Distillation Decision Tree

Xuetao Lu [a] and J. Jack Lee [a*]
[a] Department of Biostatistics, The University of Texas MD Anderson Cancer Center
[*] Author for correspondence: jjlee@mdanderson.org

Abstract

Black-box machine learning models are criticized as lacking interpretability, although they tend to have good prediction accuracy. Knowledge Distillation (KD) is an emerging tool to interpret the black-box model by distilling its knowledge into a transparent model. With well-known advantages in interpretation, decision tree is a competitive candidate of the transparent model. However, theoretical or empirical understanding for the decision tree generated from KD process is limited. In this paper, we name this kind of decision tree the distillation decision tree (DDT) and lay the theoretical foundations for tree structure stability which determines the validity of DDT's interpretation. We prove that the structure of DDT can achieve stable (convergence) under some mild assumptions. Meanwhile, we develop algorithms for stabilizing the induction of DDT, propose parallel strategies for improving algorithm's computational efficiency, and introduce a marginal principal component analysis method for overcoming the curse of dimensionality in sampling. Simulated and real data studies justify our theoretical results, validate the efficacy of algorithms, and demonstrate that DDT can strike a good balance between model's prediction accuracy and interpretability.

Keywords: Knowledge distillation, Decision tree, Machine learning, Model interpretability, Prediction accuracy.

1 Introduction

In the past decade, we have witnessed the rise of machine learning (ML), which has been used in multiple research fields and industries (e.g., healthcare, e-commerce, finance, cybersecurity). Netflix awarded a $1 million prize to a developer team in 2009 for an ensemble ML algorithm that increased the accuracy of the company's recommendation engine by 10 percent. Regardless of such success, most ML black-box models (e.g., neural networks, gradient boosting models, or complicated ensemble models) face increased skepticism and criticism. Black-box model inner mechanisms are difficult to understand, making people question whether related model-reliant decisions are well-grounded and trustworthy. Model

interpretability is essential to avoiding gross errors and hidden bias in decision-based applications such as medical diagnosis(Xie et al., 2019) and the safety-critical and automotive industries. Alternatively, a simple and transparent model can help researchers explore and verify underlying biological mechanisms in medicine as they seek effective treatments. Knowledge distillation (KD) was proposed for model compression by transferring knowledge from a cumbersome teacher model to a small student model(Hinton et al., 2015). Teacher models were trained for extracting complex knowledge from data. By applying a different kind of training, called "distillation," the teacher model condenses to a student model, which is lightweight and more suitable for deployment without sacrificing too much prediction accuracy. This idea was first proposed by Buciluǎ et al. (2006). KD then gained popularity and increased attention in the computer science community (Ba and Caruana, 2014; Hinton et al., 2015; Urban et al., 2017). Recent research on KD mostly focused on: (1) understanding why and when KD works (Hinton et al., 2015; Stanton et al., 2021; Wang and Yoon, 2021; Allen-Zhu and Li, 2021; Menon et al., 2021), (2) developing algorithms to achieve efficient KD (Shi et al., 2019; Kang and Gwak, 2020; Du et al., 2020), and (3) interpreting the black-box (teacher) model by distilling it into a transparent (student) model. In this final focus area, Johansson et al. (2011) first proposed choosing an individual decision tree as the transparent student model, due to its excellent interpretability and prediction accuracy. Distilling knowledge into a decision tree to deploy and interpret black-box models, especially with deep neural networks, is becoming a hot field in computer science research (Frosst and Hinton, 2017; Coppens et al., 2019; Li et al., 2020; Song et al., 2021; Ding et al., 2021). Shen et al. (2020) applied KD to a decision tree to reconcile repeat buyer predictive and interpretable performance for e-commerce companies. By using a decision tree generated from an aggregated ML model, they identified the reputation of the merchant's store as the most important factor, consistent with the intuition that people often prefer to purchase in stores with better reputations. They also found that a merchant with a repeat buyer ratio above 8.7% is considered to have a good reputation which can influence customers' decisions. However, all of above studies ignored decision tree stability when discussing interpretability. Yu (2013) argued that interpretability needs stability. The conclusions of statistical analysis must be robust to small data perturbations. Unfortunately, it is well-known that decision trees are highly sensitive to their training data sets. Without

considering tree structure stability, decision tree interpretation is questionable. Zhou et al. (2018) examined tree structure stability in KD. To ensure the stability of each split in the tree, they developed tests based on an asymptotic distribution of the Gini index. However, they did not answer whether and under what conditions a split will achieve convergence (stability). Also, they narrowed their discussions to decision tree classification with a particular splitting criteria, the Gini index(Breiman et al., 1984). Notably, a decision tree is capable in both classification and regression applications. Even in classification, there are many splitting criteria such as the Shannon entropy(Quinlan, 1986), gain ratio(Quinlan, 1993), and the Gini index. Since it is not clear which criterion is most suitable for certain applications, a comprehensive study of all above mentioned criterions for tree structure stability in KD is necessary.

For the first time in the paper, we provide theoretical foundations and practical algorithms for the interpretability of the decision tree generated from KD, which we name the distillation decision tree (DDT) hereafter. Since the credibility of the tree's interpretation is conditional on structural stability—which is further determined by split (branching) stability—we conduct a thorough theoretical study on split stability and prove that splits will converge in probability with a particular convergence rate under some mild assumptions. The theoretical results hold in both classification and regression, and they cover the most popular splitting criteria: Sum Square Error/Mean Square Error (SSE/MSE) for regression and Tsallis entropy for classification. Tsallis entropy is a general framework that unifies Shannon entropy, gain ratio, and the Gini index (Wang and Xia, 2017). We propose and implement algorithms for the induction of DDT. Since the algorithms are computationally intensive, we also provide parallel solutions. For high-dimension data, however, computation is infeasible even under these parallel settings. In this case, we develop a marginal principal component analysis (PCA) method to overcome the curse of dimensionality. In summary, DDT provides a good balance between prediction accuracy and model interpretability. We develop methods to facilitate its implementation.

The remainder of the paper is organized as follows. In Section 2, DDT prerequisites are introduced. Section 3 presents theoretical results for DDT stability. In Section 4, algorithms for DDT induction are developed. Section 5 exhibits case studies on both simulation and real data analyses. Section 6 concludes the paper with a brief discussion. Theoretical proofs

are provided in the Appendix A. An open source R implementation is available on GitHub (see https://github.com/lxtpvt/ddt.git).

2 Distillation Decision Tree

Simply speaking, DDT is a decision tree transferred from the more complex teacher model through KD. It has all the attributes of an ordinary decision tree (ODT) plus additional properties to enhance interpretability. In this section, we first review ODT, then delineate DDT features and induction.

2.1 Decision Tree

The decision tree's history began in 1963 (Morgan and Sonquist, 1963). It is a non-parametric supervised learning method used for classification and regression (Quinlan, 1986). A decision tree is an excellent candidate for KD due to the following advantages:

- A decision tree is simple and straightforward with if-else statements that are easy to understand, interpret, and visualize. It is a transparent model, which closely mimics the human decision-making process. Decision trees enable dataset-acquired knowledge to be extracted in a readable form.

- A decision tree follows a non-parametric method, meaning it does not depend on any parametric assumption of the distribution of data. It also has good performance on high-dimensional data (Klusowski, 2021).

- A decision tree is naturally suited for inferring non-linear relationships and complex interactions among variables.

- A decision tree can handle both continuous and categorical variables. It requires little data pre-processing. Feature scaling (standardization and normalization) and dummy variables encoding are not required as decision trees use a rule-based approach instead of distance measures.

- A decision tree can handle missing values (Rahman and Islam, 2011) and is usually robust to outliers (John, 1995).

Since the splitting criteria and split search algorithms play central roles in the induction of (distillation) decision tree and are necessary for our theoretical studies, we will review them first.

2.1.1 Splitting Criteria

The splitting criteria used in regression and classification are different. In regression, the major criterion is to minimize the sum square errors (1) or the mean square errors (2).

$$min\{\sum_{i=1}^{n_l}(y_{li}-\bar{y}_l)^2+\sum_{j=1}^{n_r}(y_{rj}-\bar{y}_r)^2\} \tag{1}$$

$$min\{\frac{1}{n_l}\sum_{i=1}^{n_l}(y_{li}-\bar{y}_l)^2+\frac{1}{n_r}\sum_{j=1}^{n_r}(y_{rj}-\bar{y}_r)^2\} \tag{2}$$

where the subscripts l, r represent the left and right node of a stump, $n_l + n_r = n$, $\bar{y}_l = \frac{1}{n_l}\sum_{i=1}^{n_l} y_{li}$ and $\bar{y}_r = \frac{1}{n_r}\sum_{j=1}^{n_r} y_{lj}$.

In classification, we choose the criteria that maximizes the entropy reduction after splitting.

$$max\{E-(E_l+E_r)\}, \tag{3}$$

where E is the total entropy before splitting, and E_l and E_r are the left and right sets of entropy after splitting. The entropy we have chosen is called the Tsallis entropy, which is defined as follows:

$$S_q(Y)=\frac{1}{1-q}(\sum_{i=1}^{C}p(y_i)^q-1), \quad q\in\mathbb{R}, \tag{4}$$

where Y is a random variable that takes value in $\{y_1, ..., y_C\}$, $p(y_i)$ is the corresponding probabilities of y_i, $i = 1, ..., C$, and q is an adjustable parameter.

The classical and prevalent decision tree algorithms, ID3, C4.5 (Quinlan, 1993) and CART (Breiman et al., 1984), use the Shannon entropy, gain ratio, and the Gini index as their splitting criteria, respectively. All of these splitting criteria can be unified in a Tsallis entropy framework (Wang and Xia, 2017). The Tsallis entropy converges to the Shannon entropy (5) in the limit $q \to 1$.

$$ShEnt(Y)=\lim_{q\to 1}S_q(Y)=\lim_{q\to 1}\frac{1}{1-q}(\sum_{i=1}^{C}p(y_i)^q-1)=-\sum_{i-1}^{C}p(y_i)ln(p(y_i)) \tag{5}$$

Let $q = 2$, we can obtain the Gini index (6).

$$GI(Y) = S_q(Y)_{q=2} = \frac{1}{1-2}(\sum_{i=1}^{C} p(y_i)^2 - 1) = 1 - \sum_{i-1}^{C} p(y_i)^2 \tag{6}$$

As for the Gain ratio, it can be defined as (7).

$$GR(Y) = \frac{ShEnt(Y) - \frac{n_l}{n} ShEnt(Y_l) - \frac{n_r}{n} ShEnt(Y_r)}{ShEnt(\frac{n_l}{n}) + ShEnt(\frac{n_r}{n})}, \tag{7}$$

where Y_l and Y_r are the two data sets corresponding to the left and right child nodes. $ShEnt(\cdot)$ denotes the Shannon entropy in (5). Thus, the Gain ratio is also covered by the Tsallis entropy, adding a normalized factor with $q = 1$.

2.1.2 Split Search Algorithm

The most commonly used split search algorithm is the greedy search algorithm, which heuristically makes local optimal choices at each stage with the hope of finding a global optimum. The algorithm works on the tree induction as follows: (a) for each split, search all covariates; (b) for each covariate, search all values; (c) considering a pair (covariate, value) as a candidate, calculate the loss (gain) that is defined by the splitting criteria for each candidate; and (d) find the best split by searching for the minimum loss (max gain). The greedy search algorithm has the drawback that it may stuck in a local optimum. Many improvements or algorithms have been proposed to solve this issue, but, in theory, searching for the global optimum is a nondeterministic polynomial time (NP) problem. In this paper, we choose the greedy search algorithm because it is the most commonly used one.

2.2 Tree Induction via Knowledge Distillation

A simple process of knowledge distillation is shown in Figure 1. First, the complex and cumbersome black-box teacher model learns information from data. Then, the teacher model distills the learned information to knowledge. Finally, the lightweight transparent student model learns the knowledge distilled from teacher model. Researchers can modify this process to fit their particular applications. For example, in the knowledge distillation of deep neural networks the student model can be a lightweight neural network (black-box model) (Gou et al., 2021).

In this paper, we specify the knowledge distillation process (Figure 1) as follows:

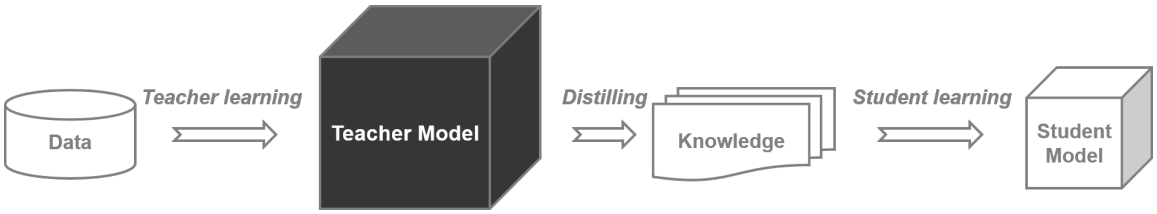

Figure 1: Knowledge distillation process

- **Data.** We denote data as $D = \{Y, X\}$, where Y is the observation of response variable y, X is the observation of covariates $\mathbf{x} = \{x_1, ...x_p\}$. Both response and covariates can be categorical or continuous variables.

- **Teacher model.** Although we usually think of the teacher model as a cumbersome black-box, it can be any model (assumption free). We use $y = f(\mathbf{x})$ to denote the teacher model.

- **Knowledge.** Since we assume nothing on the teacher model, a reasonable choice of the distilled knowledge is pseudo data randomly sampled and generated through the teacher model. We denote the sampled pseudo data as $D' = \{Y', X'\}$, where X' is the data sampled from the support of $\mathbf{x}$, and Y' is generated from the teacher model by using $Y' = f(X')$, which is the "distilling" process in Figure 1.

- **Student model.** We specify the student model as a decision tree and name it distillation decision tree (DDT). The construction of DDT is corresponding to the "student learning" process in Figure 1.

ODT is exceptionally sensitive and slight changes in data can bring about enormous changes in structure. This significantly impairs interpretability. In contrast, DDT structure has to be stable to reflect that the distilled knowledge is stable. The goal of the proposed DDT is to take its advantage of gaining interpretability without significantly sacrificing prediction accuracy. We propose gaining distilled knowledge and tree structure stability by sampling a large number of pseudo data $D' = \{Y', X'\}$ through the teacher model. Since we do not have any prior knowledge of the teacher model, a reasonable way to generate D' is to sample $\mathbf{x}$ on its support uniformly at random. This uniform sampling assumption plays a crucial role in the theoretical proofs in the next section.

