entropy

MDPI

Article

Meta-Tree Random Forest: Probabilistic Data-Generative Model and Bayes Optimal Prediction

Nao Dobashi [1,†], Shota Saito [2,*], Yuta Nakahara [3] and Toshiyasu Matsushima [1]

[1] Department of Pure and Applied Mathematics, Waseda University, 3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555, Japan; nao-0846htt@toki.waseda.jp (N.D.); toshimat@waseda.jp (T.M.)
[2] Faculty of Informatics, Gunma University, 4-2, Maebashi, Gunma 371-8510, Japan
[3] Center for Data Science, Waseda University, 1-6-1 Nisniwaseda, Shinjuku-ku, Tokyo 169-8050, Japan; yuta.nakahara@aoni.waseda.jp
* Correspondence: shota.s@gunma-u.ac.jp
† Current address: Digital Printing Business Operations, Canon Inc., Tokyo 146-8501, Japan.

Abstract: This paper deals with a prediction problem of a new targeting variable corresponding to a new explanatory variable given a training dataset. To predict the targeting variable, we consider a model tree, which is used to represent a conditional probabilistic structure of a targeting variable given an explanatory variable, and discuss statistical optimality for prediction based on the Bayes decision theory. The optimal prediction based on the Bayes decision theory is given by weighting all the model trees in the model tree candidate set, where the model tree candidate set is a set of model trees in which the true model tree is assumed to be included. Because the number of all the model trees in the model tree candidate set increases exponentially according to the maximum depth of model trees, the computational complexity of weighting them increases exponentially according to the maximum depth of model trees. To solve this issue, we introduce a notion of meta-tree and propose an algorithm called MTRF (Meta-Tree Random Forest) by using multiple meta-trees. Theoretical and experimental analyses of the MTRF show the superiority of the MTRF to previous decision tree-based algorithms.

Keywords: bayes decision theory; data-generative model; meta-tree; prediction; random forest

Citation: Dobashi, N.; Saito, S.; Nakahara, Y.; Matsushima, T. Meta-Tree Random Forest: Probabilistic Data-Generative Model and Bayes Optimal Prediction. *Entropy* **2021**, *23*, 768. https://doi.org/10.3390/e23060768

Academic Editor: Udo Von Toussaint

Received: 19 May 2021
Accepted: 8 June 2021
Published: 18 June 2021

1. Introduction

Various studies in pattern recognition deal with a prediction problem of a targeting variable y_{n+1} corresponding to an explanatory variable $\boldsymbol{x}_{n+1}$ given pairs of explanatory and targeting variable $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$. In many of them, the targeting variable y_{n+1} is predicted with a tree T. One way to use a tree T is to represent a *function* $y_{n+1} = f(\boldsymbol{x}_{n+1}; T)$ and predict y_{n+1}. A tree T used to represent the function $y_{n+1} = f(\boldsymbol{x}_{n+1}; T)$ is called a decision tree in the literature. In this paper, however, this tree is called a function tree. This is because we distinguish a tree used to represent a function from a tree used to represent a data-generative model (A tree used to represent a data-generative model will be explained in the next paragraph.). In the previous studies, algorithms in which a single function tree is used are discussed in, e.g., CART [1]; algorithms in which multiple function trees are used are discussed, e.g., Random Forest [2] is an algorithm that constructs multiple function trees from $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and aggregates them to predict y_{n+1} from $\boldsymbol{x}_{n+1}$. There are various extensions of Random Forest, e.g., Generalized Random Forest [3] generalizes the splitting rule of a function tree. Boosting is an algorithm that constructs a function tree sequentially from $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and combines the constructed function trees to predict y_{n+1} from $\boldsymbol{x}_{n+1}$. There are various Boosting methods, e.g., gradient boosting method in Gradient Boost [4] and XGBoost [5]. Further, previous studies, such as Alternating Decision Forest [6] and Boosted Random Forest [7], combine the ideas of Random Forest and Boosting method. Moreover, combinations of the function trees and neural networks

have also been discussed. In Neural Decision Forest [8], inner nodes are replaced by randomized multi-layer perceptrons. In Deep Neural Decision Forest [9], they are replaced by deep convolutional neural networks (CNN). In Adaptive Neural Trees [10], not only their nodes are replaced by CNNs but also their edges are replaced by nonlinear functions. In Deep Forest [11], Random Forests are combined in a cascade structure like deep CNNs. The function tree is also used to understand the deep neural network as an interpretable input-output function in Reference [12]. In these algorithms, a function tree is used to represent the function from $\boldsymbol{x}_{n+1}$ to y_{n+1}. Although this function is trained for the data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ under a certain criterion, statistical optimality for prediction of y_{n+1} is not necessarily discussed theoretically because these algorithms usually do not assume a probabilistic data-generative structure of $\boldsymbol{x}$ and y. As we will describe later in detail, on the other hand, we assume a probabilistic data-generative structure of $\boldsymbol{x}$ and y, and consider a statistical optimal prediction of y_{n+1}. This is the crucial difference between our study and the related works.

To predict the targeting variable y_{n+1}, another way to use a tree T is to represent a data-generative model $p(y|\boldsymbol{x}, T)$ that represents a conditional probabilistic structure of y given $\boldsymbol{x}$. A tree T used to represent the data-generative model $p(y|\boldsymbol{x}, T)$ is called a model tree throughout this paper. Although not so many studies have assumed the model tree, Reference [13] has assumed it. Because Reference [13] assumed that a targeting variable y_{n+1} is generated according to $p(y|\boldsymbol{x}_{n+1}, T)$, statistical optimality for prediction of y_{n+1} can be discussed theoretically. Specifically, based on the Bayes decision theory [14], Reference [13] proposed the optimal prediction of y_{n+1}.

In order to explain the optimal prediction based on the Bayes decision theory in more detail, we introduce the terminology *model tree candidate set*. The model tree candidate set is a set of model trees in which the true model tree (When the model tree T is used to represent the true data-generative model $p(y|\boldsymbol{x}, T)$, we call it true model tree.) is assumed to be included. One example of a model tree candidate set is a set of model trees whose depth is up to $d \in \mathbb{N}$. The optimal prediction based on the Bayes decision theory is given by weighting all the model trees in the model tree candidate set and the optimal weight of each model tree is given by the posterior probability $p(T|\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n)$ of the model tree T.

Because the number of all the model trees in the model tree candidate set increases *exponentially* according to the maximum depth of model trees, the computational complexity of weighting them increases *exponentially* according to the maximum depth of model trees. One way to reduce the computational complexity is to restrict the model tree candidate set. To represent a restricted model tree candidate set, Reference [13] proposed a notion of *meta-tree*. The concept of a meta-tree was originally used for data compression in information theory (see, e.g., Reference [15]) (Recently, the concept of a meta-tree was also used for image compression [16].). As shown in Figure 1, a model tree candidate set is composed of the model trees represented by the meta-tree.

A meta-tree is also used for the prediction of y_{n+1} in the algorithm proposed by Reference [13]. In summary, a meta-tree has the following two roles: (i) it represents a model tree candidate set; and (ii) it is used for the prediction algorithm. The characteristics of a meta-tree are as follows:

- If the true model tree is in a model tree candidate set represented by a meta-tree, the statistically optimal prediction—optimal prediction based on the Bayes decision theory—can be calculated.
- The number of model trees in a model tree candidate set represented by a meta-tree increases *exponentially* according to the depth of the meta-tree.
- The computational cost in learning processes of a single meta-tree has the same order as that of a single function tree.

Under the assumption that the true model tree is in the restricted model tree candidate set represented by a meta-tree, Reference [13] proposed the optimal prediction based on the Bayes decision theory.

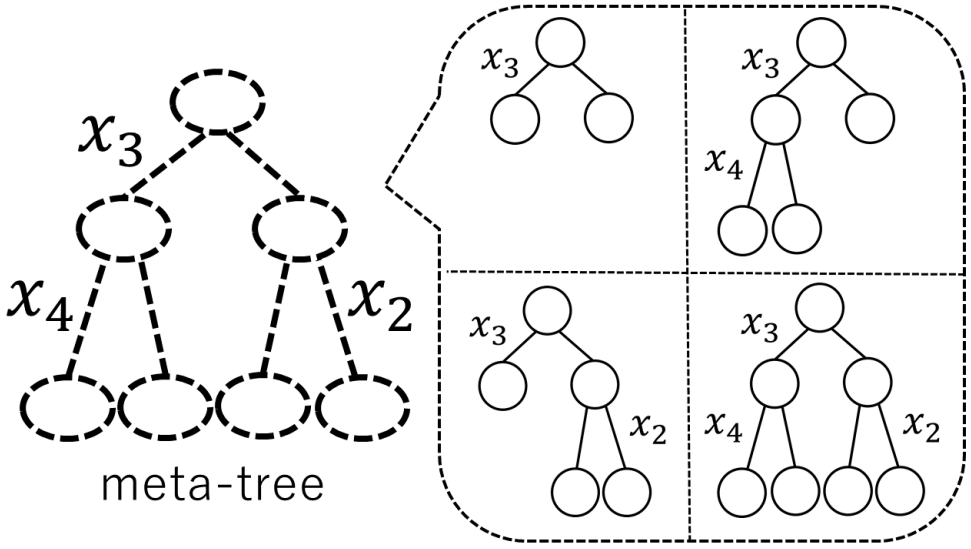

Figure 1. An example of a meta-tree and four model trees represented by it.

As we have described above, if the true model tree is actually included in the model tree candidate set, the optimal prediction based on the Bayes decision theory is calculated. Hence, it is desirable that we construct the model tree candidate set that includes as many model trees as possible within the allowed computational cost. Motivated by this fact, this paper extends the model tree candidate set compared with Reference [13]. Instead of considering a *single* meta-tree as in Reference [13], we consider *multiple* meta-trees. By using the model tree candidate set represented by multiple meta-trees, we predict y_{n+1} based on the Bayes decision theory. We call this proposed algorithm *MTRF* (*Meta-Tree Random Forest*). The characteristics of MTRF are as follows:

- If the true model tree is in a model tree candidate set represented by any of the meta-trees of MTRF, the statistically optimal prediction—optimal prediction based on the Bayes decision theory—can be calculated.
- The number of model trees in the model tree candidate set represented by multiple meta-trees is constant times larger than those contained in a single meta-tree.
- The computational cost in learning processes of meta-trees of MTRF is the same order as that of multiple function trees of Random Forest.

