Acta Mathematica Sinica, English Series
Feb., 2025, Vol. 41, No. 2, pp. 498–521
Published online: February 15, 2025
https://doi.org/10.1007/s10114-025-3349-5
http://www.ActaMath.com

Acta Mathematica Sinica, English Series

Segmented Linear Regression Trees

Xiangyu Zheng
JD Technology, Beijing 100176*, P. R. China*
E-mail: *zhengxiangyu8@jd.com*

Songxi Chen[1)]
Department of Statistics and Data Science, Tsinghua University, Beijing 100084*, P. R. China*
E-mail: *sxchen@tsinghua.edu.cn*

Abstract Tree-based models have been widely applied in both academic and industrial settings due to the natural interpretability, good predictive accuracy, and high scalability. In this paper, we focus on improving the single-tree method and propose the segmented linear regression trees (SLRT) model that replaces the traditional constant leaf model with linear ones. From the parametric view, SLRT can be employed as a recursive change point detect procedure for segmented linear regression (SLR) models, which is much more efficient and flexible than the traditional grid search method. Along this way, we propose to use the conditional Kendall's τ correlation coefficient to select the underlying change points. From the non-parametric view, we propose an efficient greedy splitting method that selects the splits by analyzing the association between residuals and each candidate split variable. Further, with the SLRT as a single-tree predictor, we propose a linear random forest approach that aggregates the SLRTs by a weighted average. Both simulation and empirical studies showed significant improvements than the CART trees and even the random forest.

Keywords Regression trees, segmented linear regression, split selection, tree pruning

MR(2020) Subject Classification 62F10

1 Introduction

Tree-based models are a class of predictive modeling approaches that use decision trees to visually represent a series of binary rules and generate predictions. Research and applications in tree-based models have seen rapid growth due to natural interpretability, good predictive accuracy, and high scalability. Our work focuses on the application and extension of tree-based methods in the regression scenario.

The first regression tree algorithm can be traced back to Automatic Interaction Detection (AID) in [22]. The AID method proposed a recursive binary partitioning algorithm to construct a regression tree. Starting at the root node including the full data, AID recursively splits the data in each node into two children nodes according to whether $X_j \le c$ or $X_j > c$, where $X_j \in \{X_i\}_{i=1}^p$ is the split variable and c is the split boundary. At any node t, let $I(t)$ denote the sum of squared deviations in node t, i.e., $I(t) = \sum_{y_i \in t}(y_i - \bar{y}_t)^2$, where $\bar{y}_t$ denotes the mean

Received July 15, 2023, revised November 5, 2023, accepted November 24, 2023
1) Corresponding author

value inside note t. AID chooses the split variable and boundary that minimizes the sum of impurities in the two children nodes and the splitting stops when the reduction in impurity is less than a given fraction of the impurity at the root node. For the prediction of y for a sample unit with $X = x$, the first step is to locate the leaf node t that matches x and then use the sample mean $\bar{y}_t$ as a prediction for y. Despite the novelty, the AID method did not attract much attention in the statistics community. [7] showed that AID can severely overfit the data by simulation. [1] criticized AID for ignoring the inherent sampling variability of the data. Classification And Regression Trees (CART, [4]) played an important role in regenerating the interest in tree-based methods. CART follows the same greedy search approach as AID in constructing the binary tree but proposed a different way to control the tree size. While AID adopted an "early" stopping rule, CART firstly constructs an overly grown tree and then prunes it to obtain an optimal subtree with the lowest cross-validation errors. Except for the pruning method, CART proposed to use a series of "surrogate" splits to deal with the missing values and defined an importance score to measure the influence of each independent variable.

On top of the single tree algorithm, there has been much interest in developing the tree-based ensemble approaches, which can mainly be divided into two groups, parallel and sequential ensemble methods. Bagging, Random Forest are the representatives of parallel ensemble methods. Bagging [2] combines a number of un-pruned CART trees constructed from bootstrap samples of the data to reduce the variance. Random Forest [3] weakens the dependence among the CART trees by randomly selecting a subset of variables for split selection at each node. As for sequential methods, Boosting [9] sequentially constructs the classifiers in the ensemble by putting more weight on the observations misclassified in the previous step. Gradient Boosted Decision Trees [11] generalized Boosting to allow optimization of arbitrary differentiable loss functions. XGBoost [5] introduced regularization and built-in parallel processing in the boosting algorithm and greatly improved the accuracy and scalability. Light GBM [13] proposed two novel techniques of gradient-based one-side sampling and exclusive feature bundling, which greatly speeds up the training process while achieving almost the same accuracy.

Note that although the ensemble method can greatly improve the predictive accuracy and robustness, it ruins the interpretable structure of the single-tree method and only outputs a black box for prediction. Therefore, to improve the predictive power and maintain the interpretability meanwhile, our work focuses on improving the single-tree method by replacing the constant leaf model with linear ones, namely, *Segmented Linear Regression Trees* (SLRT). From the parametric view, SLRT can be employed as a recursive change point detect procedure for segmented linear regression (SLR) models, which is much more efficient and flexible than the traditional grid search method. Along this way, we propose to use the conditional Kendall's τ correlation coefficient to select the underlying change points and construct an SLRT for the SLR models. Such recursive partitioning procedure is supported by the theoretical analyses of the conditional Kendall's τ, allows the multivariate split variables, and does not require pre-specifying the number of segments. From the non-parametric view, we propose an efficient greedy splitting method that selects the splits by analyzing the association between residuals and each candidate split variable and only requires fitting the model once in splitting each intermediate node. Further, with the SLRT as a single-tree predictor, we propose a linear random

forest approach that aggregates the SLRTs by a weighted average where the weights depend on the predictive accuracy of each tree predictor. Both simulation and empirical studies showed significant improvements than the CART trees and even the random forest.

This paper is organized as follows. Section 2 introduces the work in the parametric setting, which includes the SLR model assumption, the tree structure construction for the SLR model, and the experimental results. Section 3 introduces the work in the non-parametric setting, which includes the greedy splitting algorithm, the tree pruning procedure for linear regression tree, the weighted linear random forest, and the experimental results. Section 4 concludes with discussions and comparisons about both parametric and non-parametric methods.

2 Recursive Partitioning for Segmented Linear Regression Models

2.1 Segmented Linear Regression Models

Consider the relationship between a univariate response Y and a multivariate explanatory variable $X = (X_1, \ldots, X_p)'$. Let $X^{(r)}$ and $X^{(s)}$ be, respectively, p_r-variate explicit regressors and p_s-dimensional implicit regressors as candidate split variables, which are both pre-given subsets of X. We assume that the regression function $m(X) = E(Y|X)$ is segmented linear over L_0 unknown partitions $\{D_l\}_{l=1}^{L_0}$ in the domain of $X^{(s)}$ such that

$$Y = \sum_{l=1}^{L_0} (\alpha_l + {X^{(r)}}'\beta_l)\mathbb{I}(X^{(s)} \in D_l) + \varepsilon, \tag{2.1}$$

where (α_l, β_l) are regression coefficients over D_l and ε is the random error independent of $\boldsymbol{X}$. In this paper, we consider the case where each partition D_l is axis-aligned in the form of $I_{l_1} \times \cdots I_{l_{p_s}}$ for I_{l_i} being an interval in $\mathbb{R}$ that may also be the entire real line. In practice, the selection of $X^{(r)}$ and $X^{(s)}$ relies on the background knowledge. The categorical variables that denote the change of environment can be included in $X^{(s)}$. For example, the variable *year* in the air pollution prediction problem. The SLR model maintains the simplicity and interpretability of the classical linear models, meanwhile, it is much more flexible and expressive. The core step of the SLR model is to detect the boundary of $\{D_l\}$ that forms the partitions. Existing methods are mainly on the score-based change point detection techniques that detect all the boundaries simultaneously on the Cartesian product space of the split variables. This is expensive in computational cost. Partly due to this, the existing works tend to assume the partitioning variable as univariate or bivariate (e.g., time) and put an upper bound of the number of segments L_0. For instance, [12] and [19] assumed the split variable to be known and univariate. They minimized a modified Bayesian information criterion (BIC) to estimate the number of segments and their boundaries at the same time. It was expensive in computation since the minimizer needs to be searched over the Cartesian product space of many split boundaries. [24] proposed a recursive procedure to estimate the change-points for a univariate split variable with a pre-specified number of segments. [16] selected the change-points via the sum of squared residuals in conjunction with the permutation test, which also assumed a known split variable. [17] studied the asymptotic properties of the estimated regression coefficients for the case of two known partitioning variables and assumed the number of segments was known as well. [23] proposed to assign a 0-1 activeness function to each grid and optimize them through a global

convex loss function. In this section, we will introduce how to identify the segment boundary in a recursive way through tree construction.

Note that as the boundaries of D_l are axis-aligned, (2.1) can be written as

$$Y = \sum_{l=1}^{L_0} \left\{ (\alpha_l + X^{(r)'}\beta_l) \prod_{j_L} \mathbb{I}(X_{j_L}^{(s)} \le a_L) \prod_{j_R} \mathbb{I}(X_{j_R}^{(s)} > a_R) \right\} + \varepsilon, \tag{2.2}$$

and hence can be represented by a linear regression tree including splits $\{X_{j_L}^{(s)} \le a_L, X_{j_R}^{(s)} > a_R\}$ with linear models in the leaf nodes. Each data partition D_l corresponds to a leaf node of the tree and each boundary of D_l corresponds to a binary split in the internal node of the tree. In the following, we will show how to construct a linear regression tree for the SLR models.