3 Tree Structure Stability

We hypothesize that DDT structure will achieve stability as the sample size of pseudo data goes to infinity. This is an essential hypothesis to achieve meaningful interpretability. In Section 3.1, we justify this hypothesis by proving spit convergence under mild assumptions. Section 3.2 discusses two types of split oscillation. Section 3.3 defines the two-level stability of splitting and provides both analytical and empirical methods to measure it.

3.1 Split Convergence

Tree structure is uniquely determined by its splits. We discuss the tree structure stability by focusing on split stability. Besides random sampling, splitting criteria and the greedy search discussed in Section 2.1, the next two assumptions are also necessary for theoretical studies in this section.

(a) **Unary relationship.** To study split convergence, we assume teacher model $f(x)$ is unary, i.e., x is a single covariate. That is because only one covariate is selected in each split. However, typically, $\mathbf{f}(\mathbf{x})$ could include more than one, say p, covariates, $\mathbf{x} = \{x_1, ...x_p\}$. In this case, we can define the one-dimensional teacher model in a marginal form.

$$f(x) = f_k(x_k) = \int ... \int \mathbf{f}(\mathbf{x})\, d\mathbf{x}_{\bar{k}}, \quad k = 1, ..., p, \tag{8}$$

where $d\mathbf{x}_{\bar{k}} = \prod_{i \neq k} dx_i$. For convenience, if x_j is a categorical variable with C categories (values are $\{1, ..., C\}$), we set $\int f(\mathbf{x}_{\bar{j}}, x_j)\, dx_j = \sum_{l=1}^{C} f(\mathbf{x}_{\bar{j}}, x_j) I(x_j = l)$, where $\mathbf{x}_{\bar{j}}$ is the vector $\{x_i\}_{i \neq j}$, and $I(\cdot)$ is an indicator function.

(b) **Unique optimal split.** For every split in Section 3.1, we assume that it has a unique optimal value and call it the optimal split. Optimal split is defined in Definition 3.1 as follows:

Definition 3.1 (Optimal split). A variable x takes values in Ω. $z_i^l(x)$ and $z_i^r(x)$ are functions $\Omega \to \mathbb{R}$, where $i = 1, ..., C$ and C is a constant in $\mathbb{N}^+$. $g(z_1^t(x), ..., z_C^t(x))$ is a function $\mathbb{R}^C \to \mathbb{R}$, where $t = l$ or r.

Table 1: Cases defined by combinations of variable types

x \ y	Continuous	Categorical
Continuous	Case 1	Case 2
Categorical	Case 3	Case 4

Then, we define the optimal split x_s in Ω as follows:

$$x_s = \underset{x\in\Omega}{\operatorname{argmin}}[g(z_1^l(x), ..., z_C^l(x)) + g(z_1^r(x), ..., z_C^r(x))]. \tag{9}$$

In the teacher model $y = f(x)$, both y and x can be continuous or categorical variables. So, for each split, we need to consider four situations (Table 1).

We will prove split convergence in all cases. Lemma 3.2 is presented first for simplifying proofs in Case 1 and Case 2.

Lemma 3.2. *$y = f(x)$ is a teacher model, where x is a continuous variable and $x \in [a, b]$, where $a, b \in \mathbb{R}$. Let $z_c^l(x) = \int_a^x h_c^l(t)\,dt$, $z_i^r(x) = \int_x^b h_c^r(t)\,dt$, where $c = 1, ..., C$, C is a constant in $\mathbb{N}^+$. $h_c^l(\cdot)$ and $h_c^r(\cdot)$, are integrable functions in $[a, b]$. $g(\cdot) : \mathbb{R}^{\mathbb{C}} \to \mathbb{R}$ is the function defined in Definition 3.1. Assume x_s is a unique optimal split in (a, b) that is defined in (9).*

$n - 1$ data points are uniformly sampled at random from $[a, b]$ and we rank them in ascending order (set $x_0 = a$ and $x_n = b$), $\{x_0, x_1, ..., x_{n-1}, x_n\}$. A stump can be fitted on the pseudo data $\{(x_0, f(x_0)), (x_1, f(x_1)), ..., (x_{n-1}, f(x_{n-1})), (x_n, f(x_n))\}$ by applying the greedy split search algorithm. The split criterion is defined as follows:

$$x_s^n = \underset{x_k, k\in\{1,...,n-1\}}{\operatorname{argmin}} [g(z_1^{l(n)}(x_k), ..., z_C^{l(n)}(x_k)) + g(z_1^{r(n)}(x_{k+1}), ..., z_C^{r(n)}(x_{k+1}))], \tag{10}$$

where, $z_c^{l(n)}(x_k) = \sum_{i=1}^k h_c^l(x_i) * \Delta_i$, $z_j^{r(n)}(x_{k+1}) = \sum_{j=k+1}^n h_c^r(x_j) * \Delta_j$ and $\Delta_i = x_i - x_{i-1}, i = 1, ..., n$. Let k_s^n denote the k that minimized (10). We can get $x_s^n = x_{k_s^n}$.

Then, the following holds:

$$x_s^n \xrightarrow{p} x_s, \quad as \ \ n \to \infty.$$

The rate of convergence is $O(n^{-1})$.

See proof on page 38.

Case 1 : Both y and x are continuous variables.

Theorem 3.3 (Continuous split convergence under the criteria SSE)**.** *X is a continuous random variable taking values $x \in [a, b]$, where $a, b \in \mathbb{R}$. $y = f(x)$ is a teacher model and $f(x)$ is integrable in $[a, b]$. An unknown unique optimal split x_s in (a, b) is defined as follows:*

$$x_s = \underset{x\in(a,b)}{\operatorname{argmin}}[\int_a^x (f(t) - \mu_l(x))^2\, dt + \int_x^b (f(t) - \mu_r(x))^2\, dt], \tag{11}$$

where,

$$\mu_l(x) = \frac{1}{x-a}\int_a^x f(u)\, du \quad , \quad \mu_r(x) = \frac{1}{b-x}\int_x^b f(u)\, du.$$

Let us randomly sample $n-1$ instances of X from $[a, b]$ and rank them in an ascending order (set $x_0 = a$ and $x_n = b$), $\{x_0, x_1, ..., x_{n-1}, x_n\}$. A stump can be fitted on the pseudo data $\{(x_0, f(x_0)), (x_1, f(x_1)), ..., (x_{n-1}, f(x_{n-1})), (x_n, f(x_n))\}$ by using the greedy search algorithm and splitting criteria SSE in (1). x_s^n denotes the split of the stump.

Then, 1) x_s^n converges to x_s in probability as $n \to \infty$. 2) The values of two fitted nodes converge to $\mu_l(x_s)$ and $\mu_r(x_s)$ in probability respectively as $n \to \infty$. 3) The rate of convergence is $O(n^{-1})$.

See proof on page 40.

Case 2 : y is a categorical variable and x is a continuous variable.

Theorem 3.4 (Continuous split convergence under the Tsallis entropy criteria)**.** *X is a continuous random variable taking values $x \in [a, b]$, where $a, b \in \mathbb{R}$. $y = f(x)$ is a teacher model. $Y = f(X)$ is a discrete random variable taking values $y \in \{y_1, ..., y_C\}$, where $C \in \mathbb{N}^+$. Let $S_i = \{x | f(x) = y_i, x \in [a, b]\}$, $i = 1, ..., C$. The probability mass function of Y in $[a, b]$ is that*

$$p(y_i) = \int_{S_i} \frac{1}{b-a}\, dx, \quad i = 1, ..., C.$$

And, the probability mass function of Y in $[a, x]$ is that

$$p_{[a,x]}(y_i) = \int_{S_i \cap [a,x]} \frac{1}{x-a}\, dt.$$

Then, a Tsallis entropy can be calculated in $[a, x]$,

$$S_q([a,x]) = \frac{1}{1-q}(\sum_{i=1}^{C} p_{[a,x]}(y_i)^q - 1), \quad q \in \mathbb{R}.$$

An unknown unique optimal split x_s *in* (a, b) *can be defined as follows:*

$$x_s = \underset{x\in(a,b)}{\operatorname{argmin}}[S_q([a,x]) + S_q([x,b])]. \tag{12}$$

Let us randomly sample $n-1$ *instances of* X *from* $[a, b]$ *and rank them in ascending order (set* $x_0 = a$ *and* $x_n = b$*),* $\{x_0, x_1, ..., x_{n-1}, x_n\}$*. A stump can be fitted on the pseudo data* $\{(x_0, f(x_0)), (x_1, f(x_1)), ..., (x_{n-1}, f(x_{n-1})), (x_n, f(x_n))\}$ *by using the greedy search algorithm and splitting Tsallis entropy criteria in* (3) *and* (4)*.* x_s^n *denotes the split of the stump.*

Then, 1) x_s^n *converges to* x_s *in probability as* $n \to \infty$*. 2) The rate of convergence is* $O(n^{-1})$*.*

See proof on page 41.

Case 3 : y is a continuous variable and x is a categorical variable.

Theorem 3.5 (Categorical split convergence under MSE criteria)**.** *X is a discrete random variable taking values $x \in \{1, ..., C_x\}$, where $C_x \in \mathbb{N}^+$. Y is a continuous random variable taking values $y \in [c, d]$, where $c, d \in \mathbb{R}$. Y has a finite mean μ. $y = f(x)$ is a teacher model that defines a conditional distribution $Y|X$ with properties,*

$$E(Y|X = k) = \mu_k, \quad k = 1, ..., C_x \quad \text{and} \quad \mu = \frac{1}{C_x}\sum_{k=1}^{C_x} \mu_k. \tag{13}$$

Let us randomly sample n instances of X from $\{1, ..., C_x\}$, $\{x_1, ..., x_n\}$. The n corresponding samples of Y, $\{y_1, ..., y_n\}$, is generated according $\{x_1, ..., x_n\}$ and based on conditional distribution $Y|X$ and its properties (13). The uniform sampling assumption indicates $\lim_{n\to\infty} \frac{n_k}{n} = \frac{1}{C_x}$, where $n_k = \sum_{i=1}^{n} I(x_i = k)$, $k = 1, ..., C_x$.

Let an unknown unique optimal split x_s in $\{1, ..., C_x\}$ be defined as follows:

$$x_s = \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} \lim_{n\to\infty} \frac{1}{n}[\sum_{i=1}^{n_k}(y_{ki} - \mu_k)^2 + \sum_{l\neq k}(\sum_{j=1}^{n_l}(y_{lj} - \mu_{\bar{k}})^2)], \tag{14}$$

where $\mu_{\bar{k}} = E(Y|X \neq k)$.

A stump can be fitted on the pseudo data $\{(x_1, y_1), ..., (x_n, y_n)\}$ *by using the greedy search algorithm and splitting MSE criteria in* (2). x_s^n *denotes the split of the stump.*

Then, 1) x_s^n *converges to* x_s *in probability as* $n \to \infty$. *2) The rate of convergence is* $O(n^{-1})$.

See proof on page 41.

Case 4 : Both y and x are categorical variables.

Theorem 3.6 (Categorical split convergence under Tsallis entropy criteria)**.** *X is a discrete random variable taking values $x \in \{1, ..., C_x\}$, where $C_x \in \mathbb{N}^+$. Y is a discrete random variable taking values $y \in \{1, ..., C_y\}$, where $C_y \in \mathbb{N}^+$. $y = f(x)$ is a teacher model that defines a joint distribution (X, Y). The joint probability mass function of (X, Y) can be denoted as $p(x = i, y = j) = p_{ij}$, where $i = 1, ..., C_x, \quad j = 1, ..., C_y$.*

Let us randomly sample n instances of X from $\{1, ..., C_x\}$, $\{x_1, ..., x_n\}$. The n corresponding samples of Y, $\{y_1, ..., y_n\}$, is generated according to $\{x_1, ..., x_n\}$ and based on joint distribution (X, Y). The uniform sampling assumption indicates $\lim_{n\to\infty} \frac{n_k}{n} = \frac{1}{C_x}$, where $n_k = \sum_{i=1}^{n} I(x_i = k)$, $k = 1, ..., C_x$.

An unknown unique optimal split x_s in $\{1, ..., C_x\}$ can be defined as follows:

$$x_s = \underset{k \in \{1,...,C_x\}}{\operatorname{argmin}} [S_q(k) + S_q(\bar{k})], \tag{15}$$

where, $S_q(\cdot)$ is Tsallis entropy,

$$S_q(k) = \frac{1}{1-q}(\sum_{j=1}^{C_y}(p_{kj})^q - 1),$$

$$S_q(\bar{k}) = \frac{1}{1-q}(\sum_{j=1}^{C_y}(\sum_{i\neq k} p_{ij})^q - 1), \quad k, i \in \{1, ..., C_x\}, \quad q \in \mathbb{R}.$$

A stump can be fitted on the pseudo data $\{(x_1, y_1), ..., (x_n, y_n)\}$ *by using the greedy search algorithm and splitting Tsallis entropy criteria in* (3) *and* (4). x_s^n *denotes the split of the stump.*

Then, 1) x_s^n *converges to* x_s *in probability as* $n \to \infty$. *2) The rate of convergence is* $O(n^{-1})$.

See proof on page 42.

Based on above theorems, we can define the concepts split convergence and tree convergence as follows:

Definition 3.7 (Split convergence). For the teacher model $y = f(x)$, $x \in \Omega$, assume x_s is a unique optimal split on Ω defined in Definition 3.1. A stump T can be fitted with pseudo data $\{(x_1, y_1), ..., (x_n, y_n)\}$ sampled uniformly at random on Ω. If the split of T converges to x_s in probability as the sample size $n \to \infty$, we say that the split of T is convergent.

Definition 3.8 (Tree convergence). In a fitted DDT T, if all splits converge in probability as the uniformly sampled pseudo data size n goes to infinity. we say that the tree converge in probability as n goes to infinity.

3.2 Split Oscillation

In Section 3.1, we assume that a stump has an unique optimal split. To loosen this assumption, we assume that there is a set of optimal splits. In this case, the final split will oscillate in a set, which we call a split oscillation (Definition 3.9).

Definition 3.9 (Split oscillation). $y = f(x)$, $x \in \Omega$ is the teacher model. Assume there is a set of optimal splits on Ω. A stump T can be fitted using the pseudo data $\{(x_1, y_1), ..., (x_n, y_n)\}$ sampled uniformly at random on Ω. Then, instead of converging to a unique point, the split of T oscillates in the set of optimal splits, and even the sample size n goes to ∞. In this case, we say that the split of T is oscillating.

Some types of oscillation are identifiable under an assumption that the greedy search algorithm can randomly choose an optimal split in the set of all optimal splits. If an oscillation can be recognized, we call it an identifiable oscillation. We discuss two typical identifiable oscillations (Figure 2) that are easy to identify, and their combinations can cover most oscillations in practice. Others that are rare or meaningless in practice are not included (e.g., all of the points on Ω are optimal splits).