Regarding a meta-tree in Reference [13] or multiple meta-trees of MTRF, how to construct the meta-tree/multiple meta-trees is a problem. One way to solve this issue is to use the function tree-based algorithms; for example, a meta-tree in Reference [13] can be constructed by regarding a function tree in CART as a meta-tree; multiple meta-trees of MTRF can be constructed by regarding function trees in Random Forest as meta-trees. In this way, by regarding a function tree as a meta-tree, our proposed method can be applied to any practical applications in which a function-tree based algorithm is used—insurance claim task (e.g., Reference [5]), letter classification task (e.g., Reference [6]), semantic segmentation (e.g., Reference [8]), face recognition (e.g., Reference [11]), music classification (e.g., Reference [11]), etc.—and we can improve every previous function tree-based algorithm. This is because we can perform a prediction of y_{n+1} by using more model trees compared with the previous algorithms. More detailed discussion will be given in Sections 3 and 4.

The rest of this paper is organized as follows. In Section 2, we state the preliminaries of our study. Section 3 explains our proposed method—MTRF. In Section 4, we revisit the previous studies, such as CART [1] and Random Forest [2], and compare them with MTRF. Section 5 shows the results of experiments. Section 6 concludes this paper.

2. Preliminaries

2.1. Model Tree

Let K denote the dimension of explanatory variables. The space of feature values is denoted by $\mathcal{X} := \{0, 1, \ldots, |\mathcal{X}| - 1\}$, and the explanatory variable is denoted by $\boldsymbol{x} := (x_1, \ldots, x_K)^\top \in \mathcal{X}^K$. In addition, the space of label values is denoted by $\mathcal{Y} := \{0, 1 \ldots, |\mathcal{Y}| - 1\}$, and the targeting variable is denoted by $y \in \mathcal{Y}$.

In the field of pattern recognition, the probabilistic structure of $\boldsymbol{x}$ and y is not necessarily assumed. In particular, it is not assumed in most of function tree-based algorithms. In contrast, we consider a triplet of $(T, \boldsymbol{k}, \boldsymbol{\theta})$ defined in Definition 1 below and assume the probabilistic structure $p(y|\boldsymbol{x}, \boldsymbol{\theta}, T, \boldsymbol{k})$ as in Definition 2. We call $(T, \boldsymbol{k})$ a *model tree* (In Section 1, we called T a model tree. More precisely speaking, however, a model tree is $(T, \boldsymbol{k})$.) and $\boldsymbol{\theta}$ a *tree parameter*.

Definition 1. *The notation T denotes the $|\mathcal{X}|$-ary regular (Here, "regular" means that all inner nodes have $|\mathcal{X}|$ child nodes.) tree whose depth is up to $D_{\max}$, and $\mathcal{T}$ denotes the set of T. Let s be a node of a tree and $\mathcal{S}$ be a set of s. The set of the inner nodes of T is denoted by $\mathcal{I}(T)$, and the set of the leaf nodes of T is denoted by $\mathcal{L}(T)$. Let $k_s \in \{1, 2, \ldots, K\}$ be a component of the feature assign vector of node s. The explanatory variable $x_{k_s} \in \mathcal{X}$ is assigned to the inner node $s \in \mathcal{I}(T)$. Let $\boldsymbol{k} := (k_s)_{s \in \mathcal{S}}$ denote a feature assign vector, and $\mathcal{K}$ denote a set of $\boldsymbol{k}$. Let $\boldsymbol{\theta}_s := (\theta_{0|s}, \theta_{1|s}, \ldots, \theta_{|\mathcal{Y}|-1|s}) \in (0,1)^{|\mathcal{Y}|}$, where $\sum_{y=0}^{|\mathcal{Y}|-1} \theta_{y|s} = 1$, be a parameter assigned to $s \in \mathcal{S}$. We define $\boldsymbol{\theta} := (\boldsymbol{\theta}_s)_{s \in \mathcal{S}}$, and $\boldsymbol{\Theta}$ is the set of $\boldsymbol{\theta}$.*

Definition 2. *Because the value of explanatory variable $\boldsymbol{x}$ corresponds to a path from the root node to the leaf node of T whose feature assign vector is $\boldsymbol{k}$, let $s(\boldsymbol{x}) \in \mathcal{L}(T)$ denote the corresponding leaf node. Then, for given $\boldsymbol{x}$, $\boldsymbol{\theta}$, T, and $\boldsymbol{k}$, the label variable y is generated according to*

$$p(y|\boldsymbol{x}, \boldsymbol{\theta}, T, \boldsymbol{k}) := \theta_{y|s(\boldsymbol{x})}. \tag{1}$$

Figure 2a shows two examples of $(T, \boldsymbol{k}, \boldsymbol{\theta})$, and Figure 2b illustrates $p(y|\boldsymbol{x}, \boldsymbol{\theta}, T, \boldsymbol{k})$.

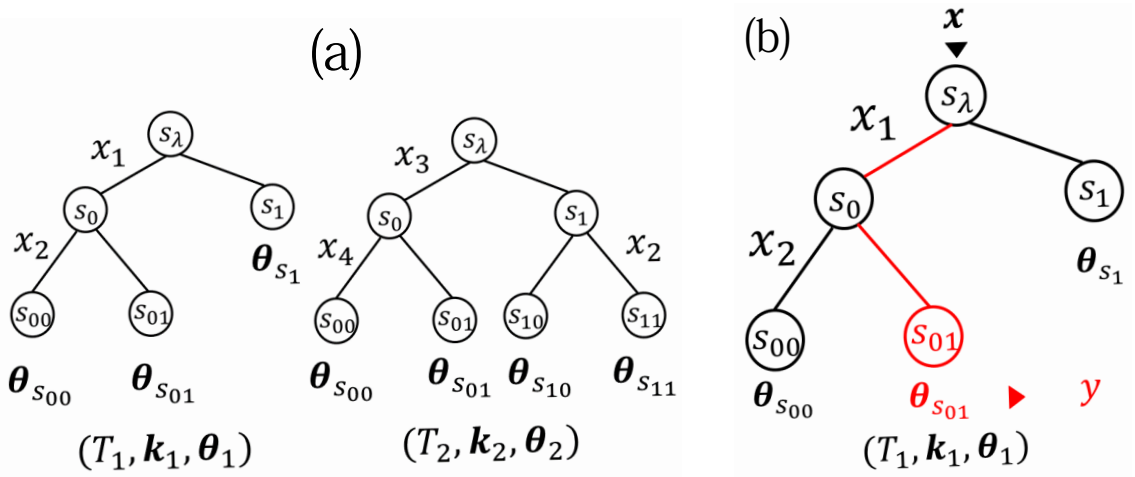

Figure 2. (**a**) Two examples of $(T_1, \boldsymbol{k}_1, \boldsymbol{\theta}_1)$ and $(T_2, \boldsymbol{k}_2, \boldsymbol{\theta}_2)$, where T_1 and T_2 are 2-ary regular trees, $\boldsymbol{k}_1 = (k_{s_\lambda}, k_{s_0}) = (1, 2)$, and $\boldsymbol{k}_2 = (k_{s_\lambda}, k_{s_0}, k_{s_1}) = (3, 4, 2)$. (**b**) $p(y|\boldsymbol{x} = (0,1)^\top, \boldsymbol{\theta}_1, T_1, \boldsymbol{k}_1) = \theta_{y|s_{01}}$.

2.2. Problem Setup

In this subsection, we introduce our problem setup. Let $\boldsymbol{x}_i := (x_{i1}, \ldots, x_{iK})^\top \in \mathcal{X}^K (i = 1, \ldots, n)$ denote the value of explanatory variable of the i-th data and $\boldsymbol{x}^n := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$. We assume that $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are i.i.d. random variables drawn from a probability distribution. Let $y_i \in \mathcal{Y} (i = 1, \ldots, n)$ denote the targeting variable of the i-th data and $y^n := (y_1, \ldots, y_n)$. We assume that $y_1, \ldots, y_n$ are generated according to $p(y|\boldsymbol{x}, \boldsymbol{\theta}, T, \boldsymbol{k})$ defined in Section 2.1. In regard to this, we assume that the true model tree $(T^\star, \boldsymbol{k}^\star)$ and true tree parameter $\boldsymbol{\theta}^\star$

are unknown, but a model tree candidate set $\mathcal{M} \subset \mathcal{T} \times \mathcal{K}$ is known. Note that, as we have explained in Section 1, the model tree candidate set $\mathcal{M}$ is a set of model trees $(T, \boldsymbol{k})$ in which the true model tree is assumed to be included. Further, we assume that a class of parametrized distribution $\{p(y|\boldsymbol{x}, \boldsymbol{\theta}, T, \boldsymbol{k}) : \boldsymbol{\theta} \in \boldsymbol{\Theta}, (T, \boldsymbol{k}) \in \mathcal{M}\}$ is known.

The purpose of the prediction problem is to predict unknown targeting variable y_{n+1} corresponding to an explanatory variable $\boldsymbol{x}_{n+1}$ by using the given data $\boldsymbol{x}^n$, y^n, $\boldsymbol{x}_{n+1}$.

2.3. Optimal Prediction Based on the Bayes Decision Theory

We introduce a statistically optimal prediction under the problem setup in Section 2.2. From the viewpoint of the Bayes decision theory [14], we shall derive the optimal prediction based on the Bayes decision theory. To this end, we assume priors $p(\boldsymbol{\theta})$, $p(T)$, and $p(\boldsymbol{k})$.

First, we define a decision function as

$$\delta : \mathcal{X}^{Kn} \times \mathcal{Y}^n \times \mathcal{X}^K \to \mathcal{Y} \, ; (\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) \mapsto \delta(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}). \tag{2}$$

Second, we define a 0–1 loss as follows:

$$l(y_{n+1}, \delta(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})) := \begin{cases} 0 & (y_{n+1} = \delta(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})), \\ 1 & (y_{n+1} \neq \delta(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})). \end{cases} \tag{3}$$

Third, using the 0–1 loss, we define the loss function, which is the expectation of the 0–1 loss taken by new targeting variable y_{n+1}:

$$L(\delta(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}), \boldsymbol{\theta}, T, \boldsymbol{k}) := \sum_{y_{n+1} \in \mathcal{Y}} p(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{\theta}, T, \boldsymbol{k}) l(y_{n+1}, \delta(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})). \tag{4}$$

Next, given the loss function, we define the risk function, which is the expectation of the loss function taken by training data $\boldsymbol{x}^n$, y^n, and $\boldsymbol{x}_{n+1}$:

$$\begin{aligned} &R(\delta(\cdot, \cdot, \cdot), \boldsymbol{\theta}, T, \boldsymbol{k}) \\ &:= \sum_{\boldsymbol{x}^n \in \mathcal{X}^{Kn}} \sum_{y^n \in \mathcal{Y}^n} \sum_{\boldsymbol{x}_{n+1} \in \mathcal{X}^K} p(\boldsymbol{x}_{n+1}) p(\boldsymbol{x}^n) p(y^n|\boldsymbol{x}^n, \boldsymbol{\theta}, T, \boldsymbol{k}) L(\delta(\boldsymbol{x}^n, y^n \boldsymbol{x}_{n+1}), \boldsymbol{\theta}, T, \boldsymbol{k}). \end{aligned} \tag{5}$$

Finally, the Bayes risk function, which is the expectation of the risk function taken by $\boldsymbol{\theta}$, T, and $\boldsymbol{k}$, is defined as

$$\mathrm{BR}(\delta(\cdot, \cdot, \cdot)) := \sum_{(T, \boldsymbol{k}) \in \mathcal{M}} p(\boldsymbol{k}) p(T) \int_{\boldsymbol{\Theta}} p(\boldsymbol{\theta}) R(\delta(\cdot, \cdot, \cdot), \boldsymbol{\theta}, T, \boldsymbol{k}) d\boldsymbol{\theta}. \tag{6}$$

Then, we have the following theorem:

Theorem 1. *Under the setup in Section 2.2, the decision function $\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$ that minimizes the Bayes risk function is given as*

$$\begin{aligned} &\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) = \\ &\quad \arg\max_{y_{n+1}} \sum_{(T, \boldsymbol{k}) \in \mathcal{M}} p(\boldsymbol{k}|\boldsymbol{x}^n, y^n) p(T|\boldsymbol{x}^n, y^n, \boldsymbol{k}) \int_{\boldsymbol{\Theta}} p(\boldsymbol{\theta}|\boldsymbol{x}^n, y^n, T, \boldsymbol{k}) p(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{\theta}, T, \boldsymbol{k}) \mathrm{d}\boldsymbol{\theta}. \end{aligned} \tag{7}$$

The proof of Theorem 1 is in Appendix A. We call $\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$ the *Bayes decision*.

