, 2008.

[16] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar conference on signals, systems and computers*, pages 40–44. IEEE, 1993.

[17] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1:81–106, 1986.

[18] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[19] L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Taylor & Francis, 1984.

[20] Hendrik Blockeel, Laurens Devos, Benoît Frénay, Géraldin Nanfack, and Siegfried Nijssen. Decision trees: from efficient prediction to responsible ai. *Frontiers in Artificial Intelligence*, 6:1124553, 2023.

[21] Nina Narodytska, Alexey Ignatiev, Filipe Pereira, and Joao Marques-Silva. Learning optimal decision trees with sat. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 1362–1368, 2018.

[22] David GT Denison, Bani K Mallick, and Adrian FM Smith. A bayesian cart algorithm. *Biometrika*, 85:363–377, 1998.

[23] Hugh A Chipman, Edward I George, and Robert E McCulloch. Bayesian cart model search. *Journal of the American Statistical Association*, 93:935–948, 1998.

[24] Hugh A Chipman, Edward I George, and Robert E McCulloch. Bayesian treed models. *Machine Learning*, 48:299–320, 2002.

[25] Hugh A Chipman, Edward I George, and Robert E McCulloch. Bart: Bayesian additive regression trees. 2010.

[26] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. 2015.

[27] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[28] Ulf Johansson, Rikard König, Tuve Löfström, and Henrik Boström. Evolved decision trees as conformal predictors. In *2013 IEEE Congress on Evolutionary Computation*, pages 1794–1801. IEEE, 2013.

[29] Armin Biere, Marijn Heule, and Hans van Maaren. *Handbook of satisfiability*, volume 185. IOS press, 2009.

[30] Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

[31] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9:371–421, 2008.

[32] Ulf Johansson, Henrik Boström, and Tuve Löfström. Conformal prediction using decision trees. In *2013 IEEE 13th international conference on data mining*, pages 330–339. IEEE, 2013.

[33] Christian Bessiere, Emmanuel Hebrard, and Barry O'Sullivan. Minimising decision tree size as combinatorial optimisation. In *Principles and Practice of Constraint Programming-CP 2009: 15th International Conference, CP 2009 Lisbon, Portugal, September 20-24, 2009 Proceedings 15*, pages 173–187. Springer, 2009.

[34] Hélene Verhaeghe, Siegfried Nijssen, Gilles Pesant, Claude-Guy Quimper, and Pierre Schaus. Learning optimal decision trees using constraint programming. *Constraints*, 25:226–250, 2020.

[35] Mikoláš Janota and Antonio Morgado. *SAT-Based Encodings for Optimal Decision Trees with Explicit Paths*, pages 501–518. 06 2020.

[36] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[37] Rafael Dutra, Kevin Laeufer, Jonathan Bachrach, and Koushik Sen. Efficient sampling of sat solutions for testing. In *Proceedings of the 40th International Conference on Software Engineering*, page 549–559, 2018.

[38] Mate Soos, Stephan Gocht, and Kuldeep Meel. *Tinted, Detached, and Lazy CNF-XOR Solving and Its Applications to Counting and Sampling*, pages 463–484. 07 2020.

[39] Jiawen Li, Jinzhe Zhang, Ryo Tamura, and Koji Tsuda. Self-learning entropic population annealing for interpretable materials design. *Digital Discovery*, 1:295–302, 2022.

[40] Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. Inductive confidence machines for regression. In *Proceedings of the 13th European Conference on Machine Learning*, pages 345–356. Springer, 2002.

[41] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[42] Frank J Poelwijk, Michael Socolich, and Rama Ranganathan. Learning the pattern of epistasis linking genotype and phenotype in a protein. *Nature communications*, 10:4213, 2019.

[43] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How we analyzed the compas recidivism algorithm. *ProPublica*, May 23, 2016. Accessed: 2025-11-09.