2.2 Construct Tree Structure for SLR Models

The construction of tree structures is in essence a recursive partitioning process that outputs a set of data partitions. Starting at the root node with the full data, we aim to identify the underlying change point of the SLR models in each binary split. Algorithm 2.1 describes the generic algorithm of recursive partitioning, where the details of *the stopping rule* and *the split selection algorithm* will be introduced in the following.

Algorithm 2.1 Recursive Partitioning Algorithm

Input: *Training data* $D_{t_0} = \{(X_i, Y_i)\}_{i=1}^n$ *in the root node* t_0.

Output: *Data partitions* $\mathcal{D} = \{D_i\}_{i=1}^L$ *in the leaf nodes.*

1: ***Initialize:*** *No pre-specified partitions,* $\mathcal{D} = \varnothing$;

2: ***if*** *the stopping rule is satisfied over node* t_0 ***then***

3: ***return*** $\mathcal{D} = \mathcal{D} \cup \{D_{t_0}\}$

4: ***else***

5: *Applying the split selection algorithm on node* t_0 *to obtain the split variable* $X_{\hat{j}}^{(s)}$ *and split level* $\hat{a}$.

6: *Let* t_L *and* t_R *be the left and right child-nodes of* t_0, *with data subsets satisfying* $X_{\hat{j}}(i) \le \hat{a}$ *and* $X_{\hat{j}}(i) > \hat{a}$.

7: $t_0 \Leftarrow t_L$ *and repeat Steps* 2–8;
$t_0 \Leftarrow t_R$ *and repeat Steps* 2–8.

8: ***end if***

As the split selection algorithm is recursively applied on each node, and the model specification are segmented linear in all internal nodes if the split boundaries are correctly selected in the preceding partitions. Therefore, it suffices to discuss the split selection in one node, denoted as t. Without prior information on the data partitions, we should (i) justify whether t needs to be split or is a leaf node, and (ii) select the split according to the properties of the underlying boundary when t needs to be split.

To accomplish (i), we firstly fit an ordinary least square (OLS) over the full data and obtain the residuals. Suppose that the underlying model in node t is $Y = \sum_{l=1}^{L}(\alpha_l + \beta_l' \boldsymbol{X}^{(r)})\mathbb{I}(\boldsymbol{X}^{(s)} \in D_l) + \varepsilon$, where ε is independent with the covariates $\boldsymbol{X}$. Let $(\alpha_t^*, \beta_t^*) = \min_{(\alpha,\beta)} \mathrm{E}[\sum_{l=1}^{L_t}(Y - \alpha - \beta' \boldsymbol{X}^{(r)})^2 \mathbb{I}(\boldsymbol{X}^{(s)} \in D_l)]$ and $(\hat{\alpha}_t, \hat{\beta}_t)$ be the OLS estimation. Let $e^* = \varepsilon + \sum_{l=1}^{L_t}[(\alpha_l - \alpha^*) + (\beta_l - \beta^*)' \boldsymbol{X}^{(r)}]\mathbb{I}(\boldsymbol{X}^{(s)} \in D_l)$ and $\hat{e} = Y - \hat{\alpha} - \hat{\beta}' \boldsymbol{X}^{(r)}$ be the OLS residuals. To justify whether the node

t should be split, we test the null hypothesis $H_0 : L_t = 1$ against the alternative hypothesis $H_1 : L_t > 1$. Under the null hypothesis $L_t = 1$, $e^* = \varepsilon$ and hence $e^* \perp \boldsymbol{X}^{(r)}$. Therefore, $\tau(e^*, X_j^{(r)}) = 0$ for any regressor $X_j^{(r)} \in \boldsymbol{X}^{(r)}$, where $\tau(e^*, X_j^{(r)})$ is the population version of Kendall's τ correlation coefficient. In implementation, we use $\widehat{\tau}(\hat{e}, X_j^{(r)})$, the sample version as the test statistic, of which the asymptotic distribution can be obtained by the Hoeffdings theorem on U-statistics.

To accomplish (ii), the split selection is also based on Kendall's τ correlation coefficient between the residuals and the regressor, but conditional on the candidate split. Suppose that the split boundaries over a split variable $X_j^{(s)}$ are $\{a_{j1} \le \cdots \le a_{j,K_j}\}$, then $\tau(X_i^{(r)}, e^* | X_j^{(s)} \le x_j^{(s)})$ keeps constant for any $x_j^{(s)} \le a_{j1}$ when $X_j^{(s)} \notin \boldsymbol{X}^{(r)}$. Therefore, when $X_j^{(s)}$ is not a regressor, we may apply the standard change-point detection algorithm on the sequence of conditional Kendall's τ coefficients to detect the split boundary. When $X_j^{(s)} \in \boldsymbol{X}^{(r)}$, we need to remove the $X_j^{(s)}$-related part from the residuals and then calculate the rank correlation between $X_i^{(r)}$ and the remaining part $\tilde{e} = \hat{e} - e(x_j^{(s)})$.

The above discussions can be summarized and formalized in the following proposition, which provides the theoretical foundation for the stopping rule and the split selection.

Proposition 2.2 *Suppose that the boundaries of $\{D_l\}$ are $\{\{X_j^{(s)} = a_{jk}\}_{k=1}^{K_j}\}_{j=1}^{p_s}$, where $a_{j1} < \cdots < a_{j,K_j}$ are the K_j split boundaries on the $X_j^{(s)}$.*

- *When $L_t = 1$, we have $\sqrt{n_t}(\widehat{\tau}(X_i^{(r)}, \widehat{e}) - \tau(X_i^{(r)}, e^*)) \xrightarrow{d} N(0, \frac{4}{9})$ and $\tau(X_i^{(r)}, e^*) = 0$.*
- *When $L_t > 1$, (i) if $X_j^{(s)} \notin \boldsymbol{X}^{(r)}$, then $\tau(e^*, X_i^{(r)} | X_j^{(s)} \le (>) x)$ keeps constant as x changes when $x \le (>)\ a_{j,K_j}$; (ii) if $X_j^{(s)} \in \boldsymbol{X}^{(r)}$, then $\tau(e^* - e^*(X_j^{(s)}), X_i^{(r)} | X_j^{(s)} \le (>)\ x$ keeps constant as x changes when $x \le (>)\ a_{j,K_j}$, where $e^*(X_j^{(s)}) = \mathrm{E}[e^* | X_j^{(s)}]$.*

Illustrating Examples In the next, we firstly illustrate Proposition 2.2 with the following Example 1 and 2 to explain the differences between the cases when $X_j^{(s)} \notin \boldsymbol{X}^{(r)}$ and $X_j^{(s)} \in \boldsymbol{X}^{(r)}$, and then provide the detailed algorithm for the general cases.

Example 1 We consider the simple case with univariate regressor $\boldsymbol{X}^{(r)} = \{X_1\}$ and one split variable $\boldsymbol{X}^{(s)} = \{X_2\}$. Suppose that the data generating process in the root node t_0 follows

$$Y = X_1 \cdot \mathbb{I}\Big(X_2 \le -\frac{1}{3}\Big) - X_1 \cdot \mathbb{I}\Big(-\frac{1}{3} < X_2 \le \frac{1}{3}\Big) - 2X_1 \cdot \mathbb{I}\Big(X_2 > \frac{1}{3}\Big) + \varepsilon$$

To select the split at node t_0, we fit an OLS model and obtain the residuals $\widehat{e}$ and then calculate the Kendall's correlation coefficients conditional on $X_2 \le a_i$ and $X_2 > a_i$ for $a_i \in \{x_{2i}\}_{i=1}^{n_{t_0}}$, respectively.

$$\widehat{\tau}(\widehat{e}, X_1 | X_2 \le a_i) = \frac{\sum_{i:x_{2i} \le a_i} \mathbb{I}((\widehat{e}_i - \widehat{e}_j)(x_{1i} - x_{1j}) > 0) - \mathbb{I}((\widehat{e}_i - \widehat{e}_j)(x_{1i} - x_{1j}) < 0)}{n_L(a_i)(n_L(a_i) - 1)/2};$$

$$\widehat{\tau}(\widehat{e}, X_1 | X_2 > a_i) = \frac{\sum_{i:x_{2i} > a_i} \mathbb{I}((\widehat{e}_i - \widehat{e}_j)(x_{1i} - x_{1j}) > 0) - \mathbb{I}((\widehat{e}_i - \widehat{e}_j)(x_{1i} - x_{1j}) < 0)}{n_R(a_i)(n_R(a_i) - 1)/2},$$

where $n_L(a_i) := \#\{i \,|\, x_{2i} \le a_i\}$ and $n_R(a_i) := \#\{i \,|\, x_{2i} > a_i\}$ are the sub-sample size on the left and right side. Figure 1 shows that $\widehat{\tau}(\widehat{e}, X_1 | X_2 \le a_i)$ kept approximately constant when $a_i \le -\frac{1}{3}$ and started to change when a_i surpassed $-\frac{1}{3}$. Similarly, Figure 2 shows that $\widehat{\tau}(\widehat{e}, X_1 | X_2 > a_i)$ stayed at the same level when $a_i > \frac{1}{3}$ and changed with a_i when $a_i \le \frac{1}{3}$.

The above observations are consistent with the statement (i) in Proposition 2.2, which suggests that we may identify the leftmost split boundary by detecting the change of $\widehat{\tau}(\widehat{e}, X_1|X_2 \leq a_i)$ in mean or variance from the left side, and identify the rightmost split boundary by checking $\widehat{\tau}(\widehat{e}, X_1|X_2 > a_i)$ from the right side.