2.4. Previous Study

In this subsection, we introduce how to calculate the Bayes decision in Reference [13]. Because the proof of Reference [13] lacks some explanation, we give more rigorous discussion in comparison. This is one of the contributions in this paper.

As we have explained in Section 1, Reference [13] introduced the concept of a "meta-tree" to represent a restricted model tree candidate set. For T and $\boldsymbol{k}$, a meta-tree—denoted

by $\mathrm{M}_{T,\boldsymbol{k}}$—represents a model tree candidate set that is composed of model trees $(T', \boldsymbol{k}')$ where T' is a sub-tree of T and $\boldsymbol{k}' = \boldsymbol{k}$. An example of $\mathrm{M}_{T,\boldsymbol{k}}$ is shown in Figure 3, where T is a complete tree with $D_{\max} = 2$ and $\boldsymbol{k} = (3, 4, 2)$. In this example, a meta-tree $\mathrm{M}_{T,\boldsymbol{k}}$ represents a model tree candidate set which is composed of four model trees. A meta-tree is also used for the prediction of y_{n+1} in the algorithm proposed by Reference [13]. In short, a meta-tree has the two roles: first, it represents a model tree candidate set, and, second, it is used for the prediction algorithm.

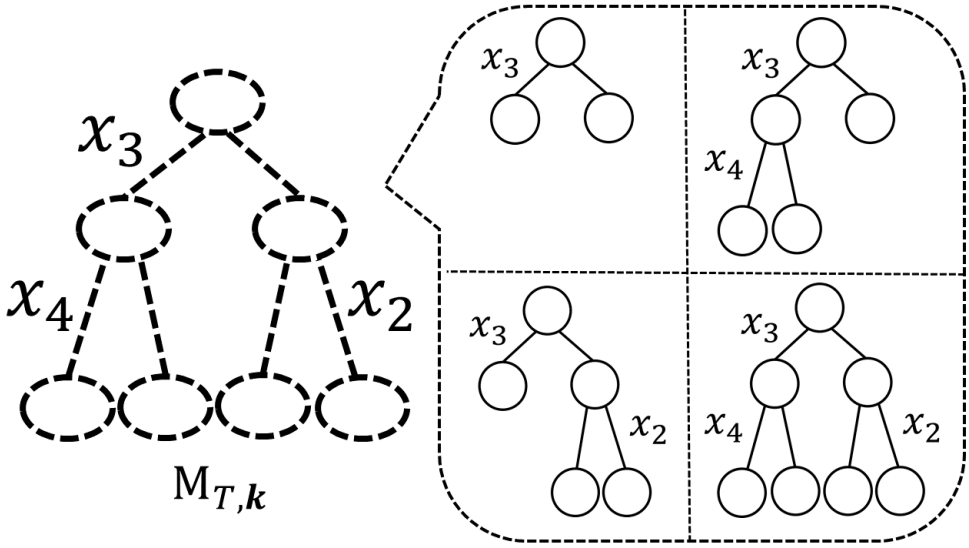

Figure 3. An example of a meta-tree $\mathrm{M}_{T,\boldsymbol{k}}$, where T is a complete tree with $D_{\max} = 2$ and $\boldsymbol{k} = (3, 4, 2)$.

Let $\mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}$ denote a set of T represented by a meta-tree $\mathrm{M}_{T,\boldsymbol{k}}$ and let $\mathcal{M} = \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}} \times \{\boldsymbol{k}\}$. Under the assumption that the true model tree is included in $\mathcal{M}$, the Bayes decision in Theorem 1 is expressed as

$$\begin{aligned}&\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) \\ &\quad = \arg\max_{y_{n+1}} \sum_{T' \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}} p(T'|\boldsymbol{x}^n, y^n, \boldsymbol{k}) \int_{\boldsymbol{\Theta}} p(\boldsymbol{\theta}|\boldsymbol{x}^n, y^n, T', \boldsymbol{k}) p(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{\theta}, T', \boldsymbol{k}) \mathrm{d}\boldsymbol{\theta}. \end{aligned} \tag{8}$$

Now, (8) can be decomposed into two parts:

$$1.\quad q(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, T, \boldsymbol{k}) \coloneqq \int_{\boldsymbol{\Theta}} p(\boldsymbol{\theta}|\boldsymbol{x}^n, y^n, T, \boldsymbol{k}) p(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{\theta}, T, \boldsymbol{k}) \mathrm{d}\boldsymbol{\theta}. \tag{9}$$

$$2.\quad \tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k}) \coloneqq \sum_{T' \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}} p(T'|\boldsymbol{x}^n, y^n, \boldsymbol{k}) q(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, T', \boldsymbol{k}). \tag{10}$$

Reference [13] exactly calculates (9) and (10) under some assumptions. We describe them in Sections 2.4.1 and 2.4.2, respectively.

2.4.1. Expectation over the Parameters

In this subsection, we explain how to calculate (9). To calculate (9) analytically, a conjugate prior of $\boldsymbol{\theta}_s$ given T and $\boldsymbol{k}$, is assumed.

Assumption 1. *We assume* $p(\boldsymbol{\theta}_s|T, \boldsymbol{k}) = \mathrm{Dirichlet}(\boldsymbol{\theta}_s|\boldsymbol{\alpha}_s)$ *for* $s \in \mathcal{L}(T)$, *where* $\boldsymbol{\alpha}_s$ *denotes the hyper-parameter of the Dirichlet distribution.*

Under this assumption, $q(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, T, \boldsymbol{k})$ has a closed form expression and (9) can be calculated exactly. In addition, $q(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, T, \boldsymbol{k})$ has an important property. We describe this in the next lemma as a preparation of Section 2.4.2.

Lemma 1. *For any* $T, T' \in \mathcal{T}$, $\boldsymbol{x}^n \in \mathcal{X}^{Kn}$, $y^n \in \mathcal{Y}^n$, *and* $\boldsymbol{x}_{n+1} \in \mathcal{X}^K$, *if* $s(\boldsymbol{x}_{n+1}) \in \mathcal{L}(T) \cap \mathcal{L}(T')$, *then*

$$q(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, T, \boldsymbol{k}) = q(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, T', \boldsymbol{k}). \tag{11}$$

The proof of Lemma 1 is in Appendix B. From this lemma, the right- and left-hand sides of (11) are denoted by $q_{s(\boldsymbol{x}_{n+1})}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k})$ because they depend on not T and T' but $s(\boldsymbol{x}_{n+1})$.

2.4.2. Summation over All Model Trees Represented by a Meta-Tree

In this subsection, we explain how to calculate (10). The exact calculation of (10) needs to take summation over all model trees represented by a single meta-tree. However, the computational complexity of this calculation is huge because the number of model trees increases exponentially according to the depth of meta-tree. Reference [13] solved this problem. The advantages of using the method proposed in Reference [13] are as follows:

- This method calculates (10) exactly.
- The computational complexity of calculating (10) in learning parameters is $O(nD_{\max})$, which is the same as that of building a single function tree.

To execute this method, an assumption about the prior probability distribution of T—Assumption 2—is required. It should be noted that $p(T)$ is different from the construction method of function trees, given $\boldsymbol{x}^n$ and y^n.

Assumption 2. *Let* $g_s \in [0, 1]$ *denote a hyper-parameter assigned to any node* $s \in \mathcal{S}$. *Then, we assume the prior of* $T' \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}$ *as*

$$p(T') = \prod_{s \in \mathcal{I}(T')} g_s \prod_{s' \in \mathcal{L}(T')} (1 - g_{s'}), \tag{12}$$

where $g_s = 0$ *for a leaf node* s *of a meta-tree* $\mathrm{M}_{T,\boldsymbol{k}}$.

Remark 1. *It should be noted that* (12) *is a probability distribution, i.e.,* $\sum_{T' \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}} p(T') = 1$. *See Lemma A1 in Appendix C.*

As we have described in Section 1, a meta-tree is used for the prediction of y_{n+1} in the algorithm proposed by Reference [13]. By using a meta-tree $\mathrm{M}_{T,\boldsymbol{k}}$, the recursive function to calculate the Bayes decision (10) is defined as follows.

Definition 3. *We define the following recursive function* $\tilde{q}_s(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \mathrm{M}_{T,\boldsymbol{k}})$ *for any node* s *on the path in* $\mathrm{M}_{T,\boldsymbol{k}}$ *corresponding to* $\boldsymbol{x}_{n+1}$.

$$\tilde{q}_s(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \mathrm{M}_{T,\boldsymbol{k}}) := \begin{cases} q_s(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k}), & (\text{if } s \text{ is the leaf node of } \mathrm{M}_{T,\boldsymbol{k}}), \\ (1 - g_{s|\boldsymbol{x}^n, y^n}) q_s(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k}) & \\ \quad + g_{s|\boldsymbol{x}^n, y^n} \tilde{q}_{s_{\mathrm{child}}}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \mathrm{M}_{T,\boldsymbol{k}}), & (\text{otherwise}), \end{cases} \tag{13}$$

where s_{child} *is the child node of* s *on the path corresponding to* $\boldsymbol{x}_{n+1}$ *in* $\mathrm{M}_{T,\boldsymbol{k}}$, *and* $g_{s|\boldsymbol{x}^n, y^n}$ *is also recursively updated as follows:*

$$g_{s|\boldsymbol{x}^i, y^i} := \begin{cases} g_s, & (\text{if } i = 0), \\ \frac{g_{s|\boldsymbol{x}^{i-1}, y^{i-1}} \tilde{q}_{s_{\mathrm{child}}}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}})}{\tilde{q}_s(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}})}, & (\text{otherwise}). \end{cases} \tag{14}$$

Now, (10) can be calculated as shown in the following theorem.