(a) **Finite points.** Assume there are m different optimal splits, x_{s1}, ..., x_{sm} on Ω.

(b) **An interval on a real line.** Assume $x \in [a, b]$, $a, b \in \mathbb{R}$. Any split falling into an interval $[c, d] \in [a, b]$ is an optimal split.

Because we assume that the greedy search algorithm randomly selects an optimal split, the final split follows a uniform distribution in the set of all optimal splits. In case (a), it is

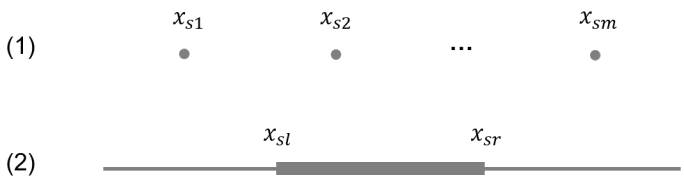

Figure 2: Two typical split oscillations

a uniform distribution with the probability mass function, $p(x_s) = p_i = 1/m$, $i = 1, ..., m$. In case (b), it is a continuous uniform distribution with the probability density function, $p(x_s) = 1/(x_{sr} - x_{sl})$, $x_s \in [x_{sl}, x_{sr}]$. So, in practice, we can execute the greedy search algorithm many times. Then, we check the consistency between the obtained empirical distribution and the theoretical ones to identity the oscillation type. There are many statistical tests available to check consistency. For example, we can perform the χ^2 test for discrete uniform distribution and the Kolmogorov-Smirnov Goodness-of-Fit test for continuous uniform distribution.

3.3 Measure of Split Stability

In Section 3.1, we proved that split convergence happens as the pseudo sample size reaches towards infinity. However, only finite samples are available in practice. In this section, we investigate split stability under the condition of finite pseudo data. We propose a measure consisting of two-level split stability and discuss ways to evaluate the stabilities either in an analytical method or through a Monte Carlo simulation.

The concept of two-level stability (Figure 3) is motivated by the greedy search algorithm.

- **First-level stability.** The first-level stability is defined by a discrete distribution with the probability mass function $p(x)$ that measures the stability of selecting covariate x_k $(k = 1, ..., p)$ as the splitting variable.

- **Second-level stability.** The second-level stability is conditional on the selected splitting covariate x_k and measures its variation. When x_k is a discrete variable, the second-level stability is defined by by a discrete distribution with the probability mass function $p(x_k)$. When x_k is a continuous variable, the second-level stability is defined by a continuous distribution with the probability density function $f(x_k)$.

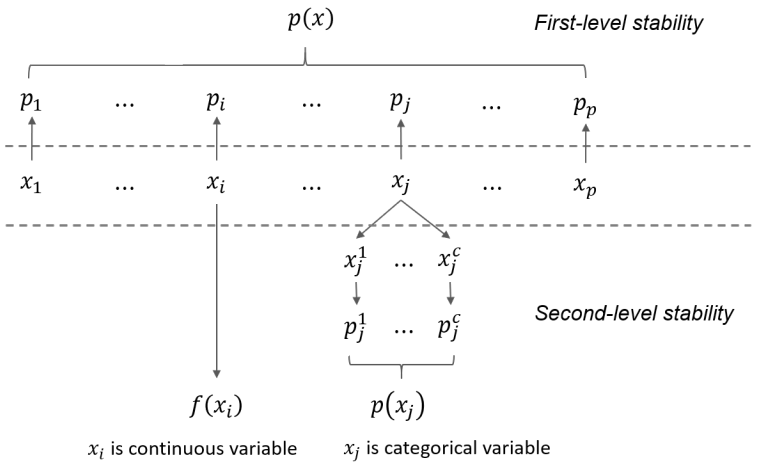

Figure 3: Illustration of two-level split stability

Note: In this section, $f(x_k)$ is used to denote a probability density function, rather than a teacher model as in Sections 3.1 and 3.2.

3.3.1 Measuring split stability in theory

Recall the proof of Lemma 3.2. Similarly, we can obtain some theoretical results of the second-level stability for continuous variables.

Proposition 3.10. x is a continuous variable taking values in $[a, b], a, b \in \mathbb{R}$. Assume x_s as an unique optimal split in (a, b). Let us sample $n-1$ data points uniformly at random in $[a, b]$. Then, we rank them in an ascending order and set $x_0 = a$ and $x_n = b$. Let x_m denote the nearest point to x_s,

$$|x_s - x_m| = min\{|x_s - x_i|\}, \quad i = 1, ..., n-1.$$

Then, the probability of $|x_s - x_m| \leq d$ with $0 < d \leq \frac{b-a}{2}$ is as follows:

$$P(|x_s - x_m| \leq d) = 1 - (1 - \frac{2d}{b-a})^n.$$

Proof. $P(|x_s - x_m| > d)$ indicates the probability of sample beyond the interval $[x_s - d, x_s + d]$. So,

$$P(|x_s - x_m| > d) = (1 - \frac{2d}{b-a})^n.$$

Then, $P(|x_s - x_m| \leq d) = 1 - (1 - \frac{2d}{b-a})^n$. □

A more useful interpretation of $P(|x_s - x_m| \leq d)$ is the probability that the true optimal split x_s falls into the interval $[x_m - d, x_m + d]$. In other words, the interval $[x_m - d, x_m + d]$ is the confidence interval of true optimal split x_s under the significant level $\alpha = 1 - (1 - \frac{2d}{b-a})^n$.

Unfortunately, the true optimal split x_s is unknown. Thus, we do not have any clue to find the true value of x_m. However, the good news is that we know the value of x_s^n. Recall the proof of Lemma 3.2. We can obtain that

$$x_s^n \xrightarrow{p} x_s, \quad x_m \xrightarrow{p} x_s \quad as \quad n \to \infty,$$

such that

$$|x_s - x_m| \xrightarrow{p} |x_s - x_s^n| \quad as \quad n \to \infty.$$

Moreover, Lemma 3.2 proved that both x_m and x_s^n have exactly the same convergence speed. So, it is reasonable to substitute x_m with x_s^n in the confidence interval $[x_m - d, x_m + d]$. Then, the interval $[x_s^n - d, x_s^n + d]$ can be used as an approximation for the $(1-\alpha) * 100\% = (1 - \frac{2d}{b-a})^n$ confidence interval of the optimal split x_s.

Let $d = \frac{3(b-a)}{2n}$. We can obtain that

$$\lim_{n\to\infty}[1 - (1 - \frac{2d}{b-a})^n] = \lim_{n\to\infty}[1 - (1 + \frac{-3}{n})^n] = 1 - e^{-3} \approx 0.950.$$

So, the interval

$$[x_s^n - \frac{3(b-a)}{2n}, x_s^n + \frac{3(b-a)}{2n}], \tag{16}$$

will be a good approximation of the 95% confidence interval of the true optimal split x_s when the sample size n is large (see the simulation study in Section 5.3). Or, given d, we can calculate the sample size that needs to approximate the 95% confidence interval of the true optimal split x_s,

$$n = \lceil \frac{3(b-a)}{2d} \rceil.$$

3.3.2 Measuring split stability via Monte Carlo simulation

It is difficult to calculate the two-levels of split stability analytically because their definition is tightly coupled with the greedy search algorithm in which some steps cannot be translated to mathematical forms. Even the method (in Section 3.3.1) that can calculate the second-level stability theoretically is difficult to apply in practice. For example, to achieve the

unary form $f(x)$, integrals need to be calculated through equation (8) if there are multiple covariates. Fortunately, Monte Carlo simulation provides a feasible way to complete the calculation.

By using Monte Carlo simulation, first-level stability can be estimated through the empirical probability mass function,

$$p(x_k) = p_k = \frac{n_k}{n}, \quad k = 1, ..., p,$$

where n is the number of Monte Carlo simulations and n_k is the times that covariate x_k is selected as a split variable.

If x_k is a categorical variable with values $\{x_k^1, ..., x_k^C\}$, the second-level stability can be estimated through the empirical probability mass function

$$p(x_k^j) = p_k^j = \frac{n_k^j}{n_k}, \quad j = 1, ..., C,$$

where n_k^j is the number of times that value x_k^j is selected as a split value.

If x_k is a continuous variable, the second-level stability can be estimated through the empirical probability density function

$$f(x_k) = density(X_k),$$

where X_k is the set of split values obtained from the Monte Carlo simulation, and $density(X_k)$ is a function that estimates the empirical probability density function from data set X_k. With $f(x_k)$, many statistics can be easily calculated to measure the stability such as variance, coefficient of variation (mean is not close to zero), and confidence interval.

4 Algorithms for Distillation Decision Tree

Theoretical results in Section 3 guarantee the structure stability of DDT as the pseudo sample size reach towards infinity if there is no oscillation. However, it is infeasible to provide infinity pseudo samples in practice. Instead, we need to ensure that the pseudo sample size is large enough for the desired stability. This leads to additional requirements to further refine the algorithms to implement DDT. In Section 4.1, we propose a sampling based tree induction algorithm. Section 4.2 introduces several sampling strategies. In Section 4.3, a marginal PCA sampling strategy is developed to overcome the curse of dimensionality.

4.1 Induction Algorithm for Constructing DDT (Algorithm 1)

Most commonly used decision trees are constructed by algorithms relying on a top-down strategy (Rokach and Maimon, 2014), which is a recursive method by induction. It starts with the entire input dataset in the root node where a locally optimal split is identified by the greedy search algorithm, and branches conditional on the split are created. This process is repeated in the created nodes unless the stopping criteria is met. Despite similarities to ODT, the algorithm of DDT induction (denoted as Algorithm 1) has its own characteristics. Algorithm 1 applies the induction process to construct DDT. In this algorithm, N_i is is chosen to be a reasonably large number (we set $N_i = 100$). The sample size n_i can be estimated in an ad-hoc way according to the two-level stability (Section 3.3) through simulation. We select the mode of second-level stability (pmf/pdf) as the value of x_s^*. The stopping criteria can either be objective (e.g., a small distance (misclassification rate or MSE) to the teacher model $f(\mathbf{x})$), or subjective (e.g., a few specified nodes designated by the researcher sufficient for application explanation). In practice, any teacher (black-box) model has an over-fitting issue. So, as an approximation of its teacher model, the only trade-off that needs to be considered in DDT construction is the degree of approximation to the teacher model versus computation load. The methods to control model complexity in ODTs, like, tree pruning, are unnecessary for DDT. Since all input data is sampled using $f(x)$, we call this algorithm a sampling-based algorithm. Two concepts related to sampling are defined next.

Definition 4.1 (Sampling space). Let $\mathbb{S}$ denote the sampling space of the teacher model $f(\mathbf{x})$, $\mathbf{x} = \{x_1, ..., x_p\}$. $\mathbb{S}$ is the cartesian product of covariate supports, $\mathbb{S} = \{(x_1, ..., x_p) | x_i \in support\ \ x_i,\ \ i = 1, ..., p\}$.

Definition 4.2 (Sampling region). For node i in DDT, the splits of its ancestors define a region in the sampling space. The region is denoting as R_i and called the sampling region of node i. All leaves' sampling regions form a partition of the sampling space.

A naive and much simpler induction algorithm is to fit an entire tree all at once on a large pseudo dataset; we name this the all-at-once algorithm. It sounds promising, but it performs poorly in practice due to the error propagation in the top-down induction algorithm. The top-down induction strategy implicitly includes dependency chains. As

Algorithm 1: Induction of DDT

Data: Randomly sampled pseudo data

Result: A distillation decision tree

Starting from the root node, set $i = 1$, and create an empty set X_s to store splits.

while *stopping criteria are not met,* **do**

1. For node i, repeat the following processes N_i times.
   (1) Generate pseudo data, which includes n_i samples from the sampling region R_i corresponding to node i.
   (2) Fit a stump on the pseudo data and put the split of the stump into X_s.
2. Compute two-level stability with X_s to identify the best split x_s^*.
3. Apply x_s^* to create child nodes and set their id as $2i$, $2i + 1$, respectively.
4. Move to next node that needs to be split.

end

Figure 4 (a) shows, the split x_{s4} is conditional (dependent) on (R_2, x_{s2}), which is again conditional on (R_1, x_{s1}). The split variance propagates along the dependency chain, but it follows the inverse direction of dependency. In Figure 4 (b), the variance Δx_{s1} will influence $(x_{s2}, \Delta x_{s2})$ and $(x_{s3}, \Delta x_{s3})$, while Δx_{s2} will again influence $(x_{s4}, \Delta x_{s4})$. Even worse, the variance will quickly inflate as it propagates to deeper levels. For example, a small Δx_{s1} may cause a big Δx_{s4} or even a split variable change. Algorithm 1 tackles this issue with the help of generating pseudo data and performing splitting repeatedly to achieve two-level stability. According Algorithm 1, for a node i we measure its split N_i times. With these repeat measures X_s, we can calculate the two-level stability and select a split value with the lowest variance. The first-level stability can reduce the variance in selection of the split variable, while the second-level stability can reduce the variance in selection of the split value. For example, if the split variable is continuous and we choose the mean of all fitted values, say $\bar{x}_s$, by the central limit theorem, the variance of $\bar{x}_s$ will be reduced at a rate of

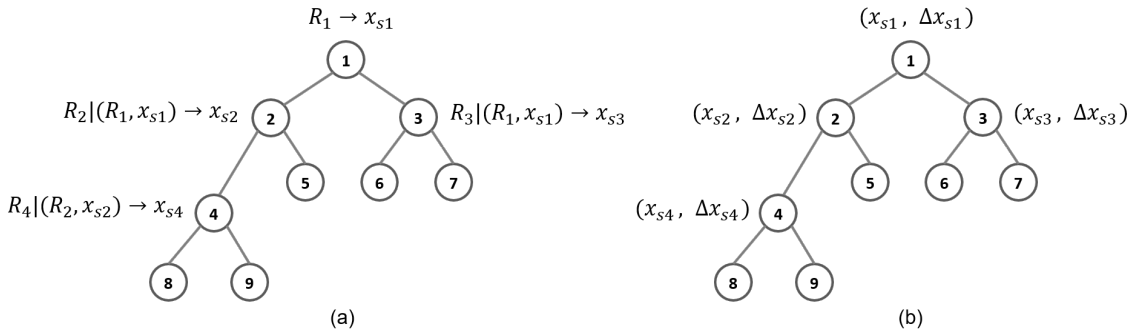

Figure 4: Examples of top-down induction and variance propagation

N_i^{-1}. Repeating this process at each split, we will obtain DDT with a stable structure.