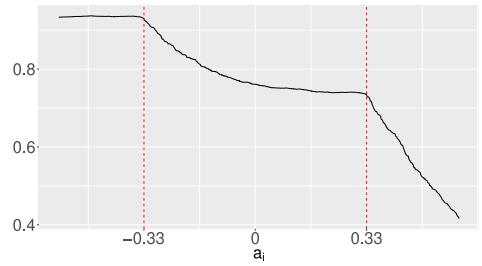

Figure 1 $\widehat{\tau}(\widehat{e}, X_1|X_2 \leq a_i)$

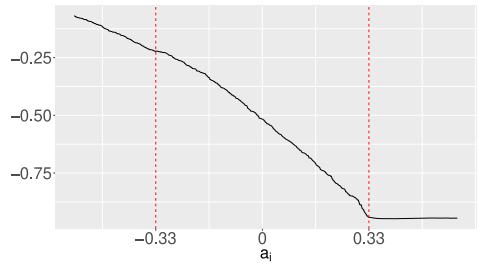

Figure 2 $\widehat{\tau}(\widehat{e}, X_1|X_2 > a_i)$

Figure. The sample version of conditional Kendall's τ coefficients between X_1 and $\hat{e}$ conditional on $X_2 \leq a_i$ and $X_2 > a_i$, respectively

Example 2 Now we consider the case where the split variable X_2 is also a regressor and explains why it is necessary to replace the residuals $\widehat{e}$ with $\widehat{e} - \widehat{e}(X_2)$.

$$Y = (X_1 + 2X_2) \cdot \mathbb{I}\left(X_2 \leq -\frac{1}{3}\right) + (-X_1 - 2X_2) \cdot \mathbb{I}\left(-\frac{1}{3} < X_2 \leq \frac{1}{3}\right)$$
$$+ (-2X_1 + X_2) \cdot \mathbb{I}\left(X_2 > \frac{1}{3}\right) + \varepsilon$$

In Example 2, we also fit an OLS model to obtain the residuals $\widehat{e}$. However, the conditional Kendall's τ coefficients $\hat{\tau}(\hat{e}, X_1|X_2 \leq a_i)$ do not have the same pattern as Example 1. As is shown in Figure 3, there is a significant trend in $\{\widehat{\tau}(\widehat{e}, X_1|X_2 \leq a_i)\}$ even when $a_i \leq -\frac{1}{3}$. Therefore, we can not identify the split boundary $X_2 = -\frac{1}{3}$ by the sequential change point detection. Let $\widehat{e}(X_2)$ be the kernel regression function obtained by regressing $\widehat{e}$ on X_2. Figure 4 shows the results by replacing $\widehat{e}$ with $\widehat{e} - \widehat{e}(X_2)$, which recovers the property of keeping constant for $a_i \leq -\frac{1}{3}$ and hence can be used to identify the split boundary at $X_2 = -\frac{1}{3}$.

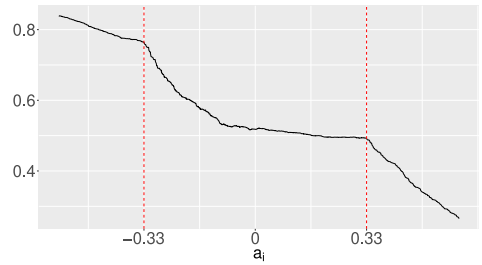

Figure 3 $\widehat{\tau}(\widehat{e}, X_1|X_2 \leq a_i)$

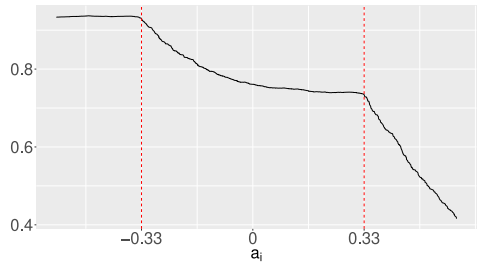

Figure 4 $\widehat{\tau}(\widehat{e} - \widehat{e}(X_2), X_1|X_2 > a_i)$

Figure. The sample version of conditional Kendall's τ $\widehat{\tau}(\widehat{e}, X_1|X_2 \leq a_i)$ and $\widehat{\tau}(\widehat{e} - \widehat{e}(X_2), X_1|X_2 > a_i)$

Now let us explain the above observations by checking the formula of $\tau(\widehat{e}, X_1|X_2 \le a)$. As $\widehat{e}$ is also segmented linear in $\boldsymbol{X}$ with splits at $X_2 = -\frac{1}{3}$ and $X_2 = \frac{1}{3}$. Denote

$$e = [b_{1L}X_1 + b_{2L}X_2] \cdot \mathbb{I}\left(X_2 \le -\frac{1}{3}\right) + [b_{1M}X_1 + b_{2M}X_2] \cdot I\left(-\frac{1}{3} < X_2 \le \frac{1}{3}\right) + [b_{1R}X_1 + b_{2R}X_2] \cdot I\left(X_2 > \frac{1}{3}\right) + \varepsilon.$$

Then when $a \le -\frac{1}{3}$,

$$\begin{aligned}\tau(\widehat{e}, X_1 \,|\, X_2 \le a) &= 2\mathrm{P}((\widehat{e} - \widehat{e}')(X_1 - X_1') > 0 \,|\, X_2 \le a, X_2' \le a) - 1 \\ &= 2\mathrm{P}[(b_{2L}(X_2 - X_2') + (\varepsilon - \varepsilon'))(X_1 - X_1') > -b_{1L}(X_1 - X_1')^2 \,|\, X_2 \le a, X_2' \le a] - 1 \\ &= 2\mathrm{P}[b_{2L}(X_2 - X_2') + (\varepsilon - \varepsilon') > -b_{1L}|X_1 - X_1'| \,|\, X_2 \le a, X_2' \le a] - 1 \\ &= 2\mathrm{P}[(\varepsilon' - \varepsilon) - b_{1L}|X_1 - X_1'| < b_{2L}(X_2 - X_2') \,|\, X_2 \le a, X_2' \le a] - 1.\end{aligned}$$

Therefore, as the distribution of $b_{2L}(X_2 - X_2')|(X_2 \le a, X_2' \le a)$ depends on the split boundary a, $\tau(\widehat{e}, X_1|X_2 \le a)$ does not keep constant even when $a \le -\frac{1}{3}$. Note the fundamental reason is that $\widehat{e}$ involves the term about X_2, so we would like to subtract the term related to X_2 and use the remaining part $\widehat{e} - \widehat{e}(X_2)$ to calculate the Kendall's τ coefficients. As is shown in Figure 4, $\{\widehat{\tau}(\widehat{e} - \widehat{e}(X_2), X_1|X_2 > a_i)\}$ are approximately constant when $a_i \le -\frac{1}{3}$ and hence can be used to identify the split boundary, which is consistent with the statement (ii) in Proposition 2.2. Although the graphical pattern is quite significant as shown in Figures 3 and 4, a formal method is required to select the split boundary from data, which will be introduced in the following.

Without loss of generality, we consider identifying the leftmost split boundary on $X_j^{(s)}$ from the conditional Kendall's τ given $X_j^{(s)}$. The data sequence is $\{\widehat{\tau}(\widetilde{e}, X_k^{(r)}|X_j^{(s)} \le a_1), \ldots, \widehat{\tau}(\widetilde{e}, X_k^{(r)}|X_j^{(s)} \le a_n)\}$, where $a_i = x_{ji}^{(s)}$ is the i-th largest sample point of $X_j^{(s)}$, $X_k^{(r)} \in \boldsymbol{X}^{(r)}$ is a regressor, and $\widetilde{e} = \widehat{e}$ if $X_j^{(s)} \notin \boldsymbol{X}^{(r)}$ and $\widetilde{e} = \widehat{e} - \widehat{e}(X_j^{(s)})$ if $X_j^{(s)} \in \boldsymbol{X}^{(r)}$.

Let a^* be the underlying leftmost split boundary on $X_j^{(s)}$, then $\mathrm{E}[\widehat{\tau}(\widetilde{e}, X_2|X_2 < a_i)]$ keeps constant for $a_i \le a^*$ and starts to change for $a_i > a^*$. Since $\{\tau(\widetilde{e}, X_2|X_2 > a_i), \ldots\ldots, \tau(\widetilde{e}, X_2|X_2 > a_n)|a_i > a^*\}$ can not be regarded as the sample points from a super-population, any i satisfying $a_i > a^*$ is a change point. Therefore, it is inappropriate to directly apply the changepoint detection method on the whole sequence. However, we may identify a^* by using the changepoint detection method sequentially from the left by testing whether there exists change in mean inside the sequence $\{\widehat{\tau}(\widetilde{e}, X_k^{(r)}|X_j^{(s)} \le a_1), \ldots, \widehat{\tau}(\widetilde{e}, X_k^{(r)}|X_j^{(s)} \le a_i)\}$ for $i = n_0, n_0 + 1, \ldots, n$, where n_0 is a hyper-parameter that specifies the minimal sample size that is allowed in a leaf node.