Theorem 2. *$\tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k})$ can be calculated by*

$$\tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k}) = \tilde{q}_{s_\lambda}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \mathrm{M}_{T,\boldsymbol{k}}), \tag{15}$$

where s_λ is the root node of $\mathrm{M}_{T,\boldsymbol{k}}$. In addition, $p(T|\boldsymbol{x}^{n+1}, y^{n+1}, \boldsymbol{k})$ can be calculated as

$$p(T|\boldsymbol{x}^{n+1}, y^{n+1}, \boldsymbol{k}) = \prod_{s \in \mathcal{I}(T)} g_{s|\boldsymbol{x}^{n+1}, y^{n+1}} \prod_{s' \in \mathcal{L}(T)} (1 - g_{s'|\boldsymbol{x}^{n+1}, y^{n+1}}). \tag{16}$$

The proof of Theorem 2 is in Appendix C. Surprisingly, the computational complexity of calculating (10) is $O(nD_{\max})$ by using this theorem.

3. Proposed Method

In this section, we introduce our proposed method. Here, we reconsider the general setup where $\mathcal{M} = \mathcal{T} \times \mathcal{K}$. Recall that the Bayes decision (7) is

$$\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) = \arg\max_{y_{n+1}} \sum_{\boldsymbol{k} \in \mathcal{K}} p(\boldsymbol{k}|\boldsymbol{x}^n, y^n) \tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k}), \tag{17}$$

where $\tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k})$ is defined as in (10). In (17), $\tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k})$ can be calculated in the same way as in Section 2.4. However, regarding the calculation of (17), we need to calculate $\tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k})$ for all $\boldsymbol{k}$ that satisfies $p(\boldsymbol{k}) > 0$. Because this computational cost increases *exponentially* depending to $D_{\max}$, it is hard to calculate all of them in general. To reduce the computational complexity of (17), we restrict the model tree candidate set to the set represented by multiple meta-trees. The advantages of building multiple meta-trees are as follows:

- As we have explained in Section 2.4.2, the number of model trees represented by each of the meta-trees increases exponentially according to the depth of meta-trees. In addition, the number of model trees in multiple meta-trees is constant times larger than those contained in a single meta-tree.
- If the true model tree is a sub-tree of any of the meta-trees, the statistically optimal prediction—optimal prediction based on the Bayes decision theory—can be calculated.
- If we build B meta-trees, the computational cost of it in learning parameters is $O(nBD_{\max})$, which is the same as that of building B function trees.

Now, we introduce the next assumption.

Assumption 3. *For B meta-trees $\mathrm{M}_{T_1,\boldsymbol{k}_1}, \ldots, \mathrm{M}_{T_B,\boldsymbol{k}_B}$, we define $\mathcal{K}' := \{\boldsymbol{k}_1, \ldots, \boldsymbol{k}_B\}$. Then, we assume a uniform distribution on $\mathcal{K}'$ as a prior probability distribution of $\boldsymbol{k} \in \mathcal{K}$.*

An example of $\mathcal{K}'$ is shown in Figure 4.

Next, we prove the following lemma about computing a posterior probability distribution of $\boldsymbol{k}$. In fact, the recursive function we have defined in Definition 3 is also useful to calculate a posterior probability distribution on $\mathcal{K}'$ sequentially. The proof of Lemma 2 is in Appendix D.

Lemma 2. *By decomposing $\boldsymbol{x}^n$ and y^n into the set of i-th data $(\boldsymbol{x}_i, y_i)(i = 1, \ldots, n)$ and performing the recursive calculation of $\tilde{q}_s(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}})$ as in (13), a posterior probability distribution of $\boldsymbol{k} \in \mathcal{K}'$ is expressed as follows:*

$$p(\boldsymbol{k}|\boldsymbol{x}^n, y^n) \propto \prod_{i=1}^{n} \tilde{q}_{s_\lambda}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}}). \tag{18}$$

From Theorem 2 and Lemma 2, we immediately obtain the next theorem.

Theorem 3. *Let us consider the model tree candidate set represented by B meta-trees, i.e.,*

$$\bigcup_{b=1}^{B} \left(\mathcal{T}_{\mathrm{M}_{T_b,\boldsymbol{k}_b}} \times \{\boldsymbol{k}_b\} \right). \tag{19}$$

Then, if the true model tree is included in (19)*, the Bayes decision* $\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$ *can be calculated as*

$$\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) = \arg\max_{y_{n+1}} \sum_{\boldsymbol{k} \in \mathcal{K}'} p(\boldsymbol{k}|\boldsymbol{x}^n, y^n) \tilde{q}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k}) \tag{20}$$

$$= \arg\max_{y_{n+1}} \sum_{\boldsymbol{k} \in \mathcal{K}'} \tilde{q}_{s_\lambda}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \mathrm{M}_{T,\boldsymbol{k}}) \prod_{i=1}^{n} \tilde{q}_{s_\lambda}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}}). \tag{21}$$

If the true model tree is not included in (19)*, the right-hand side of* (21) *is a sub-optimal prediction of the Bayes decision.*

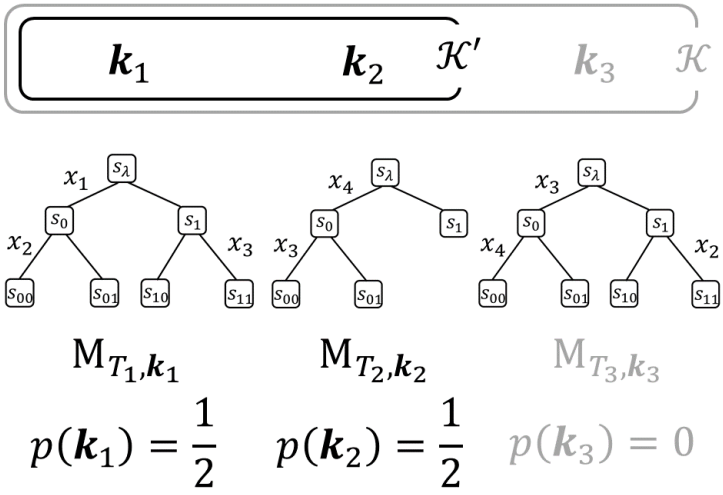

Figure 4. An example of $\mathcal{K} = \{\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3\}$ and $\mathcal{K}' = \{\boldsymbol{k}_1, \boldsymbol{k}_2\}$. In this example, the prior probability $p(\boldsymbol{k}_1) = p(\boldsymbol{k}_2) = 1/2$ and $p(\boldsymbol{k}_3) = 0$.

By using Theorem 3, we can calculate the optimal (possibly sub-optimal) prediction of the Bayes decision effectively. We name this algorithm Meta-Tree Random Forest (MTRF). We summarize MTRF in Algorithm 1.

Algorithm 1 MTRF

Input: $\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, B, g_s, \boldsymbol{\alpha}_s$
Output: $\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$
Initialize parameters g_s.
Construct B meta-trees $\mathrm{M}_{T_1,\boldsymbol{k}_1}, \ldots, \mathrm{M}_{T_B,\boldsymbol{k}_B}$ and $\mathcal{K}' := \{\boldsymbol{k}_1, \ldots, \boldsymbol{k}_B\}$.
for all $\boldsymbol{k} \in \mathcal{K}'$ **do**
  for $i = 1, \ldots, n, n+1$ **do**
    Calculate $\tilde{q}_{s_\lambda}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}})$ with (13).
    if $i \neq n+1$ **then**
      Renew parameters $g_{s|\boldsymbol{x}^i,y^i}$ with (14).
    end if
  end for
end for
Calculate $\delta^*(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$ with (21).

Remark 2. *One way to construct multiple meta-trees is to use the ideas of Random Forest. Random Forest builds function trees $\mathcal{FT}$ with the impurity (e.g., the entropy and the Gini coefficient) from the training data $\boldsymbol{x}^n$ and y^n. By regarding $(T_1, \boldsymbol{k}_1) \dots (T_B, \boldsymbol{k}_B)$ that are built by Random Forest as meta-trees $\mathrm{M}_{T_1,\boldsymbol{k}_1} \dots \mathrm{M}_{T_B,\boldsymbol{k}_B}$, we can construct B meta-trees. In our experiments in Section 5, we adopt this method. The computational complexity of MTRF is $O(nBD_{\max})$ in learning processes, which is the same as that of Random Forest. This is computable unless n, B, or $D_{\max}$ are not extremely huge. In addition, we can parallelize the procedure on each meta-tree.*

4. CART and Random Forest Revisited and Comparison with MTRF

4.1. CART and Random Forest Revisited

Although CART [1] and Random Forest [2] do not assume the model tree and they use trees to express a function $y = f(\boldsymbol{x}; T)$ (i.e., function tree), they can be regarded as methods of constructing a model tree candidate set.

First, we consider algorithms that use a single function tree, such as CART [1]. Such algorithms can be regarded as selecting a single model tree and predicting y_{n+1} by using the single function tree that corresponds to the single model tree. For example, the prediction of CART—denoted by $\delta_{\mathrm{CART}}(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$—is given by

$$\delta_{\mathrm{CART}}(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) = \arg\max_{y_{n+1}} p(y_{n+1}|\boldsymbol{x}_{n+1}, \hat{\boldsymbol{\theta}}(\boldsymbol{x}^n, y^n), \hat{T}, \hat{\boldsymbol{k}}), \tag{22}$$

where $(\hat{T}, \hat{\boldsymbol{k}})$ and $\hat{\boldsymbol{\theta}}(\boldsymbol{x}^n, y^n)$ denote an estimated model tree and tree parameter, respectively.

Second, as another example, let us consider Random Forest [2]. Random Forest can be regarded as selecting multiple model trees and predicting y_{n+1} by weighting the multiple function trees that correspond to the multiple model trees. It should be noted that the number of function trees is equal to the number of model trees in Random Forest. The prediction of Random Forest—denoted by $\delta_{\mathrm{RF}}(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1})$—is given by

$$\delta_{\mathrm{RF}}(\boldsymbol{x}^n, y^n, \boldsymbol{x}_{n+1}) = \arg\max_{y_{n+1}} \sum_{(T,\boldsymbol{k}) \in \widehat{\mathcal{M}}} \frac{1}{|\widehat{\mathcal{M}}|} p(y_{n+1}|\boldsymbol{x}_{n+1}, \hat{\boldsymbol{\theta}}(\boldsymbol{x}^n, y^n), T, \boldsymbol{k}), \tag{23}$$

where $\widehat{\mathcal{M}} = \{(\hat{T}_1, \hat{\boldsymbol{k}}_1), \dots, (\hat{T}_B, \hat{\boldsymbol{k}}_B)\} \subset \mathcal{M}$ denotes the set of model trees that are constructed by Random Forest.