Obviously, Algorithm 1 requires identifying a split by repeat measurement, which is computationally intensive. To maintain good prediction accuracy, it is imperative that the tree grows to a large size. Thus, it is usually computationally infeasible to grow a large DDT merely using Algorithm 1. Fortunately, only a small set of splits are required for interpretations in real applications. So, it is reasonable to create a hybrid DDT. First, we apply Algorithm 1 on a small set of specified splits. The large sample size (pseudo data) and repeated-measure strategy mentioned in Algorithm 1 will guarantee the correctness and stability of the specified splits for interpretation. Then, we apply the all-at-once algorithm to fit large sub-trees at the leaf nodes, which is done in order to maintain good hybrid DDT prediction accuracy. The small set of specified splits are called interpretable splits. For example, in Figure 5, the interpretable splits are $\{1, 2, 3, 4, 7, 14\}$. Then, we grow the large sub-trees $\{T_5, T_6, T_8, T_9, T_{15}, T_{28}, T_{29}\}$ at corresponding leaf nodes by applying the all-at-once algorithm. Hybrid DDT provides a way to decouple model interpretability and complexity. The small set of interpretable splits demonstrates the simplicity of model interpretation. The complexity that guarantees the model's prediction accuracy is handled by the big sub-trees at the bottom, which are not considered in interpretation. This decoupling provides DDT a possible way to gain good balance in the accuracy versus interpretability trade-off.

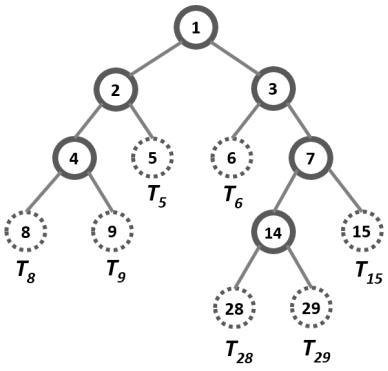

Figure 5: Hybrid distillation decision tree

4.2 Sampling Strategy

As discussed in Section 4.1, the pseudo data sampling plays a crucial role in the induction of DDT. In this section, we introduce some sampling strategies that can be embedded into the tree induction Algorithm 1 for satisfying different demands, e.g., growing a hybrid DDT or performing parallel computing.

(1) **Breadth-first sampling strategy**

Similar to the breadth-first search (BFS) algorithm (Cormen et al., 2009), breadth-first sampling starts at sampling region R_1, which corresponds to the root and conducts sampling on all present-depth sampling regions prior to moving on to regions at the next depth level. For example, at the induction of DDT in Figure 6, the breadth-first sampling strategy suggests sampling the pseudo data in the sampling regions that are ordered as $\{R_1, R_2, R_3, R_4, R_6, R_8, R_{13}\}$.

(2) **Path-based sampling strategy**

Sampling can be carried out on a route called sampling path, which is defined as follows:

Definition 4.3 (Sampling path). A sampling path is a series of nested sampling regions that are defined by the nodes in a DDT path. The sampling path $P_{i,j}$ starts from the sampling region R_i and ends at the sampling region R_j, $P_{i,j} = \{(R_i, ..., R_j) | R_i \supset ... \supset R_j\}$. Two sampling paths intersect if there exists a sampling region in one sampling path that includes any sampling region in the other sampling path.

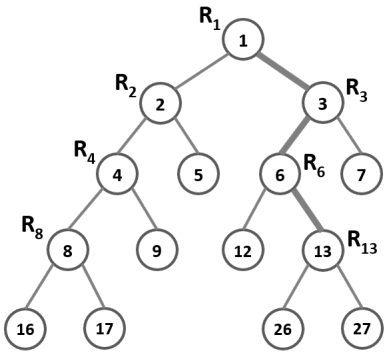

Figure 6: Sampling path and strategies

A path-based sampling strategy is used when a particular sampling region is concerned. In this case, we only grow a branch rather than an entire tree to save computations. For example, in Figure 6, we are interested in the sampling region R_{13}. Instead of growing the entire tree, we just need to grow a branch along the (bold) sampling path $P_{1,13} = \{R_1, R_3, R_6, R_{13}\}$.

(3) **Parallel sampling strategy**

Parallel sampling strategy can greatly improve the efficiency of Algorithm 1. Generally speaking, if any two sampling regions R_i, R_j cannot connect through a sampling path $P_{i,j}$, we can perform sampling in R_i and R_j parallelly. Tree structure is innately suitable for parallel computing. For example, we can apply parallel samplings in each level of the tree. In Figure 6, parallel samplings can be carried on R_2 and R_3 at level one, R_4 and R_6 at level two, or R_8 and R_{13} at level three. We can also apply this strategy with sub-trees. The sub-trees whose roots are node 2 and node 3 can grow parallelly (Figure 6). Or, we can conduct parallel sampling in the sampling paths that do not intersect (e.g., the sampling paths $P_{2,8}$ and $P_{3,13}$ in Figure 6).

4.3 Dimensionality Reduction

The hybrid tree structure and parallel sampling strategy can partially relieve the computational burden in the induction of DDT. However, the biggest computational challenge in the algorithm is the curse of dimensionality. We define the dimensionality of sampling as below.

Definition 4.4 (Dimensionality of sampling). $y = f(\mathbf{x})$ is the teacher model, where $\mathbf{x} = \{x_1, ..., x_p\}$. $\mathbb{S}$ is a p dimension sampling space that is defined by the cartesian product of $x_1, ..., x_p$. Assume that there are q continuous covariates in $\mathbf{x}$, where q $(0 \leq q \leq p)$ is a non-negative integer. Then, in $\mathbb{S}$, the dimensionality of sampling is q if $q > 0$ holds, or 1 if $q = 0$ holds.

When the dimensionality of sampling increases, the pseudo sample size grows exponentially. For instance, if there are 10 continuous covariates and we take 10 samples in each one of them, the sample size (combinations) will be 10^{10}. To tackle this issue, we propose a strategy called marginal PCA sampling. Assume that $\{x_1, ..., x_q\}$ are continuous variables and X is the $n * q$ data matrix $(n > q)$. Then, apply the PCA.

$$X^T X = W \Lambda W^T,$$

where

$$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_q \end{bmatrix}, \quad \lambda_1 \geq, ..., \geq \lambda_q, \quad WW^T = I, \quad I \textit{ denotes the identity matrix.}$$

Let $\mathbf{z} = \mathbf{x}W$. We can transfer the covariates $\mathbf{x} = \{x_1, ..., x_q\}$ to new covariates $\mathbf{z} = \{z_1, ..., z_q\}$. Then, the reduction of sampling dimensionality can be achieved by applying PCA in the space of $\mathbf{z}$. The eigen values $\lambda_1^2, ..., \lambda_q^2$ reflect the variance explained by $z_1, ..., z_q$, and the proportion of variance explained can be calculated such that

$$p_i = \frac{\lambda_i^2}{\sum_{j=1}^{q} \lambda_j^2}, \quad i = 1, ..., q.$$

Set a criteria c, $0 < c \leq 1$. Then, we can calculate the cutoff t, $1 \leq t \leq q$ as follows:

$$\sum_{i=1}^{t} p_i = \sum_{i=1}^{t} \frac{\lambda_i^2}{\sum_{j=1}^{q} \lambda_j^2} \geq c, \quad i = 1, ..., q.$$

Using t we truncate the vector $\{z_1, ..., z_q\}$ and just consider the first t covariates that can explain the percentage of variance equal to or greater than the criteria c. Then, we perform sampling in the space of $\mathbf{z}$ following the ratio (17).

$$\lceil \frac{p_1}{p_t} \rceil, ..., \lceil \frac{p_{t-1}}{p_t} \rceil, \frac{p_t}{p_t}, 1, ..., 1 \tag{17}$$

For example, let $c = 90\%$ and $p_1 = 0.6$, $p_2 = 0.2$, $p3 = 0.1$, p_4, ..., p_q. We know that $t = 3$ and the ratio is such that

$$6, 2, 1, 1, ..., 1.$$

If we sample n points in the supports of z_1, then, $\lfloor n/3 \rfloor$ and $\lfloor n/6 \rfloor$ points are needed for z_2 and z_3, respectively. We do not care much about z_4, ..., z_q, and we just give them random values in their supports because they only explain a 10% variance. The sampled data can be organized as follows:

$$\begin{matrix} z_{1,1} & z_{2,1} & z_{3,1} & \cdots \\ z_{1,2} & z_{2,2} & z_{3,1} & \cdots \\ z_{1,3} & z_{2,1} & z_{3,1} & \cdots \\ z_{1,4} & z_{2,2} & z_{3,1} & \cdots \\ z_{1,5} & z_{2,1} & z_{3,1} & \cdots \\ z_{1,6} & z_{2,2} & z_{3,1} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ z_{1,n} & z_{2,\lfloor n/3 \rfloor} & z_{3,\lfloor n/6 \rfloor} & \cdots \end{matrix}$$

In this case, the sample size is reduced from n^q to $n^3/18$. This reduction will be remarkable when q is large.

Let Z_s denote the data matrix sampled from the space of $\mathbf{z}$ using the marginal PCA sampling. Then, we convert Z_s back to the space of $\mathbf{x}$ and denote as X_s.

$$X_s = Z_s W^T$$

The marginal PCA sampling is described in Algorithm 2.

There is a challenge in step 2 that we have to identify the joint support of $\mathbf{z} = \{z_1, ..., z_q\}$. Differing from the joint support of $\mathbf{x}$, which is a p dimension hypercube, the joint support of $\mathbf{z}$ can be complicated. For example, in two-dimensional case, the PCA transformation is such that

$$[z_1, z_2] = [x_1, x_2] \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}, \quad x_1 \in [a_1, b_1], \quad x_2 \in [a_2, b_2].$$

Algorithm 2: Marginal PCA Sampling

Data: Real data X

Result: X_s

1. **If** root node, **then**

   do principal component analysis on X to find the matrix W.

   else

   randomly sample a moderate number of data in the sampling region in the space of X, then convert them into space of Z by matrix W.

2. Perform sampling in the joint support of $\mathbf{z}$ with the ratio (17) to obtain Z_s.

3. Convert Z_s to X_s by $X_s = Z_s W^T$.

The joint support of $[z_1, z_2]$ is

$$\begin{cases} a_1 \leq \frac{w_{22}z_1 - w_{21}z_2}{w_{22}w_{11} - w_{21}w_{12}} \leq b_1 \\ a_2 \leq \frac{w_{12}z_1 - w_{11}z_2}{w_{12}w_{21} - w_{11}w_{22}} \leq b_2. \end{cases}$$

An easy, but inefficient way to meet this challenge is to add a filter in Algorithm 2. First, we sample Z_s on the hypercube support of $\mathbf{z}$. Then, convert Z_s to X_s through $X_s = Z_s W^T$. Finally we filter X_s by the true support of $\mathbf{x}$,which is defined by the real data. The good news is that the filtering process is very fast. We name this modified approach the marginal PCA filter sampling algorithm (Algorithm 3).

5 Simulated Case Studies and Analysis on Real Data

Simulation studies are conducted in this section to verify the algorithms and theoretical results. In Section 5.1, we construct a (hybrid) DDT by applying Algorithm 1 on two simulated datasets. The tree structure stability and power of exploring complex structures in data are compared between DDT and ODT (ordinary decision tree). In Section 5.2, DDT is applied to analyze a real dataset. We also compare the structure stability between DDT and ODT, and we find that bootstrap and model aggregation can effectively increase split

Algorithm 3: Marginal PCA Filter Sampling

Data: Real data X

Result: X_s

1. **If** root node, **then**

   do principal component analysis on X to find the matrix W.

   else

   randomly sample a moderate number of data in the sampling region in the space of X, then convert them into the space of Z by matrix W.

2. Perform sampling in the hypercube support of $\mathbf{z}$ with the ratio (17) to obtain Z_s.

3. Convert Z_s to X_s by $X_s = Z_s W^T$.

4. Filter X_s by the true support of $\mathbf{x}$.

stability. Moreover, we discuss the accuracy versus interpretability trade-off and illustrate that DDT is able to reach a good balance between the two. Section 5.3 validates the theoretical results in Section 3.3.1. Finally, in Section 5.4, the efficiency of marginal PCA filter sampling (Algorithm 3) is exhibited by an example.

5.1 Case Studies on Simulated Data

To provide informative illustrations, we simulate two kinds of datasets. The one-dimensional data that has just one covariate offers convenience for stability comparison between ODT and DDT. The two-dimensional data has two covariates and a complex structure for examining DDT's ability to explore complex structure in data.

5.1.1 Study on one-dimensional data

The simulation data is generated from a piecewise function $y = f(x)$ with added random errors ϵ. The random error ϵ follows a normal distribution with 0 mean and 0.1 standard deviation. There are seven breakpoints corresponding to the true splits. To compare the

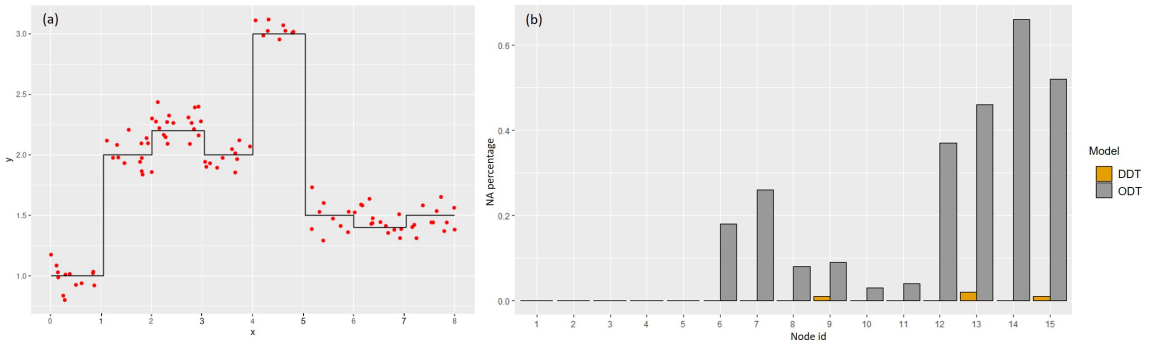

Figure 7: One-dimensional data simulation: (a) Simulated data, (b) first-level stability, and (c) second-level stability.

stability between DDT and ODT, we repeat the simulation 100 times. In each simulation, procedures are as follows:

(1) Generate simulation data. Figure 7 (a) illustrates an example of the simulation data.

(2) Fit an ODT model and a random forest model on the same generated data, respectively.

(3) Construct DDT from the fitted random forest model by applying Algorithm 1.

In the end, we obtain 100 DDT and ODT, respectively.