Algorithm 2.3 Identify a^* by the sequential changepoint detection

1: ***for*** $i = n_0, n_0 + 1 \ldots\ldots, n$ ***do***
2: *Test the following hypothesis using the general likelihood-ratio based approach:*
3: H_0 : *there exists a change point* $\tau \in \{1, \ldots, i\}$ *for* $(Z_1, \ldots, Z_i)$ *where* $Z_i := \widehat{\tau}(\widetilde{e}, X_2|X_2 \le a_i)$.
4: *Let* P_i *be the p-value and* $\alpha_i = 1 - P_i$.
5: ***end for***
6: ***for*** i *in* $n_0, n_0 + 1, \ldots\ldots, n$ ***do***
7: ***if*** *for all* $k \ge i$, $\alpha_k > 0.95$ ***then***
8: *let* $\widehat{a}^* = a_i$, *break.*

9: ***end if***

10: ***end for***

Algorithm 2.3 includes two parts. Part 1 (Line 1 to 5) calculates the corresponding p-value for each sample point a_i, where we adopt the likelihood-based single-changepoint detection method for the hypothesis testing [15]. Part 2 (Line 6 to 10) selects the sample point $\hat{a}^*$ such that there are change points in $\{\widehat{\tau}(\widetilde{e}, X_k^{(r)} \,|\, X_j^{(s)} \le a_1), \ldots, \widehat{\tau}(\widetilde{e}, X_k^{(r)} \,|\, X_j^{(s)} \le a_i)\}$ in all $a_i \ge \hat{a}^*$. Note that we do not select $\hat{a}^*$ as the sample point that leads to the first existence of changepoint but the one that leads to the continued existence. This is to avoid the influence of unexceptional values, which will be shown in the simulation example in Section 2.3.

The Complete Recursive Partitioning Algorithm Algorithm 2.3 provides the procedures to identify the leftmost split boundary for a given split variable. Suppose that $X_1^{(s)} = \widehat{a}^*$ is selected in the root node t_0, we partition the data into two subsets $D_{t_L} = \{(\boldsymbol{x}_i, y_i) | x_{1i}^{(s)} \le \widehat{a}^*\}$ and $D_{t_R} = \{(\boldsymbol{x}_i, y_i) | x_{1i}^{(s)} > \widehat{a}^*\}$. Then we continue to identify the split boundary on D_{t_L} and D_{t_R}, respectively. Generally speaking, to construct the complete tree structure, it suffices to apply Algorithm 2.3 recursively on the full data. To identify all the split boundaries, we firstly visit the all split variables one after another (Line 2 and 20) and update the set of partitions after the leftmost boundary is selected (Line 11 and 30). The detailed procedures are shown in the following Algorithm 2.4, where the overall recursive partitioning procedure firstly deal with the split levels on the non-overlapping split variables (Lines 1–18) and then the overlapping part due to the different properties (Lines 19–37). We adopt the false discovery rate (FDR) controlling procedure to deal with the issue of multiplicity in repeated testing in Line 7 and Line 25, where $c(p_r) = \sum_{j=1}^{p_r} \frac{1}{j}$.

Algorithm 2.4 Detect the Split levels on $\boldsymbol{X}^{(s)}$

Input: *Training data:* $D = \{(X(i), Y(i))\}_{i=1}^n$, $\mathcal{D} = \{D\}$;

Output: *Splits levels:* $\{X_{j_k}^{(s)} = a_k\}$ *partitions* $\mathcal{D} = \{\widehat{D}_l\}$

Part 1 *:Detect the split levels on the non-overlapping split variable* $X_j^{(s)}$ *in* $\boldsymbol{X}^{(s)} \setminus \boldsymbol{X}^{(r)}$

1: ***for*** D_0 *in* $\mathcal{D}$ ***do***

2: ***for*** $X_j^{(s)}$ *in* $\boldsymbol{X}^{(s)} \setminus \boldsymbol{X}^{(r)}$ ***do***

3: ***while*** *TRUE* ***do***

4: *Regress* Y *on* $\boldsymbol{X}^{(r)}$ *using OLS on dataset* D_0 *and obtain the residuals* $\widehat{e}$;

5: *Compute the test statistics* $\{\widehat{\tau}(X_k^{(r)}, \widehat{e})\}_{k=1}^{p_r}$ *and obtain the corresponding p-value* P_k *to test the null hypothesis* $H_0 : L_{D_0} = 1$.

6: *List the p-values in ascending order and denote them by* $P_{(1)}, \ldots, P_{(p^r)}$.

7: ***if*** *there exists* k *such that* $P_{(k)} \le \frac{k}{p_r c(p_r)} \alpha$ ***then***

8: *Compute the conditional Kendall's* τ *coefficients* $\{\widehat{\tau}(X_k^{(r)}, \widehat{e} | X_j^{(s)} \le a) | a = x_{ji}^{(s)}\}$;

9: *Apply Algorithm* 2.3 *to obtain the split boundary* a^*.

10: *Let* $D_L = \{(\boldsymbol{x}(i), Y(i)) | x_j^{(s)}(i) \le a^*\}$, $D_R = \{(\boldsymbol{x}(i), Y(i)) | x_j^{(s)}(i) > a^*\}$.

11: *Let* $\mathcal{D} = \mathcal{D} \setminus \{D_0\} \cup \{D_L\}$ *and* $D_0 = D_R$

12: ***else***

13: *Let* $\mathcal{D} = \mathcal{D} \cup \mathcal{D}_0$

14: ***break*** (*terminate the split detection process on* $X_j^{(s)}$ *on* D_0)

end if
end while
end for
end for

Part 2*: Detect the split levels on the overlapping split variable* $X_j^{(s)}$ *in* $\boldsymbol{X}^{(s)} \cap \boldsymbol{X}^{(r)}$

for D_0 *in* $\mathcal{D}$ ***do***
  for $X_j^{(s)}$ *in* $\boldsymbol{X}^{(s)} \cap \boldsymbol{X}^{(r)}$ ***do***
    while *TRUE* ***do***
      Regress Y *on* $\boldsymbol{X}^{(r)}$ *using OLS on dataset* D_0 *and obtain the residuals* $\widehat{e}$*;*
      Compute the test statistics $\{\widehat{\tau}(X_k^{(r)}, \widehat{e})\}_{k=1}^{p^r}$ *and obtain the corresponding p-value* P_k *to test the null hypothesis* $H_0 : L_{D_0} = 1$.
      List the p-values in ascending order and denote them by $P_{(1)}, \ldots, P_{(p^r)}$.
      if *there exists* k *such that* $P_{(k)} \leq \frac{k}{p_r c(p_r)}\alpha$ ***then***
        Regress $\widehat{e}$ *on* $X_j^{(s)}$ *by kernel smoothing and let* $\widetilde{e} = \widehat{e} - \widehat{e}(X_j^{(s)})$;
        Compute the conditional Kendall's τ *coefficients* $\{\widehat{\tau}(X_k^{(r)}, \widetilde{e}|X_j^{(s)} \leq a)|a = x_{ji}^{(s)}\}$;
        Apply Algorithm 2.3 *to obtain the split boundary* a^*.
        Let $D_L = \{(\boldsymbol{x}(i), Y(i))|x_j^{(s)}(i) \leq a^*\}$, $D_R = \{(\boldsymbol{x}(i), Y(i))|x_j^{(s)}(i) > a^*\}$.
        Let $\mathcal{D} = \mathcal{D} \setminus \{D_0\} \cup \{D_L\}$ *and* $D_0 = D_R$
      else
        Let $\mathcal{D} = \mathcal{D} \cup \mathcal{D}_0$
        break (*terminate the split detection process on* $X_j^{(s)}$ *on* D_0)
      end if
    end while
  end for
end for

2.3 Experimental Results

To verify the proposed method, consider the following example with regressors $\boldsymbol{X}^{(r)} = (X_1, X_2, X_3)$, split variables $\boldsymbol{X}^{(s)} = (X_1, X_2, X_4)$, and segmented linear regression function

$$\begin{aligned} m(\boldsymbol{X}) = {} & 3X_2\mathbb{I}(X_2 > 15) - 3X_1\mathbb{I}(X_2 \leq 15) + X_2\mathbb{I}(X_1 > 10) \\ & - X_2\mathbb{I}(X_1 \leq 10) + X_3\mathbb{I}(X_4 \leq 10) - 3X_3\mathbb{I}(X_4 > 10). \end{aligned} \tag{2.3}$$

There are 3 underlying splits in $m(\boldsymbol{X})$, $\{X_1 = 10, X_2 = 15, X_4 = 10\}$, which form 8 segments. The covariates were generated from $X_1 \sim U(0, 20)$, $X_2 \sim U(0, 25)$, $X_3 \sim U(0, 10)$, and $X_4 \sim U(0, 20)$. The response variable $Y = m(\boldsymbol{X}) + \varepsilon$ with $\varepsilon \sim N(0, 1)$.

According to Algorithm 2.4, we firstly consider to identify the split boundary of $X_4 \in \boldsymbol{X}^{(s)} \setminus \boldsymbol{X}^{(r)}$ in the root node t with sample size n_t. Suppose $\{x_{41} \leq \cdots \leq x_{4,n_t}\}$ are the sample points of X_4 in the ascending order. We select the split level on X_4 from its sample points. Figure 5 shows the Kendall's τ coefficients between $\widehat{e}$ and X_3 conditional on $X_4 \leq a$, where a was taken from $\{x_{4i}\}_{i=1}^{n_t}$ in the root node t. Figure 6 demonstrates the p-values for testing the null hypothesis H_0: *there is no changepoint within* $\{\widehat{\tau}(X_3, \hat{e}|X_4 \leq x_{41}), \ldots, \widehat{\tau}(X_3, \hat{e}|X_4 \leq a)\}$ for $a = x_{4,n_0+1}, \ldots, x_{4,n_t-n_0}$. It can be seen that when $a \geq 10.24$, the confidence levels are

all above 0.95, which suggests that there exists at least one changepoint in $\{\widehat{\tau}(X_3, \hat{e}|X_4 \leq x_{41}), \ldots, \hat{\tau}(X_3, \widehat{e}|X_4 \leq a)\}$ for all $a \geq 10.24$. So $X_4 = 10.24$ is selected as the split in root node, which is close to the underlying split $X_4 = 10$.