4.2. Comparison of Random Forest with MTRF

Let us consider the situation where the trees that represent the function trees constructed by Random Forest are meta-trees of MTRF. Then, the size of model tree candidate set is far larger in MTRF than in Random Forest.

Random Forest can be regarded as approximating (9), (10) and (17) as in (23). Further, the approximation of the weight $p(T, \boldsymbol{k}|\boldsymbol{x}^n, y^n) \approx 1/|\widehat{\mathcal{M}}|$ is not accurate and the optimality of each function tree's weight for predicting y_{n+1} has not usually been discussed. In contrast, MTRF calculates the Bayes decision in (9) and (10) exactly and approximates (17) as in (21). Moreover, the optimal weights of T and $\boldsymbol{k}$ are given by (16) and (18), respectively.

5. Experiments

Most machine learning algorithms require rich numerical experiments to confirm the improvement independent of a specific dataset. However, in our paper, such an improvement is theoretically explained in Sections 3 and 4. If we redefine all the function trees constructed by any existing method as meta-trees, we can necessarily improve it in a sense of the Bayes risk function since our model tree candidate set contains all the model trees that correspond to the original function trees. This improvement is independent of both datasets and the original methods. Therefore, we perform our experiment only on three types of datasets and compare with the Random Forest. Similar results should be given in any other situation.

5.1. Experiment 1

Although the theoretical superiority of MTRF to Random Forest has been explained in Sections 3 and 4, we performed Experiment 1 to make sure their performance on synthetic data. The procedure of Experiment 1 is the following:

1. We initialize the hyper-parameter $\boldsymbol{\alpha}_s = (1/|\mathcal{X}|, \ldots, 1/|\mathcal{X}|)$, which is assigned on each node s.
2. To make true model trees diverse, we determinate an index of a true feature assign vector $\boldsymbol{k}$ and the true model trees as below:
   - In the true model tree (A), we assign x_j to the node whose depth is j.
     - true model tree (A-1): the complete 2-ary model tree with depth 4.
     - true model tree (A-2): the regular 2-ary model tree with depth 4. This model tree holds the structure that only left nodes have child nodes and right nodes does not.
   - In the true model tree (B), we assign all different variables to the node.
     - true model tree (B-1): the complete 2-ary model tree with depth 4.
3. We determine parameters $\boldsymbol{\theta}_s$ according to the Dirichlet distribution.
4. We generate K-dimensional n training data from the true model tree and generate K-dimensional 100 test data from the true model tree. From (7), Bayes decision takes the same value for any distribution $p(\boldsymbol{x}^n)$. Therefore, in our experiments, we set a probability distribution of $\boldsymbol{x}$ as follows: Let $U(\{0,1\})$ denote the uniform distribution on $\{0,1\}$. Then, a probability distribution of $\boldsymbol{x}$ is expressed as
$$p(x_{ij}) \overset{\text{i.i.d.}}{\sim} U(\{0,1\})(i = 1, \ldots, n, j = 1, \ldots, K). \tag{24}$$
5. We perform Random Forest that utilizes B function trees $(T_1, \boldsymbol{k}_1), \ldots, (T_B, \boldsymbol{k}_B)$. We set its impurity as the entropy and max-depth as $D_{\max}$. Afterward, we make B meta-trees $\mathrm{M}_{T_1,\boldsymbol{k}_1}, \ldots, \mathrm{M}_{T_B,\boldsymbol{k}_B}$. (Of course, their max-depth are $D_{\max}$, too.) Then, we calculate the average prediction error rate of Random Forest and MTRF, where we set the hyper-parameter g_s as $g_s = 1/2$ in MTRF.
6. We repeat Steps 3∼5 Q times.

Figure 5 shows an example of these model trees (note that, for simplicity, we illustrate the model trees with depth 2).

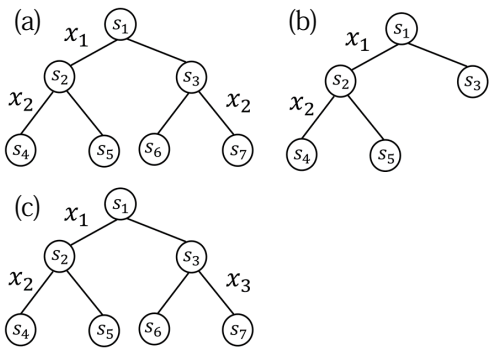

Figure 5. An example of true model tree (A-1) (**a**), model tree (A-2) (**b**), and model tree (B-1) (**c**).

We set $|\mathcal{X}| = |\mathcal{Y}| = 2$, $n = 100, 200, \ldots, 1000, K = 500, B = 10, D_{\max} = 2, 4, 6$, and $Q = 5000$. The result is shown in Figure 6.

From Figure 6, when we compare the performance of Random Forest with that of MTRF, we can confirm that MTRF outperforms Random Forest in all conditions. These results are theoretically reasonable because the trees that represent the function trees constructed by Random Forest are meta-trees of MTRF.

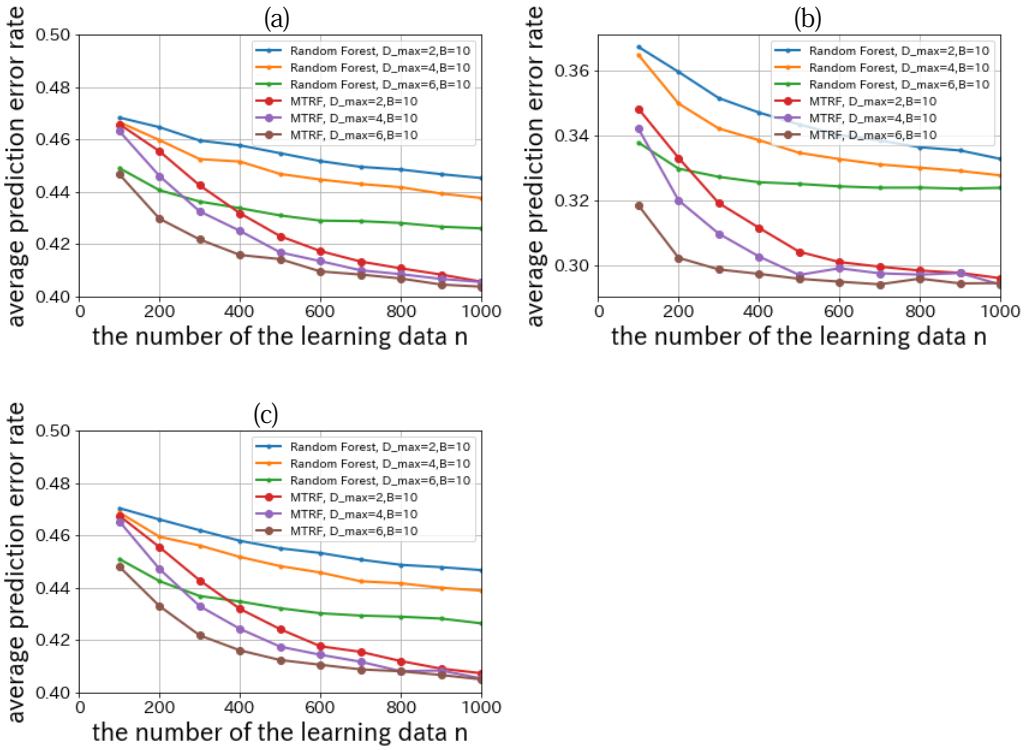

Figure 6. Relationships between the number of learning data n and the average prediction error rate where the true model tree is model (A-1) (**a**), model (A-2) (**b**), and model (B-1) (**c**).

5.2. Experiment 2

In Experiment 2, we used real data (nursery school data and mushroom data) in the UCI repository (University of California, Irvine Machine Learning Repository). The procedure of Experiment 2 is mostly the same as that of Experiment 1, but the way of sampling data is different; instead of Steps 2, 3, and 4 in Experiment 1, we randomly sampled $n + t$ data from the whole dataset and divided them to n training data and t test data. We repeated the random sampling of the data for 2500 times.

The detail on nursery school data is as follows. The explanatory variables are discrete 8 attributes of a family whose child wants to enter a nursery school (e.g., finance, health, and family structure). The dimension of each attributes is up to 5. The targeting variable is whether the nursery school should welcome him/her or not. We want to consider that a new child should be welcomed with his attributes and learning data. We set $|\mathcal{X}| = 5$, $|\mathcal{Y}| = 2$, $n = 100, 200, \ldots, 1000$, $t = 1000$, $K = 8$, $B = 5$, and $D_{\max} = 2, 3, 4$. The result is shown in Figure 7a.

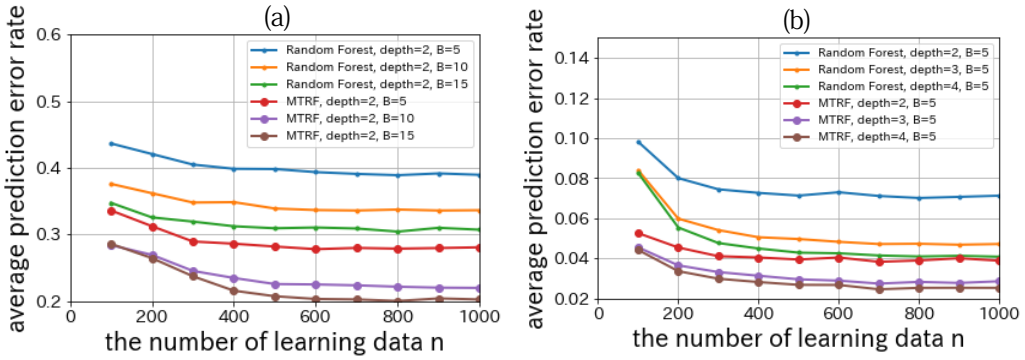

Figure 7. Relationships between the number of the learning data n and the average prediction error rate on nursery school data (**a**)/mushroom data (**b**) in UCI repository.

The detail on mushroom data is as follows. The explanatory variables are discrete 22 attributes of a mushroom (e.g., smell, shape, surface, and color). The dimension of each attributes is up to 10. The targeting variable is whether the mushroom can be eaten, not be eaten, or unknown. We want to predict that a new mushroom can be eaten, with its attributes and learning data. We set $|\mathcal{X}| = 22$, $|\mathcal{Y}| = 3$,$n = 100, 200, \ldots, 1000$, $t = 1000$, $K = 22$, $B = 5$, and $D_{\max} = 2, 3, 4$. The result is shown in Figure 7b.

From Figure 7, we can confirm MTRF outperforms Random Forest in all conditions.