In this example, we want to compare the two-level stability between ODT and DDT. Since there is just one covariate x, the first-level stability can be measured by the percentage of missing split that corresponds to the percentage of NA (missing split) value. Figure 7 (b) indicates that DDT has better performance compared to ODT in the first-level of stability. Although the split's absence in deeper levels of the tree may be reasonable on the small dataset, we pay attention to the split stability, and a pruning operation can be applied to avoid meaningless splits in the future. Figure 8 shows the second stability comparison between DDT and ODT by the distributions of the splits in node 1 to node 15. It is obvious that DDT is more stable than ODT. Moreover, DDT's splits tend to concentrate near the true splits. This observation indicates that DDT is more likely to identify the true splits.

By using the dataset shown in Figure 7 (a), we can compare the structure of ODT and DDT, which is presented in Figure 9. Satisfactorily, both of them could correctly identify the true splits. The true splits $x = 6$ and $x = 7$ are difficult to identify, because the generating

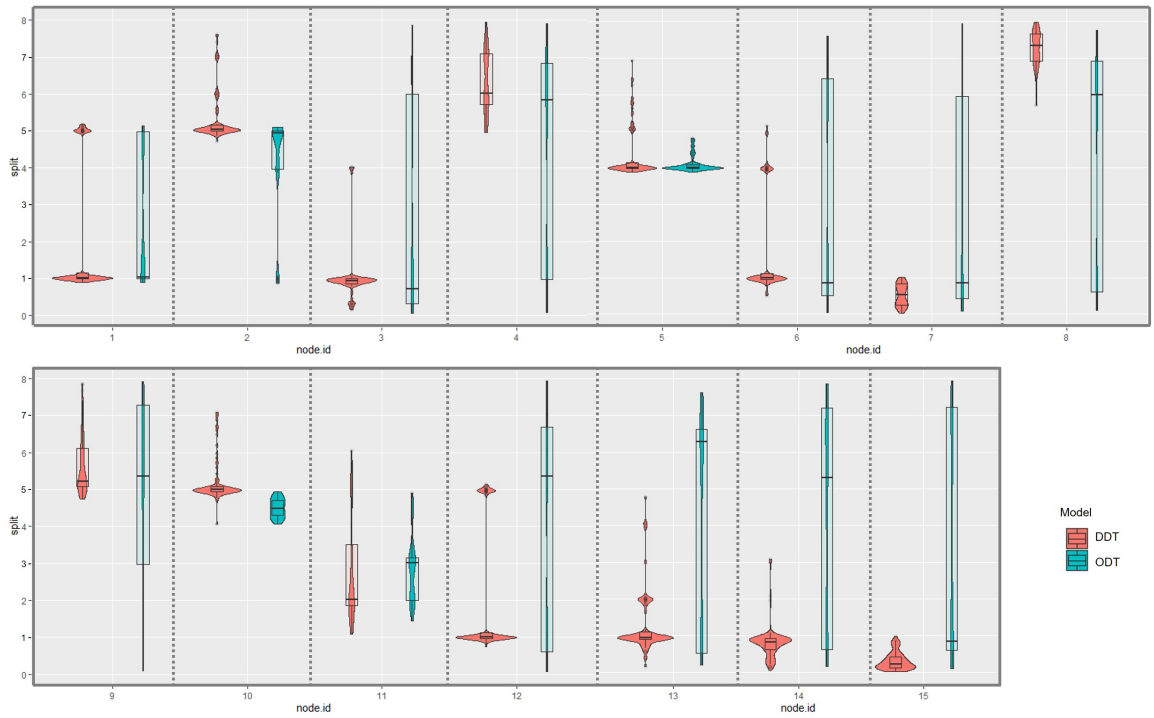

Figure 8: One-dimensional data simulation: The second-level stability.

function only has small jumps at these splits, which is equal to the standard deviation of error term. Between true splits $x = 5$ and $x = 6$, ODT has one internal split around $x = 6.3$ while DDT has two internal splits around $x = 5.3$ and $x = 6.3$. Similarly, between true splits $x = 6$ and $x = 7$, ODT has one internal split around $x = 7.3$ while DDT has two internal splits around $x = 7.3$ and $x = 6.6$. These indicate that DDT has the ability that finds more subtle structures in the data compared to ODT. Although the subtle structures near split $x = 6$ and $x = 7$ are actually caused by noise in this example, we should not simply judge this ability as bad. Conversely, DDT's ability to identify subtle structure is desired in modern data analysis where such exploration may reveal important underlying data generation mechanisms. Determining whether the subtle structures found by DDT make sense in practice will require future information and validation studies. Intrinsically, One part comes from the power of the teacher (ML black-box) model in capturing the complex data structure. Another part comes from the approximation capability. The second-level split stability corresponding to the true splits is shown in Figure B.17 in Appendix B.

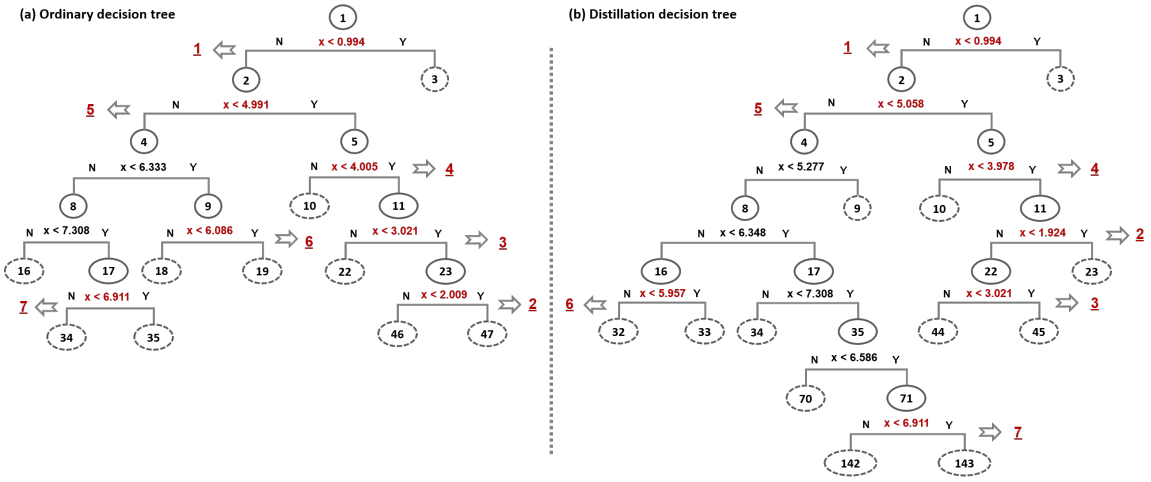

Figure 9: One-dimensional data simulation: Capability of identifying the structure of data

5.1.2 Study on two-dimensional data

The above one-dimensional example focuses on DDT's structure stability. To demonstrate DDT's power in exploring complex structure in the data, we test its use on a dataset with complex structure. Figure 10 (a) presents a two-dimension function $y = f(x_1, x_2)$ that is highly nonlinear and has complex interactions. This function serves as a data generating process. A dataset with 50 observations (Figure 10 (b)) is randomly sampled from this process. An ODT (Figure 10 (d)) and a random forest (Figure 10 (e)) are fitted on the dataset. A hybrid DDT (Figure 10 (f)) is constructed based on the random forest model. Even with a relatively small dataset (50 observations), DDT could explore more complex and meaningful structures of the true population (function) than ODT could. Essentially this power comes from DDT's ability to closely approximate the teacher model, which performs well in prediction. Meanwhile, DDT can remedy the teacher model with strong and accurate interpretability. With the true function, we could get its optimal seven parts partition (Figure 10 (a)) by the top three levels of a decision tree whose structure is demonstrated in Figure 10 (c). The ODT and hybrid DDT that are constructed based on the 50 observations (Figure 10 (b)) could also offer a seven parts partition, respectively. Similarly, there are seven comparable sets of split values of the three models. We compare them in Table B.2 in Appendix B. From Figure 10 and Table B.2, we find that (hybrid) DDT splits are closer to the true optimal ones compared to ODT splits. Especially, in the area

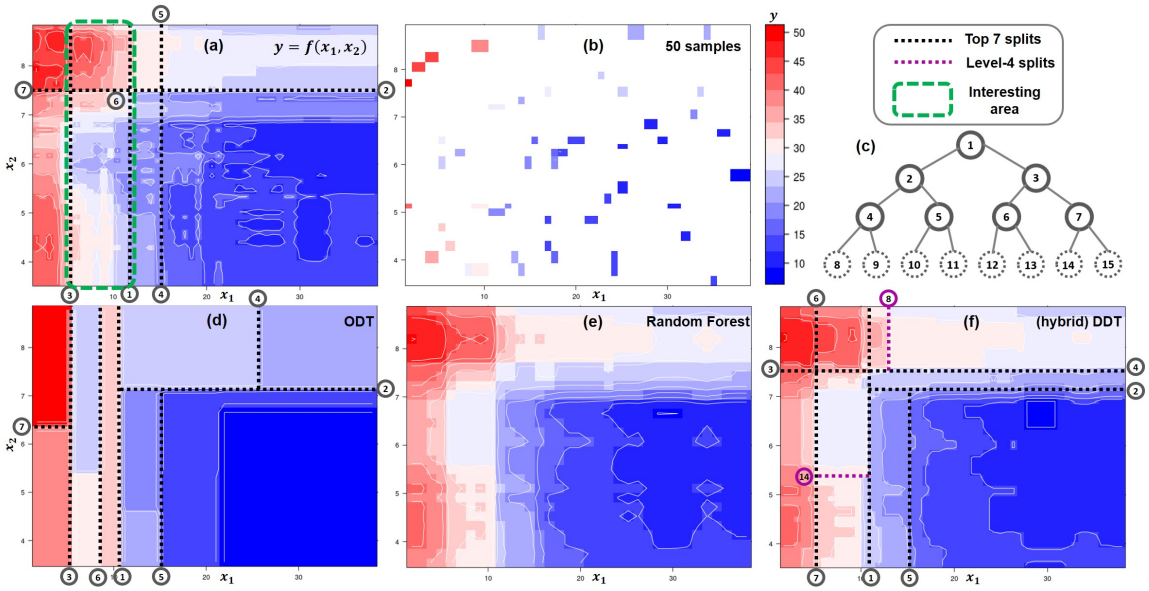

Figure 10: Two-dimensional data simulation: The power of exploring and interpreting complex structures of data

where complex interactions and non-linearity exist. The (hybrid) DDT can provide more robust and accurate explanations. Moreover, a Monte Carlo simulation based empirical pdf for each split can be constructed (Figure B.18 in Appendix B) to show the second-level stability. We do not show the empirical pmf for the first-level stability because the empirical pmfs are 100% selecting x_1 at splits 1, 5, 6, 7 and selecting x_2 at splits 2, 3, 4.

5.2 Analysis on real data

The BostonHousing dataset in R package mlbench (Leisch and Dimitriadou, 2021) is used to evaluate the performance of ODT and DDT. This dataset includes 506 observations and 14 variables. The variables are listed as follows

medv: median value of owner-occupied homes in USD 1000's.

crim: per capita crime rate by town.

zn: proportion of residential land zoned for lots over 25,000 sq.ft.

indus: proportion of non-retail business acres per town.

chas: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).

nox: nitric oxides concentration (parts per 10 million).

rm: average number of rooms per dwelling.

age: proportion of owner-occupied units built prior to 1940.

dis: weighted distances to five Boston employment centres.

rad: index of accessibility to radial highways.

tax: full-value property-tax rate per USD 10,000.

ptratio: pupil-teacher ratio by town.

b: $1000(B - 0.63)^2$ where B is the proportion of blacks by town.

lstat: percentage of lower status of the population.

Variable "medv" is the response variable. Random forest is selected as the teacher (black-box) model. In this evaluation, we first compare the structure stability of DDT and ODT through bootstrapping. Then, the prediction accuracy and interpretability of DDT are examined to illustrate that DDT is able to strike a good balance in trade-off of prediction accuracy and interpretability.

5.2.1 Comparison of structure stability

We compare the structure stability between ODT and DDT in four types of models. Simulation datasets are generated from the original BostonHousing dataset through bootstrap method. The experiment design is shown in Figure B.19 in Appendix B. The model ODT i and DDT i, $i = 1, ..., 50$, are fitted on i-th bootstrapping dataset. The model a1.DDT i, $i = 1, ..., 50$, is fitted on 10 datasets, which are bootstrapped on the i-th bootstrapping dataset. The model a2.DDT i, $i = 1, ..., 50$, is fitted on 10 datasets that are bootstrapped directly on the BostonHousing dataset.

There are 13 covariates in the BostonHousing dataset. We can learn the first-level stability through their empirical probability mass function of the covariates chosen by the split nodes. The category NA, i.e. missing split, also needs to be included in the pmf. Figure 11 presents the results. It is difficult to provide a universally accepted definition for the first-level stability based on the pmf. However, there are two reasonable criteria. In the pmf, (a) the fewer number of categories (i.e., covariates chosen) the more stable, and (b)

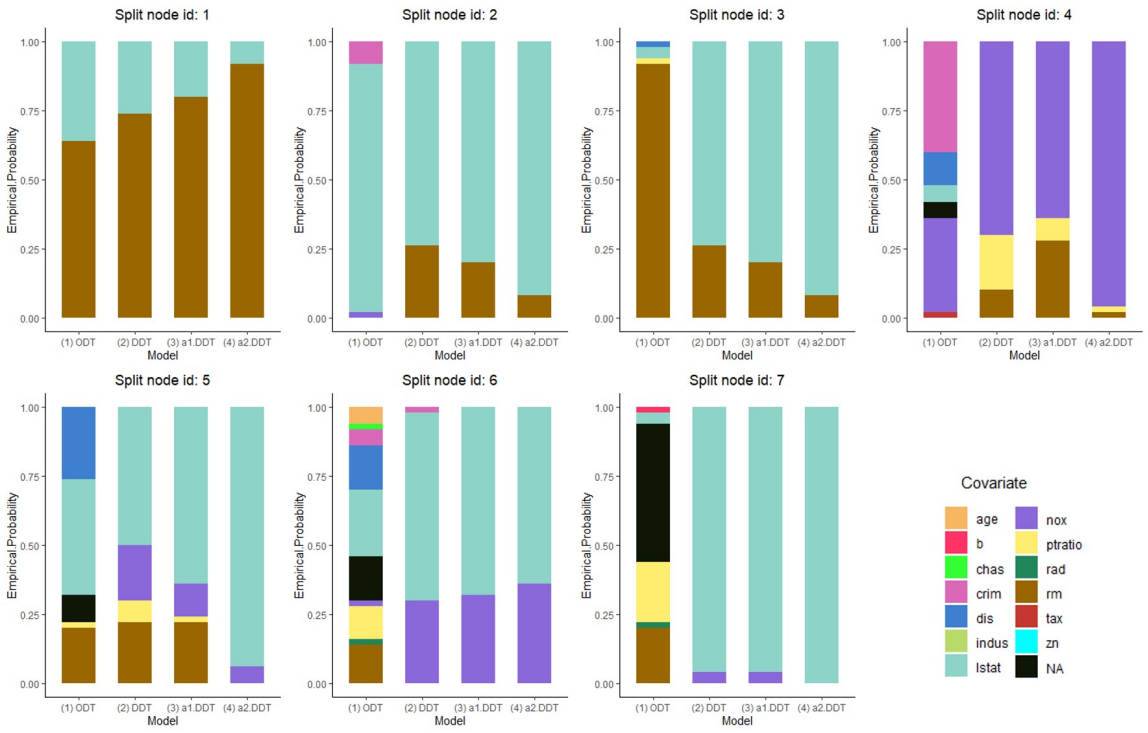

Figure 11: BostonHousing dataset: First-level stability of four methods illustrated by the empirical probability mass function of covariates chosen by the split nodes.

for each category, the higher percentage the more stable. The tricky part is deciding which criterion to use based on the project being studied. For example, according to criterion (a), DDT has the same stability at split 1 and higher stability at all other splits compared to ODT. However, based on criterion (b), DDT has lower stability than ODT at split 2 and 3. In this case, we consider the criteria (b) dominates (a) and ODT is more stable than DDT. If the first-level stability is similar (e.g., DDT and ODT at split 1), we can compare their second-level stability, which is shown in Figure 12. It is obvious that DDT is more stable than ODT for split variable "rm" at split 1.