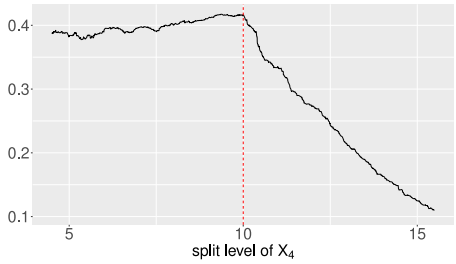

Figure 5 $\widehat{\tau}(\hat{e}, X_3|X_4 \leq a)$ for split levels $a \in \{x_{4i}\}$

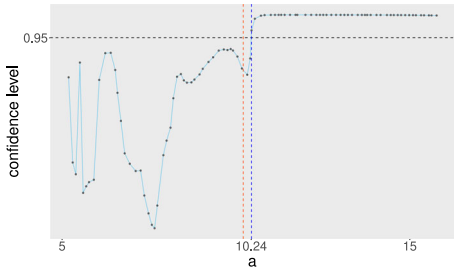

Figure 6 Confidence levels of changepoint detection

Figure. The split selection on X_4 in root node t

After the split selection in the root node, the dataset is partitioned into two child-nodes, t_L and t_R, which includes the data subsets $D_{t_L} := \{(\boldsymbol{x}_i, y_i)|x_{4i} \leq 10.24\}$ and $D_{t_R} := \{(\boldsymbol{x}_i, y_i)|x_{4i} > 10.24\}$, respectively. As the results were similar between the two branches, we only demonstrate the results in the left child-node t_L, and the counterpart in t_R can be found in the Appendix.

In the child-node t_L, we consider to identify the split boundary on X_2. Same as the root node, we also firstly fit an OLS model on data D_{t_L} and obtain the residuals $\widehat{e}$. Compared with the split selection of X_4 in the root node t, the extra step in t_L is to fit a non-parametric kernel regression between X_2 and $\widehat{e}$ and then remove the trend $\widehat{e}(X_2)$ from the residuals $\widehat{e}$. Figure 7 shows that there was significant trend in the scatter plot between OLS residuals $\widehat{e}$ and the split variable X_2, where the red line is the kernel fitting result. Figure 8 demonstrates the scatter plot between X_2 and $\widehat{e} - \widehat{e}(X_2)$, no significant trend was observed. Let $\{x_{2i}\}_{i=1}^{n_{t_L}}$ denote sample points of X_2 in node t_L sorted in the ascending order. We calculate the conditional Kendall's τ coefficients between the regressor X_1 and $\widehat{e} - \widehat{e}(X_2)$ conditional on $X_2 \leq a$ for $a \in \{x_{2i}\}_{i=1}^{n_{t_L}}$. The results are summarized in Figure 9, where $\{\widehat{\tau}(\hat{e}, X_1|X_2 \leq a)\}$ were approximately constant for $a \leq 15$. Figure 10 shows the confidence levels of the sequential changepoint detection, which suggests that $X_2 = 15.03$ was selected, which is near the underlying split $X_2 = 15$.

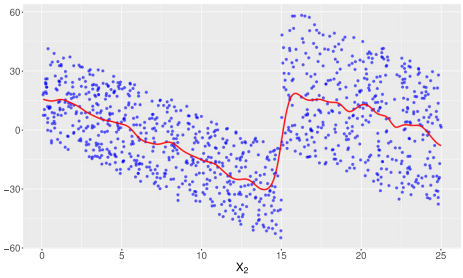

Figure 7 Scatter plot between X_2 and $\{\hat{e}, \hat{e}(X_2)\}$

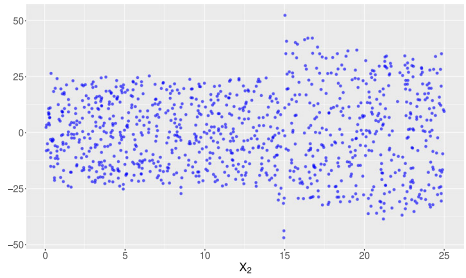

Figure 8 Scatter plot between X_2 and $\hat{e} - \hat{e}(X_2)$

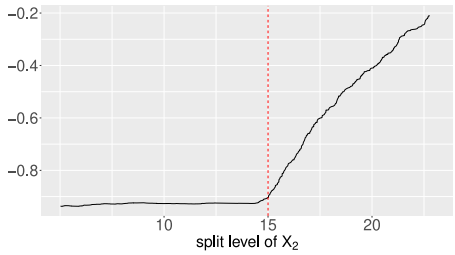

Figure 9 $\widehat{\tau}(\hat{e} - \hat{e}(X_2), X_1 | X_2 \le a)$ for split levels of X_2

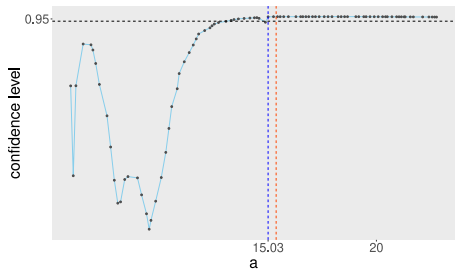

Figure 10 Confidence levels of changepoint detection

Similar to the data partition in the root node, the dataset in t_L was split into two parts t_{LL} and t_{LR}, which contains the data subset $D_{t_{LL}} := \{(\boldsymbol{x}_i, y_i) | x_{4i} \le 10.24 \text{ and } x_{2i} \le 15.03\}$ and $D_{t_{LR}} := \{(\boldsymbol{x}_i, y_i) | x_{4i} \le 10.24 \text{ and } x_{2i} > 15.03\}$, respectively. We demonstrate the results of the split selection in t_{LL}. In node t_{LL}, the task is to identify the split level on X_1 using the conditional Kendall's τ between the residuals and X_2. As X_1 is also a regressor, we need to remove the X_1-related term $\widehat{e}(X_1)$ from the residuals, which is exactly the same as the procedures in X_2. Figure 11 is the scatter plot between X_1 and the residuals $\widehat{e}$ and the red line corresponds to the kernel regression function $\widehat{e}(X_1)$. Figure 12 shows the result after removing $\widehat{e}(X_1)$ from $\widehat{e}$. Figure 13 shows that when $a \le 10$, the conditional Kendall's τ coefficients $\widehat{\tau}(\widehat{e} - \widehat{e}(X_1))$ were approximately constant in a and started to change when $a > 10$. Figure 14 shows that by applying Algorithm 2.3 on the sequence of Kendall's τ coefficients, $X_1 = 9.83$ was selected as the split level, which is around the underlying split $X_1 = 10$.

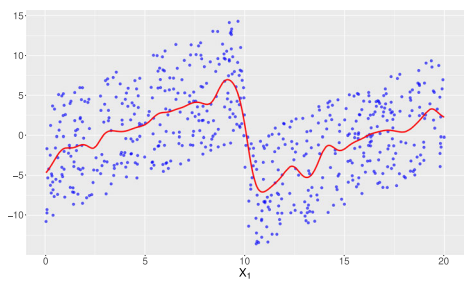

Figure 11 Scatter plot between X_1 and $\hat{e}$

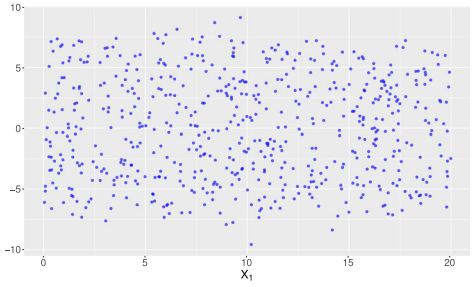

Figure 12 Scatter plot between X_1 and $\hat{e} - \hat{e}(X_1)$

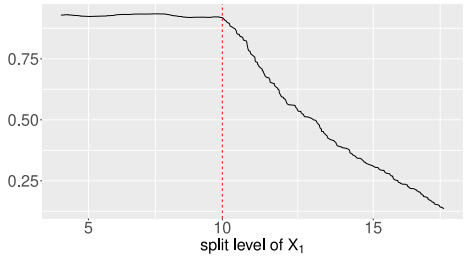

Figure 13 $\widehat{\tau}(\hat{e} - \hat{e}(X_1), X_2 | X_1 \le a)$

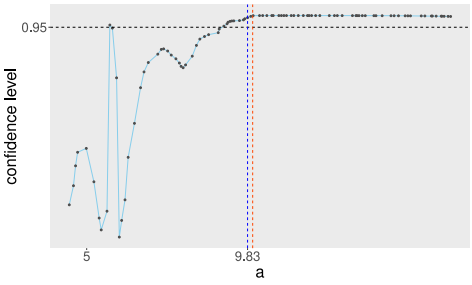

Figure 14 Confidence levels of changepoint detection

Note that here we only demonstrate the detailed results for the three nodes, $\{t, t_L, t_{LL}\}$. To construct the complete tree structure, similar procedures were also implemented on the nodes $\{t_R, t_{LR}, t_{RL}, t_{RR}\}$, of which the corresponding details can be found in the appendix. The binary tree in Figure 15 summarizes the identified split boundaries, which are all close to the underlying split $\{X_4 = 10, X_2 = 15, X_1 = 10\}$. Each leaf node contains a data subset that corresponds to a segment in the model. To obtain the estimated the segmented linear regression model $\widehat{m}(\boldsymbol{x})$, it suffices to implement the OLS fitting within each leaf node respectively. Therefore, it is a linear regression tree with a linear model in each leaf node from the perspective of tree-based modeling. At the same time, it provides a flexible estimating strategy for the segmented linear model that does not require a pre-specified number of the segments.