6. Conclusions and Future Work

In this paper, we have considered a model tree and derived the Bayes decision. We have addressed the computational complexity problem of the Bayes decision by introducing a notion of meta-tree, and proposed MTRF (Meta-Tree Random Forest), which uses multiple meta-trees. As we have shown in Theorem 3, if the true model tree is included in the model tree candidate set represented by multiple meta-trees, MTRF calculates the Bayes decision with efficient computational cost. Even if the true model tree is not included in the model tree candidate set represented by multiple meta-trees, MTRF gives the approximation of the Bayes decision. The advantages of MTRF have been theoretically analyzed. In Section 4, we have explained that, if we redefine all the function trees constructed by Random Forest as meta-trees, we can necessarily improve it in a sense of the Bayes risk function since our model tree candidate set contains all the model trees that correspond to the original function trees. We have performed experiments in Section 5 to check the theoretical analysis.

As we have described in Section 1, it is desirable that we construct the model tree candidate set that includes as many model trees as possible within the allowed computational cost. This is because if the true model tree is included in the model tree candidate set, the Bayes decision is calculated. Hence, the main problem of MTRF is how to construct multiple meta-trees. In Section 5, we have constructed multiple meta-trees by using the idea of Random Forest. However, we can consider other ways to construct multiple meta-trees. One of the possible future directions is to use other ideas, such as Boosting. This is one of the future works in our research.

Author Contributions: Conceptualization, N.D., S.S., Y.N. and T.M.; Methodology, N.D., S.S., Y.N. and T.M.; Software, N.D. and Y.N.; Validation, N.D., S.S., Y.N. and T.M.; Formal Analysis, N.D., S.S., Y.N. and T.M.; Investigation, N.D., S.S., Y.N. and T.M.; Resources, N.D.; Data Curation, N.D.; Writing—Original Draft Preparation, N.D., S.S. and Y.N.; Writing—Review & Editing, N.D., S.S., Y.N. and T.M.; Visualization, N.D., S.S. and Y.N.; Supervision, T.M.; Project Administration, T.M.; Funding Acquisition, S.S. and T.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by JSPS KAKENHI Grant Numbers JP17K06446, JP19K04914 and JP19K14989.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://archive.ics.uci.edu/ml/datasets/Nursery and https://archive.ics.uci.edu/ml/datasets/Mushroom (accessed on 8 June 2021).

Acknowledgments: The authors would like to thank all the members of Matsushima Lab. for their comments and discussions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviation is used in this manuscript:

MTRF Meta-Tree Random Forest

Appendix A. Proof of Theorem 1

Proof of Theorem 1. From (4)–(6), and Bayes' theorem, we have

$$\begin{aligned}&\mathrm{BR}(\delta(\cdot,\cdot,\cdot))\\&= \sum_{\boldsymbol{x}^n\in\mathcal{X}^{Kn}}\sum_{y^n\in\mathcal{Y}^n}\sum_{\boldsymbol{x}_{n+1}\in\mathcal{X}^K} p(\boldsymbol{x}_{n+1})\left(\sum_{y_{n+1}\in\mathcal{Y}} p(y_{n+1}|\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1})l(y_{n+1},\delta(\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1}))\right)\\&\quad\cdot p(\boldsymbol{x}^n,y^n),\end{aligned} \tag{A1}$$

where

$$p(y_{n+1}|\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1}) = \sum_{(T,\boldsymbol{k})\in\mathcal{M}}\int_{\boldsymbol{\Theta}} p(y_{n+1}|\boldsymbol{x}_{n+1},\boldsymbol{\theta},T,\boldsymbol{k})p(\boldsymbol{\theta},T,\boldsymbol{k}|\boldsymbol{x}^n,y^n)d\boldsymbol{\theta}. \tag{A2}$$

We need to calculate δ that minimizes the brackets in (A1) when we minimize the Bayes risk. Let $a=\delta(\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1})$ and $F(a)$ denote the formula in the bracket of (A1). Then,

$$\begin{aligned}F(a) &= \sum_{y_{n+1}\in\mathcal{Y}} l(y_{n+1},a)p(y_{n+1}|\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1})\\&= \sum_{y_{n+1}\in\mathcal{Y}\setminus\{a\}} 1\cdot p(y_{n+1}|\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1}).\end{aligned} \tag{A3}$$

Therefore, the decision function a that minimizes $F(a)$ is

$$a = \arg\max_{y_{n+1}} p(y_{n+1}|\boldsymbol{x}^n,y^n,\boldsymbol{x}_{n+1}). \tag{A4}$$

We can prove (7) by substituting (A2) for (A4). □

Appendix B. Proof of Lemma 1

Proof of Lemma 1. Let $\boldsymbol{\Theta}_{s(\boldsymbol{x}_{n+1})}$ denote the space of parameters on $s(\boldsymbol{x}_{n+1})$. In addition, we define

$$\boldsymbol{\Theta}_{s\neq s(\boldsymbol{x}_{n+1})} := \boldsymbol{\Theta}\setminus\boldsymbol{\Theta}_{s(\boldsymbol{x}_{n+1})}. \tag{A5}$$

Then, we can rewrite the left-hand side of (11) as below:

$$\begin{aligned}q(y_{n+1}|\boldsymbol{x}_{n+1},\boldsymbol{x}^n,y^n,T,\boldsymbol{k}) &= \int_{\boldsymbol{\Theta}} p(y_{n+1}|\boldsymbol{x}_{n+1},\boldsymbol{\theta},T,\boldsymbol{k})p(\boldsymbol{\theta}|\boldsymbol{x}^n,y^n,T,\boldsymbol{k})\mathrm{d}\boldsymbol{\theta}\\&= \int_{\boldsymbol{\Theta}_{s(\boldsymbol{x}_{n+1})}}\theta_{y_{n+1}|s(\boldsymbol{x}_{n+1})}\int_{\boldsymbol{\Theta}_{s\neq s(\boldsymbol{x}_{n+1})}} p(\boldsymbol{\theta}|\boldsymbol{x}^n,y^n,T,\boldsymbol{k})\mathrm{d}\boldsymbol{\theta}_{s\neq s(\boldsymbol{x}_{n+1})}\mathrm{d}\boldsymbol{\theta}_{s(\boldsymbol{x}_{n+1})},\end{aligned} \tag{A6}$$

where the last equality follows from (1).

Looking into the term $\int_{\boldsymbol{\Theta}_{s\neq s(\boldsymbol{x}_{n+1})}} p(\boldsymbol{\theta}|\boldsymbol{x}^n,y^n,T,\boldsymbol{k})\mathrm{d}\boldsymbol{\theta}_{s\neq s(\boldsymbol{x}_{n+1})}$ in (A6), we have

$$\begin{aligned}&\int_{\boldsymbol{\Theta}_{s\neq s(\boldsymbol{x}_{n+1})}} p(\boldsymbol{\theta}|\boldsymbol{x}^n,y^n,T,\boldsymbol{k})\mathrm{d}\boldsymbol{\theta}_{s\neq s(\boldsymbol{x}_{n+1})}\\&\overset{(a)}{\propto}\int_{\boldsymbol{\Theta}_{s\neq s(\boldsymbol{x}_{n+1})}} p(\boldsymbol{\theta}|T,\boldsymbol{k})p(y^n|\boldsymbol{x}^n,\boldsymbol{\theta},T,\boldsymbol{k})\mathrm{d}\boldsymbol{\theta}_{s\neq s(\boldsymbol{x}_{n+1})} &\text{(A7)}\\&= \int_{\boldsymbol{\Theta}_{s\neq s(\boldsymbol{x}_{n+1})}}\prod_{s\in\mathcal{S}} p(\boldsymbol{\theta}_s|\boldsymbol{\alpha}_s)\prod_{i=1}^n\theta_{y_i|s(\boldsymbol{x}_i)}\mathrm{d}\boldsymbol{\theta}_{s\neq s(\boldsymbol{x}_{n+1})} &\text{(A8)}\\&\propto p(\boldsymbol{\theta}_{s(\boldsymbol{x}_{n+1})}|\boldsymbol{\alpha}_{s(\boldsymbol{x}_{n+1})})\prod_{i\in\{j\in\mathbb{N}:s(\boldsymbol{x}_j)=s(\boldsymbol{x}_{n+1})\}}\theta_{y_i|s(\boldsymbol{x}_i)}, &\text{(A9)}\end{aligned}$$

where (a) follows from Bayes' theorem.

Because we can transform the right-hand side of (11) in the same way, we obtain (11). □

Appendix C. Proof of Theorem 2

Proof of Theorem 2. We prove Theorem 2 with mathematical induction according to the following three steps:

Step 1: We prove (15) for $n = 0$ with Lemmas A1 and A2.

Step 2: We prove (16) for $n = 0$ by using the result of Step 1.

Step 3: If we assume (15) and (16) for $n = N$, we can prove (15) and (16) for $n = N + 1$ by repeating Steps 1 and 2.

In the following, we give a proof of each step.

Step 1: First, we prove (15) for $n = 0$. Let $\mathcal{S}(\boldsymbol{x}_i)$ $(i = 1, \ldots, n+1)$ denote the set of all nodes on the path of $\boldsymbol{x}_i$ in the meta-tree $\mathrm{M}_{T,\boldsymbol{k}}$. Let $\mathcal{T}_s := \{T \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}} | s \in \mathcal{L}(T)\}$. Then, we transform $\tilde{q}(y_1|\boldsymbol{x}_1, \boldsymbol{k})$ as below:

$$\tilde{q}(y_1|\boldsymbol{x}_1, \boldsymbol{k}) \overset{(a)}{=} \sum_{T \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}} p(T) q(y_1|\boldsymbol{x}_1, T, \boldsymbol{k}) \tag{A10}$$

$$= \sum_{s \in \mathcal{S}(\boldsymbol{x}_1)} \sum_{T \in \mathcal{T}_s} p(T) q(y_1|\boldsymbol{x}_1, T, \boldsymbol{k}) \tag{A11}$$

$$\overset{(b)}{=} \sum_{s \in \mathcal{S}(\boldsymbol{x}_1)} q_s(y_1|\boldsymbol{x}_1, \boldsymbol{k}) \sum_{T \in \mathcal{T}_s} p(T), \tag{A12}$$

where (a) follows from (10), and (b) follows from Lemma 1. The right-hand side of (A12) contains the summation with respect to T as below:

$$\sum_{T \in \mathcal{T}_s} p(T). \tag{A13}$$

Regarding (A13), we prove the following lemmas.