Essentially, a part of DDT's uncertainty is inherited from the teacher model (i.e., the random forest) rather than from DDT. Because DDT is just an approximation of the teacher model. We can reduce this uncertainty by generating DDT from an aggregated teacher model (i.e., the average of 10 random forest models). Figure B.19 shows two types of aggregation. As Figure 11 shows, generally speaking, aggregation can effectively improve first-level stability.

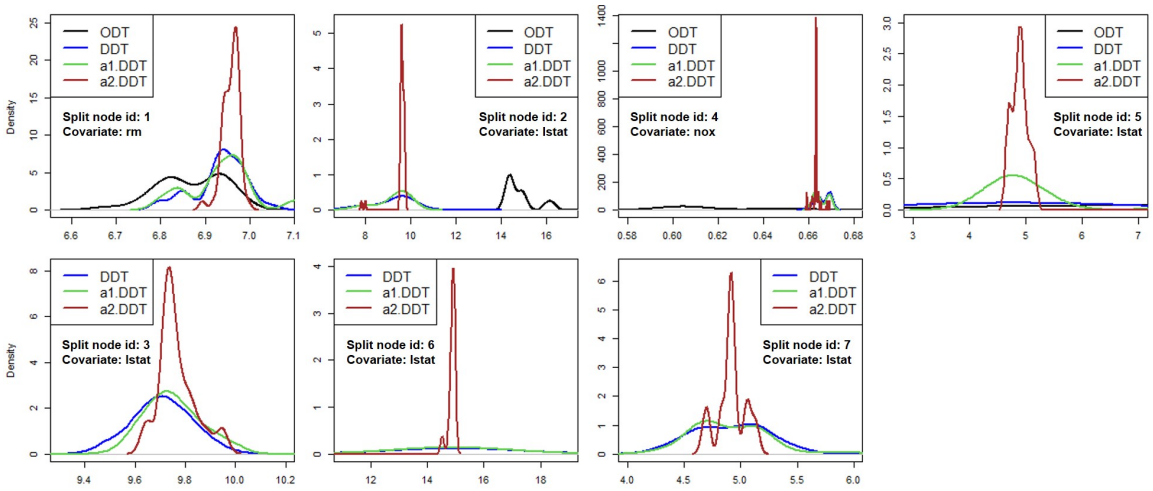

Figure 12: BostonHousing dataset: Second-level stability illustrated by the probability density function of the covariates chosen by split node 1 to 7.

Second-level stability depends on first-level stability. Therefore, we compare the second-level stability when the models have comparable first-level stability. In Figure 12, we compare the second-level stability first among all models at (split node: 1, split variable: rm), (split node: 2, split variable: lstat), (split node: 4, split variable: nox) and (split node: 5, split variable: lstat), then among the models DDT, a1.DDT, and a2.DDT at (split node: 3, split variable: lstat), (split node: 6, split variable: lstat) and (split node: 7, split variable: lstat) because of comparable first-level stability. Figure 12 indicates that DDT performs better in second-level stability than ODT in most cases. Moreover, aggregation also can improve the second-level stability, especially for aggregating on the bootstraps of the original dataset (a2.DDT).

5.2.2 Prediction accuracy

The prediction performance is compared through a tenfold cross-validation. The results are shown in Figure 13. There are two key observations. The first is that the red line (DDT) has a nearly identical shape to the green line (random forest). The second is that the prediction accuracy of DDT lies between that of random forest and ODT. These observations demonstrate that DTT has better prediction accuracy than the ODT, and it is a good approximation to the random forest (teacher) model.

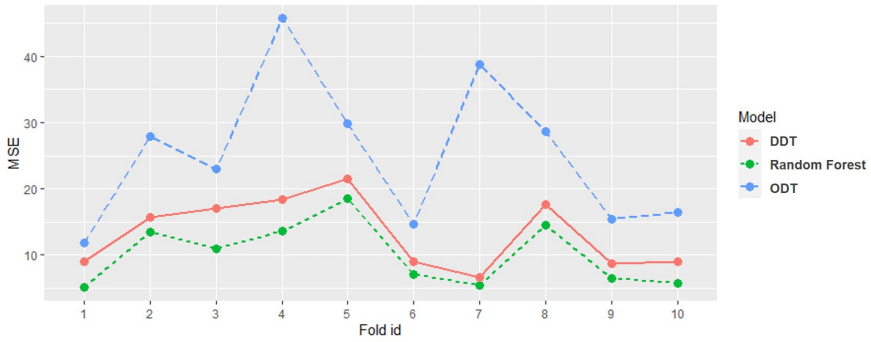

Figure 13: BostonHousing dataset: Comparison of prediction accuracy of DDT, random forest, and ODT.

5.2.3 Interpretability

In Figure 14 (a), a hybrid DDT is distilled from a random forest (teacher) model that is trained with the BostonHousing dataset. We constrain the interpretable nodes in the top two levels of the tree. Fitted on the same dataset, ODT is shown in Figure 14 (b). These two models show the two different interpretations. Simply based on the resulting models, we cannot say that one tree is better than the other tree. Each model can be explained by different theories. However, without considering structure stability, interpretations are dubious. In Figure 14 (a), there are seven splits for interpretation, and all of them have empirical probability—one in their first-level stability. Their second-level stability (i.e., empirical probability density functions) is shown in Figure B.20 in Appendix B. We can claim that DDT can provide credible interpretations to the random forest model. The credibility is guaranteed by the stability of the tree structure. Moreover, DDT gains interpretability without sacrificying much of the teacher model (i.e., random forest) prediction accuracy. This indicates that DDT strikes a good balance point in the trade-off between prediction accuracy and interpretability.

5.3 Simulations justify theoretical results

In Section 3.3.1, we discussed the second-level stability of the continuous variable and proposed a theoretical 95% confidence interval for the true optimal split. To justify the theoretical results, we design an experiment that includes a step function $f(x)$ and a unique

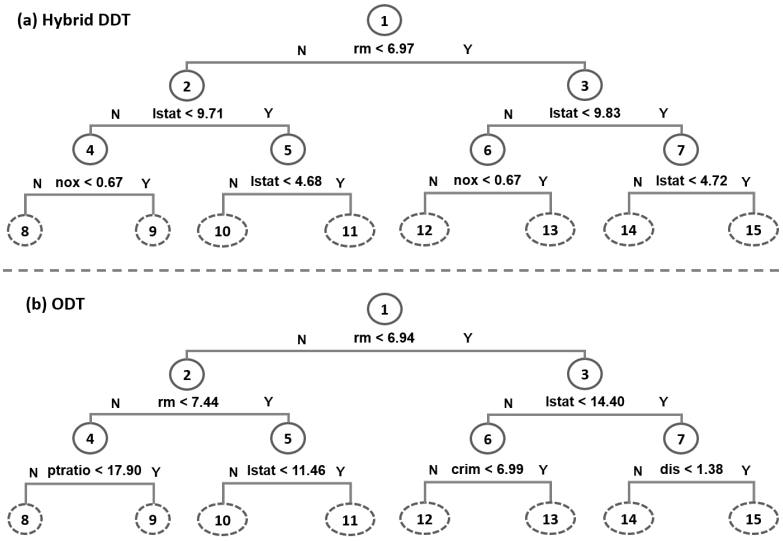

Figure 14: BostonHousing dataset: Interpretation (Hybrid DDT versus ODT)

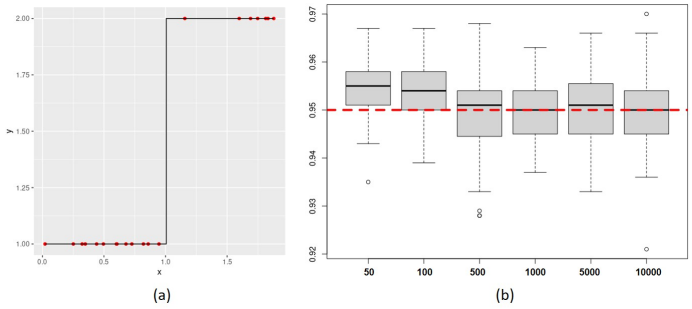

Figure 15: (a) The true split value is at $x = 1$. (b) The coverage (true split value) rate of 95% confidence intervals is converging to the theoretical value 0.95.

optimal split at $x = 1$, as shown in Figure 15 (a). The experiment is conducted as follows: For each sample size, 1) calculate theoretical 95% confidence interval 1000 times by (16), then obtain the coverage rate of the true optimal split $x = 1$ through the 1000 intervals. 2) Repeat the first step 100 times to get 100 coverage. The result is shown in Figure 15 (b), showing that the empirical expectation of coverage is around 95%, which is consistent with the theoretical 95% confidence interval when the sample size is 500 or larger. This result empirically justifies the confidence interval (16) and Lemma 3.2.

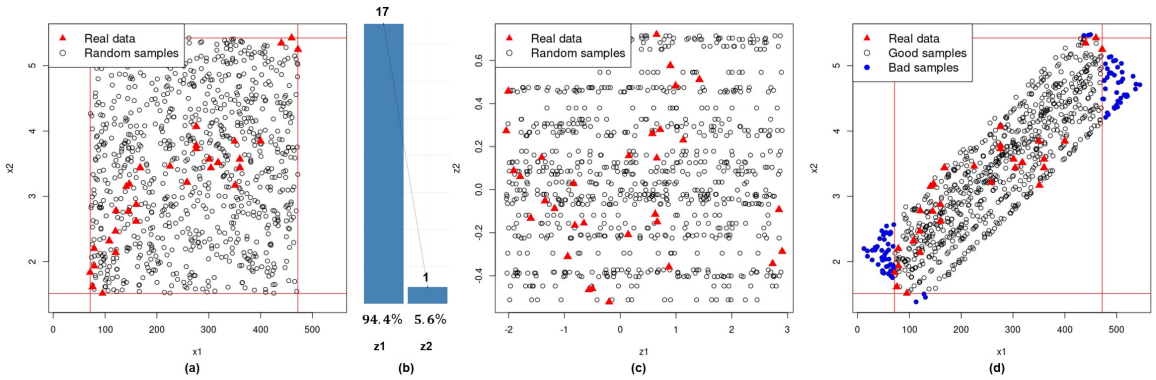

Figure 16: Random sampling versus marginal PCA filter sampling

5.4 Marginal PCA Filter Sampling

In this section, with a simulation, we show that marginal PCA filter sampling is more efficient compared to random sampling. Figure 16 presents some simulation results. In Figure 16 (a), the red triangles and black open circles represent real data and random sampling data, respectively. Random samples spread the sample space to the entire joint support of x_1 and x_2. After PCA analysis of the real data, we can obtain a sampling ratio of 17 : 1 by identifying the two principal components and mapping the original (x_1, x_2) space to the principal component (z_1, z_2) space (Figure 16 (b)). Following this sampling ratio, we conduct random sampling in space (z_1, z_2) (Figure 16 (c)). Then, the samples in space (z_1, z_2) are mapped back to space (x_1, x_2), and a part of them (the blue dots in Figure 16 (d)) are filtered out because they do not locate in the joint support of x1 and x2 defined by the real data. Comparing Figure 16 (a) and (d), it is obvious that PCA filter sampling is more efficient than random sampling.

6 Conclusion

Bénard et al. (2020) defined simplicity, stability, and predictivity as minimum requirements for interpretable models. First, DDT keeps simplicity by decoupling interpretable splits and complex subtrees in Section 4.1 (Hybrid DDT), because we just use the small set of interpretable splits for interpretation. Second, the stability requirement is clearly satisfied in DDT by the theoretical and practical studies in this paper. Finally, for the predictivity

requirement, Breiman (2001) claimed that a significant decrease of predictive accuracy compared to a state-of-the-art black-box model indicates the interpretable model misses some structures/patterns in the data and is therefore misleading. As an approximation of the black-box model, DDT retains the good prediction performance of that model, which provides understanding of the complex structures/patterns of data. In summation, DDT can serve as an interpretable simplified version of the modern machine learning black-box models and greatly expand the frontiers of their application.

A Proofs

Proof of Lemma 3.2. There must exist a point x_m such that,

$$|x_s - x_m| = min\{|x_s - x_i|\}, \;\; i = 1, ..., n-1.$$

Because $x_i, \;\; i = 1, ..., n-1$ are uniformly distributed in (a, b). For a constant $\epsilon, \;\; 0 < \epsilon < b - a$, by the theory of order statistics, it is easy to prove the following:

$$P(|x_s - x_m| > \frac{\epsilon}{2}) = (1 - \frac{\epsilon}{b-a})^n.$$

For any $\epsilon, \;\; 0 < \epsilon < b - a$,

$$\lim_{n\to\infty} P(|x_s - x_m| > \frac{\epsilon}{2}) = \lim_{n\to\infty} (1 - \frac{\epsilon}{b-a})^n = 0. \tag{18}$$

In other words, $x_m \xrightarrow{p} x_s$ as $n \to \infty$.

For any two consecutive points x_i and x_{i+1}, $i = 1, ..., n$, and a constant $\epsilon, \;\; 0 < \epsilon < b-a$, we have that

$$P(|x_i - x_{i+1}| > \frac{\epsilon}{2}) = (1 - \frac{\epsilon}{b-a})^{n-1}.$$

Similar to (18), we can prove that $x_i \xrightarrow{p} x_{i+1}$ as $n \to \infty$. This also implies that $\Delta_i \xrightarrow{p} 0$ as $n \to \infty$ for $i = 1, ..., n$.