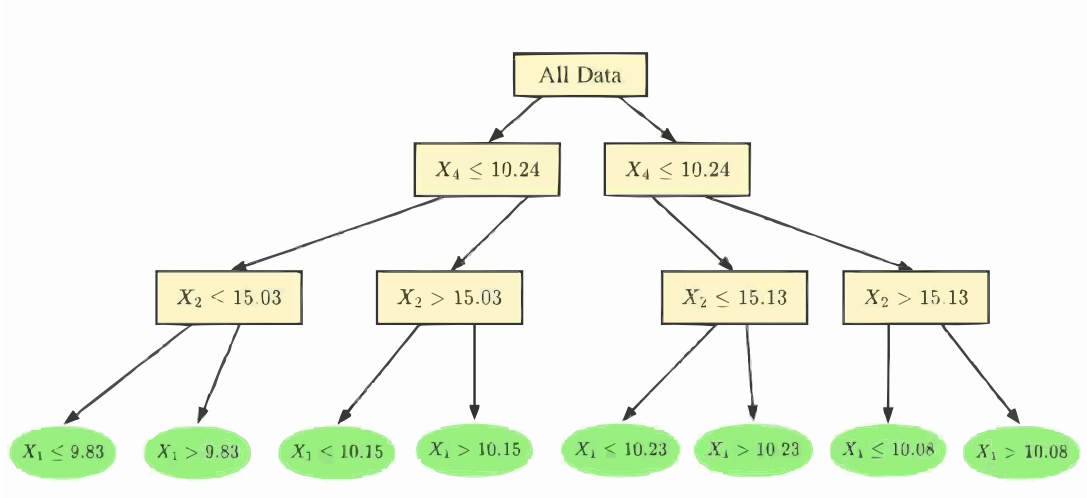

Figure 15 The binary tree constructed for the segmented linear model

3 Greedy Segmented Linear Regression Trees and Linear Random Forests

3.1 Segmented Linear Regression Trees' Greedy Partitioning Algorithm

Although Algorithm 2.4 provides a practical solution for constructing the linear regression tree for the SLR models, the algorithm heavily relies on the model specification. If the underlying regression function $\mathrm{E}[Y|\boldsymbol{x}]$ is not segmented linear in $\boldsymbol{x}$, then the properties (i) and (ii) in Proposition 2.2 will not hold and hence there may not be any candidate split that would satisfy Algorithm 2.3.

Therefore, in this section, we will provide another greedy algorithm that is also based on the Kendall's τ coefficients and selects the split in an intuitive way. Specifically, we select the split that can maximally extract the rank correlation contained between the residuals and the covariates $\boldsymbol{x}$, making the data subsets more adapted to linear models. Consider the split selection over the root node t_0. Let D_{t_0} be the dataset at node t_0 including n independent observations generated from Model (2.1), denoted by $D_{t_0} = \{X(i), Y(i)\}_{i=1}^n$, where $X(i) = (X_1(i), \ldots, X_p(i))$. We are to partition D_{t_0} into two halves to make the regression functions over the subsets more compliant to linear models. Let $\hat{Y} = \hat{\alpha}_{t_0} + X^{(r)'}\hat{\beta}_{t_0}$ be the fitted ordinary least square (OLS) regression over D_{t_0}, and $\hat{e} = Y - \hat{Y}$ be the estimated residuals. If $\mathrm{E}(Y|X^{(r)})$

is nonlinear, the non-linearity will get reflected in the residuals $\hat{e}$ and their dependence with the regressors. Indeed, if $m(X)$ is piecewise linear, $\hat{e}$ is also piecewise linear in X since

$$\hat{e} = \sum_{l=1}^{L}((\alpha_l - \hat{\alpha}_{t_0}) + X^{(r)'}(\beta_l - \hat{\beta}_{t_0}))\mathbb{I}(X^{(s)} \in D_l) + \varepsilon. \tag{3.1}$$

Thus, conditional on each partition D_l, $\hat{e}$ is linear in the explicit regressors $X^{(r)}$. For a regressor $X_k^{(r)}$ with a coefficient $(\beta_l(k) - \hat{\beta}_{t_0}(k)) \neq 0$ in (3.1), $X_k^{(r)}$ and $\hat{e}$ will be rank correlated, concordant (discordant) for positive (negative) coefficient.

We employ Kendall's τ rank correlation coefficient [14] to capture the dependence. Define the following conditional Kendall's τ statistic over the left child node t_L with dataset $D_{t_L} = \{(X(i), Y(i))|X_j^{(s)}(i) \leq a\}$,

$$\hat{\tau}(X_k^{(r)}, \hat{e}|X_j^{(s)} \leq a) = \frac{\sum_{i<i' \atop i,i' \in I_{t_L}} \operatorname{sgn}(X_k^{(r)}(i) - \hat{e}(i))\operatorname{sgn}(X_k^{(r)}(i') - \hat{e}(i'))}{N_{t_L}(j,a)(N_{t_L}(j,a) - 1)/2},$$

where $X_j^{(s)}(i)$ is the i-th sample point of the j-th variable $X_j^{(s)}$, $I_{t_L}(j,a) = \{i \ |X_j^{(s)}(i) \leq a, 1 \leq i \leq n\}$ is the set of indices for the dataset D_{t_L} and $N_{t_L}(j,a) = |I_{t_L}(j,a)|$ is the subsample size. Similar quantities $\hat{\tau}(X_k^{(s)}, \hat{e}|X_j^{(s)} > a)$, $I_{t_R}(j,a)$ and $N_{t_R}(j,a)$ are defined in the same way on the right child node t_R. Then we define the criterion function

$$\mathcal{C}(j,a) = \sum_{k=1}^{p_r}\{|\hat{\tau}(X_k^{(r)}, \hat{e}|X_j^{(s)} \leq a)| + |\hat{\tau}(X_k^{(r)}, \hat{e}|X_j^{(s)} > a)|\}, \tag{3.2}$$

for $1 \leq j \leq p_s$ and $a \in \{X_j^{(s)}(i)|(X(i), Y(i)) \in D_{t_0}\}$. Note that we only need to consider a among the realized sample values $\{X_j^{(s)}(i)\}_{i=1}^n$ because other choices lead to duplicated partitioning results. Transforming continuous ranges into discrete candidate split levels largely reduces the computation. As we expect to further exploit the information remained in the residuals $\widehat{e}$ in the child-nodes t_L and t_R by linear regressions, we select the split variable and level (j^*, a^*) by maximizing the criterion function $\mathcal{C}(j,a)$. That is, we construct the tree by selecting the split to maximize the conditional rank correlations during each split in a greedy way, which does not guarantee a globally optimal solution. Algorithm 3.1 shows the detailed procedures.

Algorithm 3.1 Greedy Split Selection Algorithm

Input: *Training data at node t:* $D_t = \{(X(i), Y(i))\}_{i=1}^{n_t}$.

Output: *Split variable index and split level* $(\hat{j}, \hat{a})$

1: *Fit the OLS linear regression of* Y *on* $X^{(r)}$ *with data* D_t *and obtain the estimated residuals* $\hat{e}$.

2: $C_{D_t} = \{(j,a)|1 \leq j \leq p_s, a \in \{X_j^{(s)}(i)|(X(i), Y(i)) \in D_t\}, \min\{N_{t_L}(j,a), N_{t_R}(j,a)\} > N_{\min}\}$.

3: $\mathcal{C}_{\max} = 0$

4: ***for*** (j,a) *in* C_{D_t} ***do***

5: *using* $\hat{e}$ *and data* D_t, *obtain* $\mathcal{C}(j,a)$ *given in* (3.2).

6: ***if*** $\mathcal{C}(j,a) > \mathcal{C}_{\max}$ ***then***

7: $(\hat{j}, \hat{a}) = (j,a)$

8: ***end if***

9: ***end for***

The proposed split selection algorithm 3.1 is similar to GUIDE [20] in that GUIDE also utilizes information based on $\hat{e}$. However, GUIDE uses the signs of $\hat{e}$, which would be less informative than using the rank of $\hat{e}$. Besides, GUIDE considers the marginal association between the signs of residuals and the regressors via the χ^2-test instead of the conditional association, which may lead to misidentified split variables even in the two-segmented case as shown in the following example (3.3). Also, GUIDE uses $\hat{e}$ just in selecting the split variable, while we select the split variable and level simultaneously in Algorithm 2.1.

In the following, we explain the intuition and motivation of Algorithm 3.1 by analyzing the OLS residuals in a two-segmented case. Consider the following example where the mean regression function consists of three covariates and one split point,

$$Y = \begin{cases} X_1 - X_2 + 2 + \varepsilon & \text{if } X_3 > 5, \\ -X_1 + 2X_2 + \varepsilon & \text{if } X_3 \le 5, \end{cases} \tag{3.3}$$

where $X_1, X_2, X_3 \overset{\text{i.i.d.}}{\sim} U(0, 10)$ and $\varepsilon \sim N(0, 0.01)$. Let $\hat{Y}$ be the OLS fitted value and $\hat{e} = Y - \hat{Y}$ be the estimated residuals. Figure 16 shows the association between regressors X_1, X_2 and $\hat{e}$ conditional on subspaces split by the correct split $X_3 = 5$, while Figure 17 corresponds to a false split $X_2 = 5$. In Figure 16 for the correct split, the variation of $\hat{e}$ tends to be discordant with X_1 and accordant with X_2. The rank correlations between each regressor and $\hat{e}$ are both significant conditional on the correct split. In Figure 17 for a false split, there's no uniform trend between X_1, X_2 and $\hat{e}$. With the increase of $\hat{e}$, part of the sample points X_1 form an increasing trend while the other tend to decrease.