Lemma A1. *The prior probability distribution $p(T)$ assumed in (12) satisfies the next property:*

$$\sum_{T \in \mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}} p(T) = 1. \tag{A14}$$

Proof of Lemma A1. In this proof, for simplicity, we use $\mathcal{T}$ instead of $\mathcal{T}_{\mathrm{M}_{T,\boldsymbol{k}}}$. Before showing the formal proof, we consider an example. Let us consider the set of tree $\mathcal{T}$ shown in Figure A1. In this example, $\mathcal{T}$ is composed of five model trees. Among these model trees, T_1 is included in $\{[s_\lambda]\}$ and T_2–T_5 are included in $T \in \mathcal{T} \backslash \{[s_\lambda]\}$. Therefore, the next equation holds:

$$\sum_{T \in \mathcal{T}} p(T) = \sum_{T \in \mathcal{T}} \left(\prod_{s \in \mathcal{I}(T)} g_s \prod_{s' \in \mathcal{L}(T)} (1 - g_{s'}) \right) \tag{A15}$$

$$= (1 - g_{s_\lambda}) + g_{s_\lambda} \sum_{T \in \{T_2, T_3, T_4, T_5\}} \left(\prod_{s \in \mathcal{L}(T)} (1 - g_s) \prod_{s' \in \mathcal{I}(T) \backslash \{s_\lambda\}} g_{s'} \right) \tag{A16}$$

$$\begin{aligned} &= (1 - g_{s_\lambda}) + g_{s_\lambda} [(1 - g_{s_0})(1 - g_{s_1}) + g_{s_0}(1 - g_{s_{00}})(1 - g_{s_{01}})(1 - g_{s_1}) \\ &+ (1 - g_{s_0}) g_{s_1} (1 - g_{s_{10}})(1 - g_{s_{11}}) + g_{s_0}(1 - g_{s_{00}})(1 - g_{s_{01}}) g_{s_1} (1 - g_{s_{10}})(1 - g_{s_{11}})] \end{aligned} \tag{A17}$$

$$= (1 - g_{s_\lambda}) + g_{s_\lambda} [\{(1 - g_{s_0}) + g_{s_0}(1 - g_{s_{00}})(1 - g_{s_{01}})\}\{(1 - g_{s_1}) + g_{s_1}(1 - g_{s_{10}})(1 - g_{s_{11}})\}] \tag{A18}$$

$$\overset{(a)}{=} (1 - g_{s_\lambda}) + g_{s_\lambda} [\{(1 - g_{s_0}) + g_{s_0}\}\{(1 - g_{s_1}) + g_{s_1}\}] \tag{A19}$$

$$= 1, \tag{A20}$$

where (a) follows from $g_s = 0$ for a leaf node s of the meta-tree (see Assumption 2).

Now, we give the formal proof. Because

$$\sum_{T\in\mathcal{T}} p(T) = \sum_{T\in\mathcal{T}} \left(\prod_{s\in\mathcal{I}(T)} g_s \prod_{s'\in\mathcal{L}(T)} (1-g_{s'}) \right), \tag{A21}$$

we prove

$$\sum_{T\in\mathcal{T}} \left(\prod_{s\in\mathcal{I}(T)} g_s \prod_{s'\in\mathcal{L}(T)} (1-g_{s'}) \right) = 1 \tag{A22}$$

by induction.

Let $[s_\lambda]$ denote the tree that consists of only the root node s_λ of T. Then, we have

$$\begin{aligned} &\sum_{T\in\mathcal{T}} \left(\prod_{s\in\mathcal{L}(T)} (1-g_s) \prod_{s'\in\mathcal{I}(T)} g_{s'} \right) \\ &\quad = (1-g_{s_\lambda}) + g_{s_\lambda} \sum_{T\in\mathcal{T}\setminus\{[s_\lambda]\}} \left(\prod_{s\in\mathcal{L}(T)} (1-g_s) \prod_{s'\in\mathcal{I}(T)\setminus\{s_\lambda\}} g_{s'} \right). \end{aligned} \tag{A23}$$

Because each tree $T \in \mathcal{T} \setminus \{[s_\lambda]\}$ is identified by $|\mathcal{X}|$ sub-trees whose root nodes are the child nodes of s_λ, let $s_{\lambda\,\mathrm{child},j}$ denote the j-th child node of s_λ for $0 \le j \le |\mathcal{X}|-1$ and $\mathcal{T}^{s_{\lambda\,\mathrm{child},j}}$ denote the set of sub-trees whose root node is $s_{\lambda\,\mathrm{child},j}$. Figure A1 shows an example of $\mathcal{T}^{s_{\lambda\,\mathrm{child},0}}$ and $\mathcal{T}^{s_{\lambda\,\mathrm{child},1}}$. Then, the summation in the right-hand side of (A23) is factorized as

$$\prod_{j=0}^{|\mathcal{X}|-1} \sum_{T_j\in\mathcal{T}^{s_{\lambda\,\mathrm{child},j}}} \left(\prod_{s\in\mathcal{L}(T_j)} (1-g_s) \prod_{s'\in\mathcal{I}(T_j)} g_{s'} \right). \tag{A24}$$

Consequently, because the goal is to show (A22), Lemma A1 is proven by induction. □

Lemma A2. *Let $\mathcal{A}_s$ denote the set of all ancestor nodes of s. Then, we have*

$$\sum_{T\in\mathcal{T}_s} p(T) = (1-g_s) \prod_{s'\in\mathcal{A}_s} g_{s'}. \tag{A25}$$

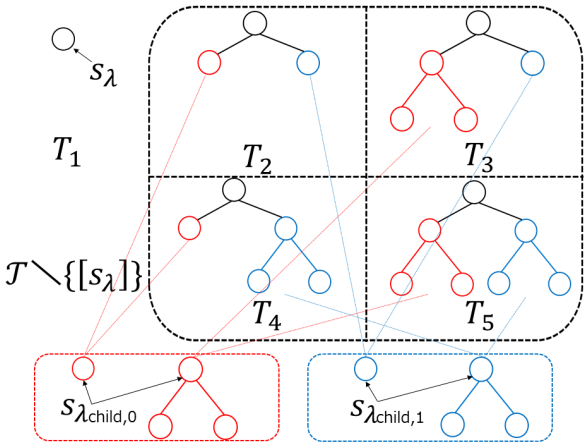

Figure A1. An illustration of $\mathcal{T}^{s_{\lambda\,\mathrm{child},0}}$ (left lower enclosure) and $\mathcal{T}^{s_{\lambda\,\mathrm{child},1}}$ (right lower enclosure). In the trees T_2–T_5, s_λ has child nodes $s_{\lambda\,\mathrm{child},0}$ and $s_{\lambda\,\mathrm{child},1}$. Therefore, we can decompose the child nodes of s_λ in the trees T_2–T_5 into the nodes included in $\mathcal{T}^{s_{\lambda\,\mathrm{child},0}}$ and the nodes included in $\mathcal{T}^{s_{\lambda\,\mathrm{child},1}}$.

Proof of Lemma A2.

$$\sum_{T\in\mathcal{T}_s} p(T) \overset{(a)}{=} \sum_{T\in\mathcal{T}_s} \prod_{s'\in\mathcal{I}(T)} g_{s'} \prod_{s''\in\mathcal{L}(T)} (1-g_{s''}) \tag{A26}$$

$$\overset{(b)}{=} (1-g_s) \prod_{s'\in\mathcal{A}_s} g_{s'} \sum_{T\in\mathcal{T}_s} \prod_{s''\in\mathcal{I}(T)\setminus\mathcal{A}_s} g_{s''} \prod_{s'''\in\mathcal{L}(T)\setminus\{s\}} (1-g_{s'''}), \tag{A27}$$

where (a) follows from Assumption 2, and (b) follows from the fact that all trees $T \in \mathcal{T}_s$ have the node s and its ancestor nodes.

Further, as we will prove later, it holds that

$$\sum_{T\in\mathcal{T}_s} \prod_{s''\in\mathcal{I}(T)\setminus\mathcal{A}_s} g_{s''} \prod_{s'''\in\mathcal{L}(T)\setminus\{s\}} (1-g_{s'''}) = 1. \tag{A28}$$

Thus, we obtain (A25) by combining (A27) and (A28).

To prove (A28), we can apply the factorization of Lemma A1 to (A28) if we regard the node whose parent node is s or an element of $\mathcal{A}_s$ as a root node, as shown in Figure A2. Let J denote the number of that factorized trees, and let r_j denote the root node of j-th factorized tree. Then, we can rewrite (A28) as follows:

$$\sum_{T\in\mathcal{T}_s} \prod_{s''\in\mathcal{I}(T)\setminus\mathcal{A}_s} g_{s''} \prod_{s'''\in\mathcal{L}(T)\setminus\{s\}} (1-g_{s'''}) = \prod_{j=0}^{J-1} \sum_{T_j\in\mathcal{T}^{r_j}} \left(\prod_{s\in\mathcal{L}(T_j)} (1-g_s) \prod_{s'\in\mathcal{I}(T_j)} g_{s'} \right) = 1. \tag{A29}$$

Hence, the proof is complete. □

Now, we transform (A12) as follows.

$$\sum_{s\in\mathcal{S}(\boldsymbol{x}_1)} q_s(y_1|\boldsymbol{x}_1,\boldsymbol{k}) \sum_{T\in\mathcal{T}_s} p(T) \overset{(a)}{=} \sum_{s\in\mathcal{S}(\boldsymbol{x}_1)} q_s(y_1|\boldsymbol{x}_1,\boldsymbol{k})(1-g_s) \prod_{s'\in\mathcal{A}_s} g_{s'} \tag{A30}$$

$$= (1-g_{s_\lambda}) q_{s_\lambda}(y_1|\boldsymbol{x}_1,\boldsymbol{k}) + g_{s_\lambda} \sum_{s\in\mathcal{S}(\boldsymbol{x}_1)\setminus\{s_\lambda\}} q_s(y_1|\boldsymbol{x}_1,\boldsymbol{k})(1-g_s) \prod_{s'\in\mathcal{A}_s\setminus\{s_\lambda\}} g_{s'}, \tag{A31}$$

where (a) follows from Lemma A2.

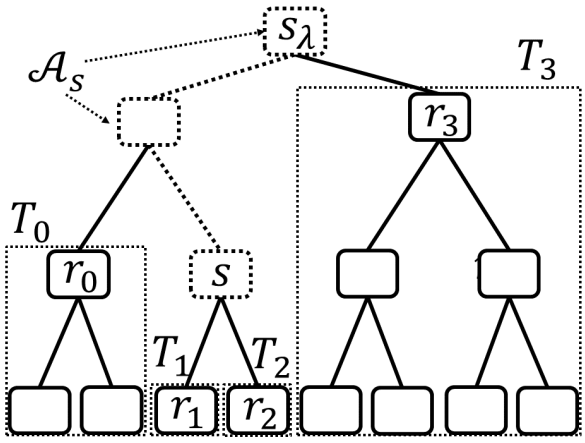

Figure A2. An illustration of r_0,r_1,r_2, and r_3. In this example, the dotted line represents the path from s_λ to s and $\mathcal{A}_s$ is the set of two parent nodes of s. Furthermore, the rectangles of the dotted line represent four factorized trees T_0, T_1, T_2, and T_3.