Because $h(\cdot)$ is integrable in $[a, b]$ and $x_m \xrightarrow{p} x_s$, $\Delta_i \xrightarrow{p} 0$, as $n \to \infty$, we can get that

$$\lim_{n\to\infty} z_c^{l(n)}(x_m) = \lim_{n\to\infty} \sum_{i=1}^{m} h_c^l(x_i) * \Delta_i \xrightarrow{p} \int_a^{x_s} h_c^l(u)\, du = z_c^l(x_s). \tag{19}$$

Similarly, and by $x_{m+1} \xrightarrow{p} x_m \xrightarrow{p} x_s$, we can prove that,

$$\lim_{n\to\infty} z_c^{r(n)}(x_{m+1}) \xrightarrow{p} z_c^r(x_s).$$

Then, we can obtain that

$$\begin{aligned} &\lim_{n\to\infty} [g(z_1^{l(n)}(x_m), ..., z_C^{l(n)}(x_m)) + g(z_1^{r(n)}(x_{m+1}), ..., z_C^{r(n)}(x_{m+1}))] \\ &\xrightarrow{p} g(z_1^l(x_s), ..., z_C^l(x_s)) + g(z_1^r(x_s), ..., z_C^r(x_s)). \end{aligned} \tag{20}$$

Recall (10) and assume that

$$x_s^n \xrightarrow{p} x_s^*, \quad as \;\; n \to \infty.$$

Because $x_s^n = x_{k_s^n}$ and $x_{k_s^n+1} \xrightarrow{p} x_{k_s^n}$ as $n \to \infty$, we know that

$$x_{k_s^n} \xrightarrow{p} x_s^* \quad , \quad x_{k_s^n+1} \xrightarrow{p} x_s^*.$$

Using the integrability of $h(\cdot)$ and the same proof procedures of (19) and (20), we can obtain the following:

$$\begin{aligned} &\lim_{n\to\infty} [g(z_1^{l(n)}(x_{k_s^n}), ..., z_C^{l(n)}(x_{k_s^n})) + g(z_1^{r(n)}(x_{k_s^n+1}), ..., z_C^{r(n)}(x_{k_s^n+1}))] \\ &\xrightarrow{p} g(z_1^l(x_s^*), ..., z_C^l(x_s^*)) + g(z_1^r(x_s^*), ..., z_C^r(x_s^*)). \end{aligned} \tag{21}$$

According to the greedy search algorithm and split criterion in (10), for all $n \in \mathbb{N}$, we know that

$$\begin{aligned} &g(z_1^{l(n)}(x_m), ..., z_C^{l(n)}(x_m)) + g(z_1^{r(n)}(x_{m+1}), ..., z_C^{r(n)}(x_{m+1})) \\ &\geq g(z_1^{l(n)}(x_{k_s^n}), ..., z_C^{l(n)}(x_{k_s^n})) + g(z_1^{r(n)}(x_{k_s^n+1}), ..., z_C^{r(n)}(x_{k_s^n+1})). \end{aligned} \tag{22}$$

The equal sign holds if and only if $m = k_s^n$.

From (20), (21), and (22) we can get,

$$g(z_1^l(x_s), ..., z_C^l(x_s)) + g(z_1^r(x_s), ..., z_C^r(x_s)) \geq g(z_1^l(x_s^*), ..., z_C^l(x_s^*)) + g(z_1^r(x_s^*), ..., z_C^r(x_s^*)).$$

Since x_s is the unique and optimal split that satisfies the split criterion (9), the following must hold:

$$g(z_1^l(x_s), ..., z_C^l(x_s)) + g(z_1^r(x_s), ..., z_C^r(x_s)) \leq g(z_1^l(x_s^*), ..., z_C^l(x_s^*)) + g(z_1^r(x_s^*), ..., z_C^r(x_s^*)).$$

So,

$$g(z_1^l(x_s), ..., z_C^l(x_s)) + g(z_1^r(x_s), ..., z_C^r(x_s)) = g(z_1^l(x_s^*), ..., z_C^l(x_s^*)) + g(z_1^r(x_s^*), ..., z_C^r(x_s^*)),$$

and

$$x_s^n \xrightarrow{p} x_s^* = x_s, \quad as \ \ n \to \infty.$$

For the rate of convergence, let's recall (18) and change ϵ to $\frac{\epsilon'}{n}$, where ϵ' is a constant in $(0, b-a)$.

$$\lim_{n\to\infty} P(|x_s - x_m| > \frac{\epsilon'}{2n}) = \lim_{n\to\infty} (1 + \frac{-\epsilon'/(b-a)}{n})^n = e^{-\epsilon'/(b-a)}$$

Because $e^{-\epsilon'/(b-a)}$ is a constant, the convergence rate of x_m is $O(n^{-1})$.

Since $|x_s - x_s^n| \geq |x_s - x_m|$ always holds x_s^n converges to x_s slower or equal to x_m. However, by (22), we know that

$$g(z_1^{l(n)}(x_{k_s^n}), ..., z_C^{l(n)}(x_{k_s^n})) + g(z_1^{r(n)}(x_{k_s^n+1}), ..., z_C^{r(n)}(x_{k_s^n+1}))$$

converges to

$$g(z_1^l(x_s), ..., z_C^l(x_s)) + g(z_1^r(x_s), ..., z_C^r(x_s))$$

faster or equal than

$$g(z_1^{l(n)}(x_m), ..., z_C^{l(n)}(x_m)) + g(z_1^{r(n)}(x_{m+1}), ..., z_C^{r(n)}(x_{m+1}))$$

in all instances. This implies x_s^n converges to x_s faster or equal than x_m.

So, the convergence rate of x_s^n is exactly the same as x_m, and it is at the level $O(n^{-1})$ too. □

Proof of Theorem 3.3. Construct that

$$g(z_c^l(x)) = z_c^l(x), \quad g(z_c^r(x)) = z_c^r(x),$$

$$z_c^l(x) = \int_a^x h_c^l(t)\, dt, \quad z_c^r(x) = \int_x^b h_c^r(t)\, dt,$$

$$h_c^l(t) = (f(t) - \mu_l(x))^2, \quad h_c^r(t) = (f(t) - \mu_r(x))^2,$$

$$z_c^{l(n)}(x_k) = \sum_{i=1}^{k} h_c^l(x_i) * \Delta_i\ , \quad z_c^{r(n)}(x_{k+1}) = \sum_{j=k+1}^{n} h_c^r(x_j) * \Delta_j\ ,$$

where $c = 1, ..., C$ and $\Delta_i = x_i - x_{i-1}, i = 1, ..., n$.

Under this construction, the optimal split x_s defined in (11) follows (9) in Definition 3.1. Obviously, $h_c^l(\cdot)$ and $h_c^r(\cdot)$ are integrable in $[a, b]$, because $f(\cdot)$ is integrable. So, by applying Lemma 3.2, x_s^n converges to x_s in probability and the rate of convergence is $O(n^{-1})$.

Since $f(\cdot)$ is integrable in $[a, b]$ and $x_{k_s^n+1} \xrightarrow{p} x_{k_s^n} = x_s^n \xrightarrow{p} x_s$, we can prove that

$$\lim_{n\to\infty} \frac{1}{x_{k_s^n} - a} \sum_{i=1}^{k_s^n} f(x_i)\Delta_i \xrightarrow{p} \frac{1}{x_s - a} \int_a^{x_s} f(u)\, du = \mu_l(x_s),$$

$$\lim_{n\to\infty} \frac{1}{b - x_{k_s^n+1}} \sum_{j=k_s^n+1}^{n} f(x_j)\Delta_j \xrightarrow{p} \frac{1}{b - x_s} \int_{x_s}^{b} f(u)\, du = \mu_r(x_s),$$

and the rate of convergence is $O(n^{-1})$. □

Proof of Theorem 3.4. Construct that

$$g(z_1^l(x),...,z_C^l(x)) = \frac{1}{1-q}(\sum_{c=1}^{C} z_c^l(x)^q - 1),$$

$$g(z_1^r(x),...,z_C^r(x)) = \frac{1}{1-q}(\sum_{c=1}^{C} z_c^r(x)^q - 1),$$

$$h_c^l(t) = \frac{1}{x-a} * I_{y_c}(f(t)), \quad h_c^r(t) = \frac{1}{b-x} * I_{y_c}(f(t)),$$

$$z_c^l(x) = \int_a^x h_c^l(t)\,dt = \int_a^x \frac{1}{x-a} * I_{y_c}(f(t))\,dt = \int_{S_i \cap [a,x]} \frac{1}{x-a}\,dt = p_{[a,x]}(y_c),$$

$$z_c^r(x) = \int_x^b h_c^r(t)\,dt = \int_x^b \frac{1}{b-x} * I_{y_c}(f(t))\,dt = \int_{S_i \cap [x,b]} \frac{1}{b-x}\,dt = p_{[x,b]}(y_c),$$

$$z_c^{l(n)}(x_k) = \sum_{i=1}^{k} h_c^l(x_i) * \Delta_i = \frac{1}{x-a}\sum_{i=1}^{k} \Delta_i * I_{y_c}(f(x_i)),$$

$$z_c^{r(n)}(x_{k+1}) = \sum_{j=k+1}^{n} h_c^r(x_j) * \Delta_j = \frac{1}{b-x}\sum_{j=k+1}^{n} \Delta_j * I_{y_c}(f(x_j)),$$

where $c = 1,...,C$, $\Delta_i = x_i - x_{i-1}, i = 1,...,n$ and $I_{y_c}(f(x))$ is an indicator function that is equal to 1 at $f(x) = y_c$ and 0 elsewhere.

Under this construction, the optimal split x_s defined in (12) follows (9) in Definition 3.1. Obviously, $h_c^l(\cdot)$ and $h_c^r(\cdot)$ are integrable in $[a,b]$. So, by applying Lemma 3.2, x_s^n converges to x_s in probability and the rate of convergence is $O(n^{-1})$. □

Proof of Theorem 3.5. Let's construct that

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n_k}(y_{ki} - \mu_k)^2 = z^l(k), \quad \lim_{n\to\infty} \sum_{l\neq k}(\sum_{j=1}^{n_l}(y_{lj} - \mu_{\bar{k}})^2) = z^r(k) \quad and \quad g(z) = z.$$

Obviously, (14) follows (9), so x_s follows Definition 3.1.

By the splitting criteria (2), the optimal split of the stump can be found that

$$x_s^n = \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} \frac{1}{n}[\sum_{i=1}^{n_k}(y_{ki} - \bar{y}_k)^2 + \sum_{l\neq k}(\sum_{j=1}^{n_l}(y_{lj} - \bar{y}_{\bar{k}})^2)],$$

where

$$\bar{y}_k = \frac{1}{n_k}\sum_{i=1}^{n_k} y_{ki} \quad and \quad \bar{y}_{\bar{k}} = \frac{1}{\sum_{l\neq k} n_l}\sum_{l\neq k}\sum_{j=1}^{n_l} y_{lj}.$$

Since we know that $n_k = \frac{1}{C_x}n, \quad k = 1, ..., C_x$. By the weak law of large numbers, we can prove that

$$\bar{y}_k \xrightarrow{p} \mu_k, \quad \bar{y}_{\bar{k}} \xrightarrow{p} \mu_{\bar{k}} \quad as \quad n \to \infty.$$

So, with probability one,

$$\begin{aligned}\lim_{n\to\infty} x_s^n &= \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} \lim_{n\to\infty} \frac{1}{n}[\sum_{i=1}^{n_k}(y_{ki} - \bar{y}_k)^2 + \sum_{l\neq k}(\sum_{j=1}^{n_l}(y_{lj} - \bar{y}_{\bar{k}})^2)] \\ &= \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} \lim_{n\to\infty} \frac{1}{n}[\sum_{i=1}^{n_k}(y_{ki} - \mu_k)^2 + \sum_{l\neq k}(\sum_{j=1}^{n_l}(y_{lj} - \mu_{\bar{k}})^2)] = x_s.\end{aligned}$$

x_s^n converges to x_s in probability and the rate of convergence is $O(n^{-1})$. □

Proof of Theorem 3.6. Let us construct that

$$p_{kj} = z^l(k), \quad \sum_{i\neq k} p_{ij} = z^r(k) \quad and \quad g(z(k)) = S_q(k).$$

Obviously, (15) follows (9), so x_s follows Definition 3.1.

By the splitting criteria (3), the optimal split of the stump can be found that

$$x_s^n = \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} [S_q^n(k) + S_q^n(\bar{k})],$$

where

$$S_q^n(k) = \frac{1}{1-q}(\sum_{j=1}^{C_y}(\frac{1}{n_k}\sum_{l=1}^{n_k} I(y_l = j))^q - 1),$$

$$S_q^n(\bar{k}) = \frac{1}{1-q}(\sum_{j=1}^{C_y}(\sum_{i\neq k}(\frac{1}{n_i}\sum_{m=1}^{n_i} I(y_m = j)))^q - 1), \quad i \in \{1, ..., C_x\}, \quad q \in \mathbb{R}.$$

Since we know that $n_k = \frac{1}{C_x}n, \quad k = 1, ..., C_x$. By Borel's law of large numbers, with probability one,

$$\lim_{n\to\infty} \frac{1}{n_k}\sum_{m=1}^{n_k} I(y_m = j) = p_{kj}, \quad k = 1, ..., C_x, \quad j = 1, ..., C_y.$$

So, with probability one,

$$\begin{aligned}\lim_{n\to\infty} x_s^n &= \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} \lim_{n\to\infty} [S_q^n(k) + S_q^n(\bar{k})] \\ &= \underset{k\in\{1,...,C_x\}}{\operatorname{argmin}} [S_q(k) + S_q(\bar{k})] = x_s.\end{aligned}$$

x_s^n converges to x_s in probability and the rate of convergence is $O(n^{-1})$. □