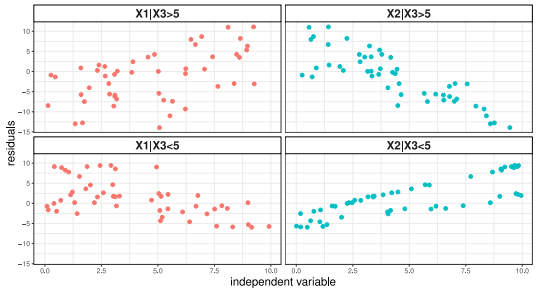

Figure 16 Conditional on Correct Split $X_3 = 5$

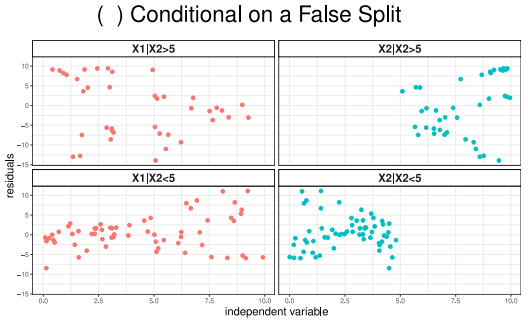

Figure 17 Conditional on a False Split $X_2 = 5$

While Figures 16 and 17 provides a visualized impression that the rank correlation between $\hat{e}$ and regressors X_1, X_2 are stronger conditional on the correct split than a false split, we further calculate the value of $\mathcal{C}(j, a)$ to validate our algorithm. In this example, $\mathcal{C}(j, a) = \sum_{k=1}^{2} \{|\hat{\tau}(X_k, \hat{e}|X_j \le a)| + |\hat{\tau}(X_k, \hat{e}|X_j > a)|\}$, where $j \in \{1, 2, 3\}$ and $a \in \{X_j(i)\}_{i=1}^{n=100}$.

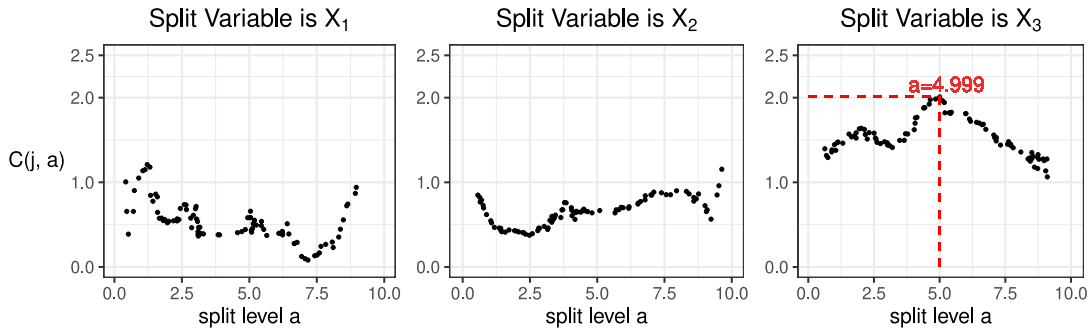

Figure 18 Criterion function based on Kendall's τ coefficient

Figure 18 shows the value of $\mathcal{C}(j, a)$ for each candidate split level $X_j = a$, where we exclude the split levels that result in subsets with sample size less then 10. The maximum is obtained at the true split $X_3 = 5$, which means we will correctly choose $X_3 = 5$ as the estimated split level. where the rank correlations between X_1, X_2 and $\hat{e}$ are quite significant.

Our proposed Algorithm 3.1 is initially enlightened by GUIDE [20, 21], for GUIDE also analyzes globally estimated residuals for the selection of splitting variable. But GUIDE fails in the above example, because it focuses on the marginal association between residuals and each covariate instead of conditional association. More specifically, GUIDE detects non-random patterns of covariates between two groups of positive residuals and negative residuals via χ^2 test. The p-values for X_1, X_2, X_3 from the χ^2 independence test were $0.0897, 0.0566, 0.0657$ respectively, where X_2 shows the most significant association with signs of residuals. According to GUIDE, X_2 was selected as the splitting variable, contrary to the ground truth.

3.2 Adaptive Tree Pruning Procedure

The construction of regression trees also requires a proper way to control the tree size. A right-sized tree is crucial for capturing the main structure and preventing over-fitting. Here we adopt the same way as CART that uses a simple stopping rule to let the splitting process continue in a greedy way and generate a sufficiently large tree, followed by a tree pruning procedure to obtain a right-sized tree.

The simple stopping rule below has two parameters $N_{\min}$ and $\mathrm{Dep}_{\max}$, which control the partitioning process by simply checking the node sample size and depth. In implementation, we could specify either or both of the parameters.

Algorithm 3.2 The simple stopping rule

Input: *Data: A node t, with dataset D_t.*
Parameters: $N_{\min}$ and $\mathrm{Dep}_{\max}$.

Output: *If the recursive partitioning process stops at node t.*

1: $\mathrm{Dep}(t) = \mathrm{Dep}(t_{\mathrm{parent}}) + 1$, *where t_{parent} is the parent node of node t.*
2: ***if*** *the sample size $|D_t| > 2N_{\min}$ or $\mathrm{Dep}(t) < \mathrm{Dep}_{\max}$* ***then***
3: *set the node t as an internal node that needs to be split.*
4: ***else***
5: *set the node t as a leaf node and stop splitting.*

6: ***end if***

With proper $N_{\min}$ and $\text{Dep}_{\max}$, a sufficiently large tree, denoted as $T_{\max}$, can be constructed by applying the split selection algorithm recursively until a stopping condition is satisfied. In the next step, we need to remove the redundant branches in $T_{\max}$ by upward pruning. Here we provide an adapted version to the minimal cost-complexity tree pruning in CART [4]. To accommodate the segmented linear regression models, we replace the node average $\bar{Y}_t$ in the definition of cost-complexity measure in CART with the estimated value by the segmented regression function.

Given a tree T, let $\widehat{m}_{_T}(\cdot)$ be the estimated segmented linear function with partitions dictated by T. Define the accuracy measure of a node t as

$$I(t) = \sum_{(X(i),Y(i))\in D_t} (Y(i) - \widehat{m}_{_T}(X(i)))^2. \tag{3.4}$$

The accuracy measure of T is defined as $I(T) = \sum_{t\in\widetilde{T}} I(t)$, where $\widetilde{T}$ denotes the set of all leaf nodes in T. While in CART, $I(t)$ was defined as $\sum_{(X(i),Y(i))\in D_t}(Y(i) - \bar{Y}_t)^2$ with $\bar{Y}_t$ being the average of Y over node t.

The model complexity of T is measured by $|\widetilde{T}|$, the number of leaf nodes. Taking both $I(T)$ and $|\widetilde{T}|$ into consideration, the cost-complexity measure of T is

$$I_\alpha(T) = n^{-1}I(T) + \alpha|\widetilde{T}|, \tag{3.5}$$

where α is a positive penalizing parameter for the model complexity and n is the sample size of the training data. Based on $I_\alpha(T)$, the smallest minimizing subtree $T(\alpha)$ is defined in the same way as that in CART.

Definition 3.3 *Let $T_{\max}$ be the sufficiently large tree. Given a complexity parameter α, the smallest minimizing subtree $T(\alpha)$ satisfies the following two conditions*:

(i) $I_\alpha(T(\alpha)) = \min_{T\subset T_{\max}} I_\alpha(T)$;

(ii) *if there exists a $T \subset T_{\max}$ satisfying $I_\alpha(T) = I_\alpha(T(\alpha))$, then $T(\alpha) \subset T$.*

The following proposition shows that under the new definition of cost-complexity, the existence and uniqueness of $T(\alpha)$ can still be ensured. Besides, $\{T(\alpha), \alpha > 0\}$ are nested as α is increased, which implies that we do not need a direct search through all possible subtrees to find the minimizer of $I_\alpha(T)$ and hence facilitates the efficient pruning algorithm.

Proposition 3.4 *Let $T_{\max}$ be a sufficiently large tree generated by the simple stopping rule given proper $N_{\min}$ and $\text{Dep}_{\max}$, then*

(i) *Given an α, there exists a smallest minimizing subtree $T(\alpha)$ of $T_{\max}$;*

(ii) *If $\alpha_2 > \alpha_1$, $T(\alpha_2) \subset T(\alpha_1)$.*

Suppose that $T_{\max}$ has K different smallest minimizing subtrees, then by (ii) in Proposition 3.4, there exists an increasing sequence of complexity parameters $\{\alpha_k \,|\, k = 1, \ldots, K\}$ such that $T(\alpha_{k+1}) \subset T(\alpha_k)$, and for $\alpha \in [\alpha_k, \alpha_{k+1})$, $T(\alpha) = T(\alpha_k)$. Therefore, we only need to consider the sequence of nested pruned trees $\{T(\alpha_k)\}_{k=1}^K$, which greatly simplifies the computation. Furthermore, the value of $\{\alpha_k\}_{k=1}^K$ can be calculated explicitly by the definition of $I_\alpha(T)$ and $T(\alpha)$. As was outlined in CART, let T_t be the subbranch of T with node t being its

root, then $\alpha_k = \min_t\{\frac{I(t)-I(T_t)}{n(|\widetilde{T}_t|-1)} \,|\, t \in T \text{ and } t \notin \widetilde{T}\}$ for $T = T_{\max}$ when $k = 1$ and $T = T(\alpha_{k-1})$ when $k > 1$.