Here, let $s_{\lambda_{\text{child}}}$ denote the node that is a child node of s_λ and an element of $\mathcal{S}(\boldsymbol{x}_1)$. Then, we can decompose

$$\begin{aligned}
&\sum_{s\in\mathcal{S}(\boldsymbol{x}_1)\setminus\{s_\lambda\}} q_s(y_1|\boldsymbol{x}_1,\boldsymbol{k})(1-g_s)\prod_{s'\in\mathcal{A}_s\setminus\{s_\lambda\}} g_{s'}\\
&=\left(1-g_{s_{\lambda\,\mathrm{child}}}\right)q_{s_{\lambda\,\mathrm{child}}}(y_1|\boldsymbol{x}_1,\boldsymbol{k})\\
&+g_{s_{\lambda\,\mathrm{child}}}\sum_{s\in\mathcal{S}(\boldsymbol{x}_1)\setminus\{s_\lambda,s_{\lambda\,\mathrm{child}}\}} q_s(y_1|\boldsymbol{x}_1,\boldsymbol{k})(1-g_s)\prod_{s'\in\mathcal{A}_s\setminus\{s_\lambda,s_{\lambda\,\mathrm{child}}\}} g_{s'}. \qquad \text{(A32)}
\end{aligned}$$

The equation (A32) has a similar structure to (A31). Like this, we can decompose (A31) recursively to the leaf node that is included in $\mathcal{S}(\boldsymbol{x}_1)$. In addition, we can utilize the assumption $g_s=0$ for any leaf nodes s of a meta-tree $\mathrm{M}_{T,\boldsymbol{k}}$. Consequently, we can prove that (A31) coincides with $\tilde{q}_{s_\lambda}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})$.

Step 2: Next, we prove (16) for $n=0$. Because the parameters g_s are updated only when their nodes are the elements of $\mathcal{S}(\boldsymbol{x}_1)$, we decompose the right-hand side of (16) for $n=0$ as follows:

$$\begin{aligned}
&\prod_{s\in\mathcal{I}(T)} g_{s|\boldsymbol{x}^1,y^1}\prod_{s'\in\mathcal{L}(T)}\left(1-g_{s'|\boldsymbol{x}^1,y^1}\right)\\
&=\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)} g_{s|\boldsymbol{x}^1,y^1}\prod_{s'\in\mathcal{L}(T)\cap\mathcal{S}(\boldsymbol{x}_1)}\left(1-g_{s'|\boldsymbol{x}^1,y^1}\right)\\
&\quad\cdot\prod_{s''\in\mathcal{I}(T)\setminus\mathcal{S}(\boldsymbol{x}_1)} g_{s''|\boldsymbol{x}^1,y^1}\prod_{s'''\in\mathcal{L}(T)\setminus\mathcal{S}(\boldsymbol{x}_1)}\left(1-g_{s'''|\boldsymbol{x}^1,y^1}\right) \qquad \text{(A33)}\\
&=\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)} g_{s|\boldsymbol{x}^1,y^1}\prod_{s'\in\mathcal{L}(T)\cap\mathcal{S}(\boldsymbol{x}_1)}\left(1-g_{s'|\boldsymbol{x}^1,y^1}\right)\prod_{s''\in\mathcal{I}(T)\setminus\mathcal{S}(\boldsymbol{x}_1)} g_{s''}\prod_{s'''\in\mathcal{L}(T)\setminus\mathcal{S}(\boldsymbol{x}_1)}(1-g_{s'''}). \qquad \text{(A34)}
\end{aligned}$$

We transform the part of (A34), whose parameters g_s are renewed by $\boldsymbol{x}^1$ and y^1, as follows:

$$\begin{aligned}
&\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)} g_{s|\boldsymbol{x}^1,y^1}\prod_{s'\in\mathcal{L}(T)\cap\mathcal{S}(\boldsymbol{x}_1)}\left(1-g_{s'|\boldsymbol{x}^1,y^1}\right)\\
&\overset{(a)}{=}\left(1-g_{s(\boldsymbol{x}_1)|\boldsymbol{x}^1,y^1}\right)\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)} g_{s|\boldsymbol{x}^1,y^1} \qquad \text{(A35)}\\
&\overset{(b)}{=}\left(1-g_{s(\boldsymbol{x}_1)}\right)\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)}\frac{g_s\tilde{q}_{s_{\mathrm{child}}}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}{\tilde{q}_s(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})} \qquad \text{(A36)}\\
&\overset{(c)}{=}\frac{\left(1-g_{s(\boldsymbol{x}_1)}\right)\tilde{q}_{s(\boldsymbol{x}_1)}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}{\tilde{q}_{s_\lambda}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)} g_s \qquad \text{(A37)}\\
&=\frac{\tilde{q}_{s(\boldsymbol{x}_1)}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}{\tilde{q}_{s_\lambda}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}\prod_{s\in\mathcal{I}(T)\cap\mathcal{S}(\boldsymbol{x}_1)} g_s\prod_{s'\in\mathcal{L}(T)\cap\mathcal{S}(\boldsymbol{x}_1)}(1-g_{s'}), \qquad \text{(A38)}
\end{aligned}$$

where (a) follows from $s'\in\mathcal{L}(T)\cap\mathcal{S}(\boldsymbol{x}_1)$ is uniquely determined as $s(\boldsymbol{x}_1)$, (b) follows from Definition 3, and (c) follows from reduction of $\tilde{q}_s(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})$.

Thus, the right-hand side of (A34) is calculated as

$$\begin{aligned}
&\frac{\tilde{q}_{s(\boldsymbol{x}_1)}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}{\tilde{q}_{s_\lambda}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})}\prod_{s\in\mathcal{I}(T)} g_s\prod_{s'\in\mathcal{L}(T)}(1-g_{s'}). \qquad \text{(A39)}\\
&\overset{(a)}{=}\frac{\tilde{q}_{s(\boldsymbol{x}_1)}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})p(T)}{\tilde{q}_{s_\lambda}(y_1|\boldsymbol{x}_1,\mathrm{M}_{T,\boldsymbol{k}})} \qquad \text{(A40)}\\
&\overset{(b)}{=}\frac{q(y_1|\boldsymbol{x}_1,T,\boldsymbol{k})p(T)}{\tilde{q}(y_1|\boldsymbol{x}_1,\boldsymbol{k})} \qquad \text{(A41)}\\
&\overset{(c)}{=}p(T|\boldsymbol{x}^1,y^1,\boldsymbol{k}), \qquad \text{(A42)}
\end{aligned}$$

where (a) follows from Assumption 2, the denominator of (b) follows from Theorem 2 for $n = 0$, the numerator of (b) follows from Definition 3 and the definition of $q_{s(\boldsymbol{x}_{n+1})}(y_{n+1}|\boldsymbol{x}_{n+1}, \boldsymbol{x}^n, y^n, \boldsymbol{k})$, and (c) follows from the Bayes' theorem.

Step 3: Finally, if we assume (15) and (16) are true when $n = N$, we can also prove (15) and (16) when $n = N + 1$ in the same way. Therefore, Theorem 2 holds. □

Appendix D. Proof of Lemma 2

Proof of Lemma 2.

$$p(\boldsymbol{k}|\boldsymbol{x}^n, y^n) \propto p(\boldsymbol{k}) p(\boldsymbol{x}^n, y^n|\boldsymbol{k}) \tag{A43}$$

$$= p(\boldsymbol{k}) \prod_{i=1}^{n} \tilde{q}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \boldsymbol{k}) p(\boldsymbol{x}_i|\boldsymbol{x}^{i-1}, y^{i-1}, \boldsymbol{k}) \tag{A44}$$

$$\overset{(a)}{\propto} \prod_{i=1}^{n} \tilde{q}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \boldsymbol{k}) \tag{A45}$$

$$\overset{(b)}{=} \prod_{i=1}^{n} \tilde{q}_{s_\lambda}(y_i|\boldsymbol{x}_i, \boldsymbol{x}^{i-1}, y^{i-1}, \mathrm{M}_{T,\boldsymbol{k}}), \tag{A46}$$

where (a) follows from Assumption 3 and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are i.i.d.; (b) follows from Theorem 2 at each $i = 1, \ldots, n$. □

References

1. Breiman, L.; Friedman, J.; Stone, C.J.; Olshen, R.A. *Classification and Regression Trees*; CRC Press: Boca Raton, FL, USA, 1984.
2. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
3. Athey, S.; Tibshirani, J.; Wager, S. Generalized Random Forests. *Ann. Stat.* **2019**, *47*, 1148–1178. [CrossRef]
4. Friedman, J.H. Stochastic Gradient Boosting. *Comput. Stat. Data Anal.* **2002**, *38*, 367–378. [CrossRef]
5. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16), San Francisco, CA, USA, 13–17 August 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 785–794. [CrossRef]
6. Schulter, S.; Wohlhart, P.; Leistner, C.; Saffari, A.; Roth, P.M.; Bischof, H. Alternating Decision Forests. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 508–515. [CrossRef]
7. Mishina, Y.; Murata, R.; Yamauchi, Y.; Yamashita, T.; Fujiyoshi, H. Boosted Random Forest. *IEICE Trans. Inf. Syst.* **2015**, *E98.D*, 1630–1636. [CrossRef]
8. Bulo, S.; Kontschieder, P. Neural Decision Forests for Semantic Image Labelling. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 81–88. [CrossRef]
9. Kontschieder, P.; Fiterau, M.; Criminisi, A.; Bulo, S.R. Deep Neural Decision Forests. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1467–1475. [CrossRef]
10. Tanno, R.; Arulkumaran, K.; Alexander, D.; Criminisi, A.; Nori, A. Adaptive Neural Trees. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 6166–6175.
11. Zhou, Z.H.; Feng, J. Deep Forest: Towards An Alternative to Deep Neural Networks. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), Melbourne, Australia, 19–25 August 2017; pp. 3553–3559. [CrossRef]
12. Frosst, N.; Hinton, G. Distilling a Neural Network Into a Soft Decision Tree. *arXiv* **2017**, arXiv:1711.09784.
13. Suko, T.; Nomura, R.; Matsushima, T.; Hirasawa, S. Prediction Algorithm for Decision Tree Model. *IEICE Tech. Rep. Theor. Found. Comput.* **2003**, *103*, 93–98. (In Japanese)
14. Berger, J.O. *Statistical Decision Theory and Bayesian Analysis*; Springer: New York, NY, USA, 1985. [CrossRef]
15. Matsushima, T.; Hirasawa, S. A Bayes Coding Algorithm Using Context Tree. In Proceedings of the 1994 IEEE International Symposium on Information Theory, Trondheim, Norway, 27 June–1 July 1994; p. 386. [CrossRef]
16. Nakahara, Y.; Matsushima, T. A Stochastic Model of Block Segmentation Based on the Quadtree and the Bayes Code for It. In Proceedings of the 2020 Data Compression Conference (DCC), Snowbird, UT, USA, 24–27 March 2020; pp. 293–302. [CrossRef]