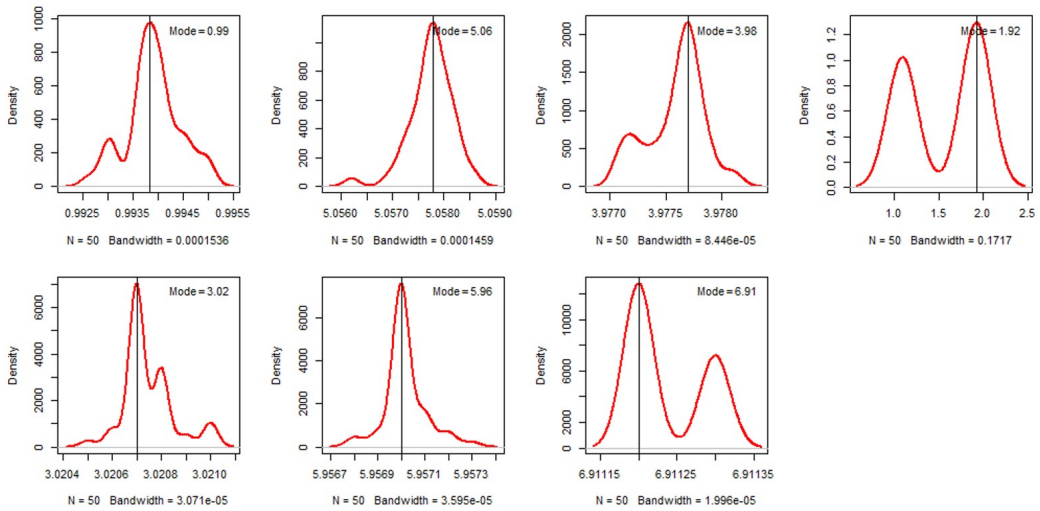

Figure B.17: One-dimensional data simulation: Density plot showing second-level stability of the splits are near true values

B Additional Simulation Results

Table B.2: Two-dimensional data simulation: Comparison of top level splits

Split sets \ **model**		true f	ODT	DDT
set 1	*split node*	*1*	*1*	*1*
	value	11.51	10.43	10.78
set 2	*split node*	*2*	*2*	*4*
	value	7.48	7.16	7.63
set 3	*split node*	*3*	*3*	*6,7*
	value	4.99	4.99	4.99
set 4	*split node*	*4*	*4*	*5*
	value	15.86	15.14	15.89
set 5	*split node*	*5*	*4*	*8*
	value	15.86	25.65	12.91
set 6	*split node*	*6*	\	*3*
	value	7.48	\	7.63
set 7	*split node*	*7*	*7*	*3*
	value	7.48	6.43	7.63
set 8	*split node*	\	*6*	\
	value	\	8.62	\
set 9	*split node*	\	\	*2*
	value	\	\	7.16

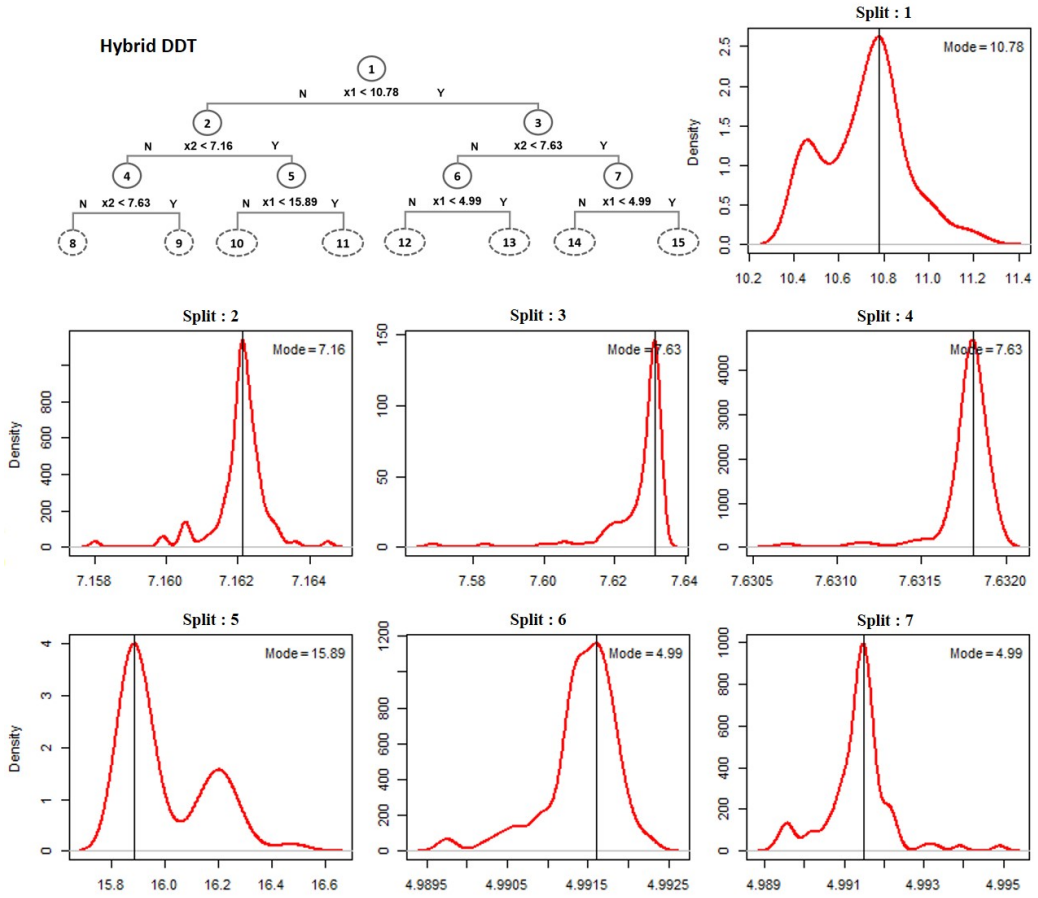

Figure B.18: Two-dimensional data simulation: Second-level stability of the top seven splits

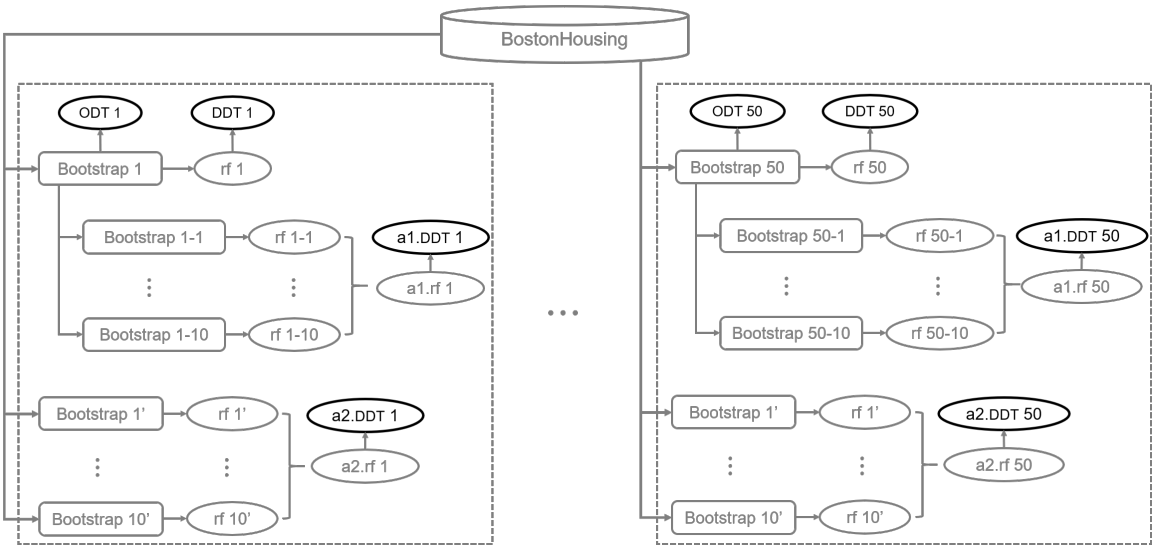

Figure B.19: BostonHousing dataset: Experiment design based on the BostonHousing data

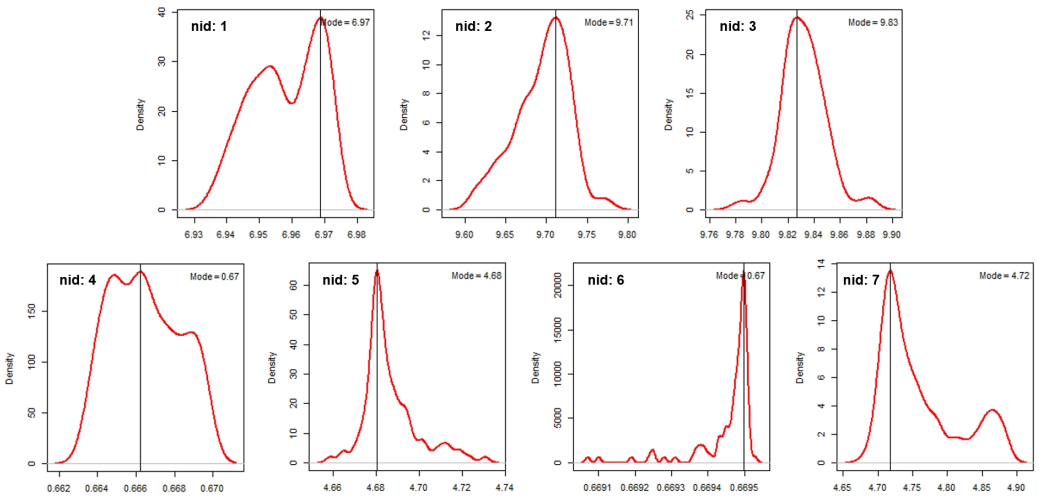

Figure B.20: BostonHousing dataset: Second-level stability of the interpretable nodes

C Acknowledgments

The authors appreciate the helpful discussions with Dr. Wei-Ying Lou and the editorial assistance from Mrs. Jessica Swann.

D funding

J. Jack Lee's research was supported in part by the grants P30CA016672, P50CA221703, U24CA224285, and U24CA224020 from the National Cancer Institute, RP150519 and RP160668 from the Cancer Prevention and Research Institute of Texas, and The University of Texas MD Anderson Cancer Center Oropharynx Cancer Program generously supported by Mr. and Mrs. Charles W. Stiefel.

References

Allen-Zhu, Z. and Y. Li (2021). Towards understanding ensemble, knowledge distillation and self-distillation in deep learning.

Ba, J. and R. Caruana (2014). Do deep nets really need to be deep? In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems*, Volume 27. Curran Associates, Inc.

Breiman, L. (2001). Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science 16*(3), 199 – 231.

Breiman, L., J. Friedman, C. J. Stone, and R. Olshen (1984). *Classification and Regression Trees*. Chapman and Hall/CRC.

Buciluǎ, C., R. Caruana, and A. Niculescu-Mizil (2006). Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, New York, NY, USA, pp. 535–541. Association for Computing Machinery.

Bénard, C., G. Biau, S. da Veiga, and E. Scornet (2020). Interpretable random forests via rule extraction.

Coppens, Y., K. Efthymiadis, T. Lenaerts, and A. Nowé (2019). Distilling deep reinforcement learning policies in soft decision trees. In *IJCAI 2019*.

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein (2009). *Introduction to Algorithms, Third Edition* (3rd ed.). The MIT Press.

Ding, Z., P. Hernandez-Leal, G. W. Ding, C. Li, and R. Huang (2021). Cdt: Cascading decision trees for explainable reinforcement learning.

Du, S., S. You, X. Li, J. Wu, F. Wang, C. Qian, and C. Zhang (2020). Agree to disagree: Adaptive ensemble knowledge distillation in gradient space. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), *Advances in Neural Information Processing Systems*, Volume 33, pp. 12345–12355. Curran Associates, Inc.

Frosst, N. and G. Hinton (2017). Distilling a neural network into a soft decision tree.

Gou, J., B. Yu, S. J. Maybank, and D. Tao (2021, Mar). Knowledge distillation: A survey. *International Journal of Computer Vision 129*(6), 1789–1819.

Hinton, G., O. Vinyals, and J. Dean (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Johansson, U., C. Sönströd, and T. Löfström (2011). One tree to explain them all. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pp. 1444–1451.

John, G. H. (1995). Robust decision trees: Removing outliers from databases. In *In Knowledge Discovery and Data Mining*, pp. 174–179. AAAI Press.

Kang, J. and J. Gwak (2020). Ensemble learning of lightweight deep learning models using knowledge distillation for image classification. *Mathematics 8*(10).

Klusowski, J. M. (2021). Universal consistency of decision trees in high dimensions.

Leisch, F. and E. Dimitriadou (2021). *mlbench: Machine Learning Benchmark Problems*. R package version 2.1-3.

Li, J., Y. Li, X. Xiang, S.-T. Xia, S. Dong, and Y. Cai (2020). Tnt: An interpretable tree-network-tree learning framework using knowledge distillation. *Entropy 22*(11).

Menon, A. K., A. S. Rawat, S. Reddi, S. Kim, and S. Kumar (2021, 18–24 Jul). A statistical perspective on distillation. In M. Meila and T. Zhang (Eds.), *Proceedings of the 38th International Conference on Machine Learning*, Volume 139 of *Proceedings of Machine Learning Research*, pp. 7632–7642. PMLR.

Morgan, J. N. and J. A. Sonquist (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American Statistical Association 58*(302), 415–434.

Quinlan, J. R. (1986). Induction of decision trees. *MACH. LEARN 1*, 81–106.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Rahman, G. and Z. Islam (2011). A decision tree-based missing value imputation technique for data pre-processing. In *Proceedings of the Ninth Australasian Data Mining Conference - Volume 121*, AusDM '11, AUS, pp. 41–50. Australian Computer Society, Inc.

Rokach, L. and O. Maimon (2014). *Data Mining With Decision Trees: Theory and Applications* (2nd ed.). USA: World Scientific Publishing Co., Inc.

Shen, Y., X. Xu, and J. Cao (2020). Reconciling predictive and interpretable performance in repeat buyer prediction via model distillation and heterogeneous classifiers fusion. *Neural Comput. Appl.*.

Shi, Y., M.-Y. Hwang, X. Lei, and H. Sheng (2019). Knowledge distillation for recurrent neural network language modeling with trust regularization.

Song, J., H. Zhang, X. Wang, M. Xue, Y. Chen, L. Sun, D. Tao, and M. Song (2021, June). Tree-like decision distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13488–13497.

Stanton, S., P. Izmailov, P. Kirichenko, A. A. Alemi, and A. G. Wilson (2021). Does knowledge distillation really work?

Urban, G., K. J. Geras, S. E. Kahou, O. Aslan, S. Wang, R. Caruana, A. Mohamed, M. Philipose, and M. Richardson (2017). Do deep convolutional nets really need to be deep and convolutional?

Wang, L. and K.-J. Yoon (2021). Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1.

Wang, Y. and S.-T. Xia (2017). Unifying attribute splitting criteria of decision trees by tsallis entropy. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2507–2511.

Xie, Y., G. Gao, and X. Chen (2019). Outlining the design space of explainable intelligent systems for medical diagnosis. *arXiv preprint arXiv:1902.06019*.

Yu, B. (2013). Stability. *Bernoulli 19*(4), 1484–1500.

Zhou, Y., Z. Zhou, and G. Hooker (2018). Approximation trees: Statistical stability in model distillation.