In practice, we use cross-validation (CV) to select the complexity parameter α^* and output the pruned tree $T(\alpha^*)$ as the final tree structure. Let $\bar{\alpha}_k = \sqrt{\alpha_k \alpha_{k+1}}$ for $1 \le k \le K-1$ and $\bar{\alpha}_K = \alpha_K$. We consider the K candidate values $\{\bar{\alpha}_k\}_{k=1}^K$ in selecting α^* by CV. Specifically, the training data $\mathcal{L}$ would be randomly divided into V subsets, $\{\mathcal{L}_v, v = 1, \ldots, V\}$. Then, the v-th testing sample is $\mathcal{L}_v$ and the v-th training sample is $\mathcal{L}^{(v)} = \mathcal{L} - \mathcal{L}_v$. Let $T_{\max}^{(v)}$ denote the sufficiently large tree based on the training sample $\mathcal{L}^{(v)}$, and $T^{(v)}(\alpha)$ be the optimal subtree of $T_{\max}^{(v)}$ given α. Let $M_{\mathcal{L}_v}(T^{(v)}(\alpha))$ be the mean squared prediction error (MSPE) of $T^{(v)}(\alpha)$ on the testing sample $\mathcal{L}_v$,

$$M_{\mathcal{L}_v}(T^{(v)}(\alpha)) = \frac{\sum_{(X(i),Y(i))\in\mathcal{L}_v}(\widehat{m}_{T^{(v)}(\alpha)}(X(i)) - Y(i))^2}{|\mathcal{L}_v|},$$

where $\widehat{m}_{T^{(v)}(\alpha)}$ denotes the regression function determined by $T^{(v)}(\alpha)$. Define the CV score function $M^{\text{cv}}(\alpha) = \frac{1}{V}\sum_{v=1}^V M_{\mathcal{L}_v}(T^{(v)}(\alpha))$. Then, the selected complexity parameter $\alpha^* = \text{argmin}_{\alpha\in\{\bar{\alpha}_1,\ldots,\bar{\alpha}_K\}} M^{\text{cv}}(\alpha)$ and the best pruned subtree of $T_{\max}$ is $T(\alpha^*)$, denoted by T^*

Suppose that T^* has $\widehat{L}$ leaf nodes, which correspond to $\widehat{L}$ data partitions $\{\widehat{D}_l\}_{l=1}^{\widehat{L}}$. Confined on each $\widehat{D}_l$, the regression coefficients (α_l, β_l) can be estimated by the OLS estimator, $(\hat{\alpha}_l,\ \hat{\beta}_l) = \text{argmin}_{\alpha,\beta}\sum_{\{i|(X(i),Y(i))\in\widehat{D}_l\}}(Y(i) - \alpha - X^{(r)}(i)'\beta)^2$. Then the final estimate is

$$\widehat{m}_{T^*}(X) = \sum_{l=1}^{\widehat{L}}(\hat{\alpha}_l + X^{(r)'}\hat{\beta}_l)\mathbb{I}(X^{(s)} \in \widehat{D}_l). \tag{3.6}$$

As $X^{(r)}$ is the overall explicit regressors, not everyone of them necessarily owns a non-zero coefficient over D_l. That is, the significant variables on each $\hat{D}_l$ may be different. Besides, as the partitioning process decreases the sample size used in the OLS estimation, we prefer to use fewer covariates on each node. Therefore, we would like to determine a smaller subset of variables that exhibits the strongest effects on each $\widehat{D}_l$. For that purpose, the LASSO (the least absolute shrinkage and selection operator, proposed by [26]) is applied for subset selection. The LASSO estimates are

$$\begin{aligned}
\hat{\alpha}_l^{\text{LASSO}} &= \bar{Y}_{\widehat{D}_l} \\
\hat{\beta}_l^{\text{LASSO}} &= \underset{\beta_l}{\text{argmin}} \sum_{\{i|X^{(s)}(i)\in\widehat{D}_l\}} \left\{(Y(i) - \bar{Y}_{\widehat{D}_l} - X^{(r)}(i)'\beta_l)^2 + \lambda_l \sum_{j=1}^p |\beta_l(j)|\right\},
\end{aligned}$$

where λ_l is a positive penalty parameter to control the extent of shrinkage and $\bar{Y}_{\widehat{D}_l}$ is the sample average of Y over $\widehat{D}_l$. The value of λ_l is determined by the CV procedure within each partition $\widehat{D}_l$. In this way, we get the following estimated function by the SLRT with LASSO as

$$\widehat{m}_{T^*}^{\text{LASSO}}(X) = \sum_{l=1}^{\widehat{L}}(\hat{\alpha}_l^{\text{LASSO}} + X^{(r)'}\hat{\beta}_l^{\text{LASSO}})\mathbb{I}(X^{(s)} \in \widehat{D}_l). \tag{3.7}$$

By doing so, we sacrifice a little bias but reduce the variance and improve the interpretability and predictability of (3.7).

3.3 Weighted Linear Random Forests

Random forests (RF) [3] usually refer to an ensemble of tree predictors of CART, with the bootstrap aggregating (bagging) and random selection of regressors at the node level. The same ensemble construction can be applied with the linear regression trees to improve the performance of SLRTs. We call this procedure linear random forests (LRF) to distinguish it from the traditional RF.

Algorithm 3.5 Linear Random Forest

Input: *Training Data:* $\mathcal{L}$.

Parameters: Ensemble size K*; the number of randomly selected regressors* $p^*(\leq p)$.

Output: *Estimated regression function* $\widehat{m}_{LRF}(\cdot)$.

1: ***for*** i *in* $1, \ldots, K$ ***do***

2: *Generate* $\mathcal{L}_i$ *from the training data* $\mathcal{L}$ *by the random sampling with replacement. The sample size of* $\mathcal{L}_i$ *is the same as that of* $\mathcal{L}$.

3: *Applying the tree construction algorithm of SLRT with simple stopping rules on training data* $\mathcal{L}_i$*, but replace the original regressor* X *with* p^* *randomly selected regressors at each node, to generate the linear regression tree* T_i.

4: *Obtain the estimated segmented linear regression function* $\widehat{m}_{T_i}(\cdot)$ *based on the partitions of* T_i.

5: ***end for***

6: ***return*** *the LRF estimator:* $\widehat{m}_{LRF}(X) = \frac{1}{K}\sum_{i=1}^{K} \widehat{m}_{T_i}(X)$.

In the regression setting, both RF and LRF take the simple average of the predictions given by all trees as the ensemble predictor. As the predictive accuracy of each tree in the forest varies, we propose a weighted version that takes the accuracy of each tree into consideration and puts the final prediction as a weighted average. This is motivated by the following Proposition 3.6 on the optimal linear estimations.

Proposition 3.6 *Let* $Z_1, \ldots, Z_K$ *be independently distributed random variables from a population* $P \in \mathcal{P}$ *having a common mean* μ *and* $\mathrm{Var}(Z_i) = \sigma_i^2$*. Let* $\mathcal{A}$ *be the class of all unbiased linear estimations for* μ*,*

$$\mathcal{A} = \left\{ \hat{\mu} = \sum_{i=1}^{K} c_i Z_i \Big| \sum_{i=1}^{K} c_i = 1 \right\}.$$

Then, the optimal estimator in $\mathcal{A}$ *that minimizes* $\mathrm{E}(\hat{\mu} - \mu)^2$ *is* $\sum_{i=1}^{K} \frac{\sigma_i^{-2}}{\sum_{j=1}^{K} \sigma_j^{-2}} Z_i$.

Since $\mathcal{L}_i$ in LRF Algorithm 3.5 are independently sampled, $\{\widehat{m}_{T_i}(\cdot)\}_{i=1}^{K}$ are conditionally independent given the training sample. Thus, given $X = x$, $\{\widehat{m}_{T_1}(x), \ldots, \widehat{m}_{T_K}(x)\}$ are K conditionally independent random variables. Let $t_{T_i}(x)$ be the leaf node containing x in T_i. As the accuracy measure $I(t_{T_i}(x))$ is the sum of squared errors within the node $t_{T_i}(x)$, $I(t_{T_i}(x))/n_{T_i}(x)$ is a bona fide variance estimator for $\widehat{m}_{T_i}(x)$. Substituting the variances by estimates, we propose the weighted LRF (wLRF) estimator,

$$\widehat{m}_{\mathrm{wLRF}}(X) = \sum_{i=1}^{K} \frac{n_{T_i}(X)/I(t_{T_i}(X))}{\sum_{j=1}^{K} n_{T_j}(X)/I(t_{T_j}(X))} \widehat{m}_{T_i}(X). \tag{3.8}$$

Similarly, we can obtain the weighted version of RF (wRF) with trees generated by CART.

3.4 Experimental Results

Simulation Results In the simulation, we evaluate the empirical performance of SLRT when the underlying model does not conform to the SLR model setting. Consider the regression function $m_2(X)$ in (3.9). Model (3.9) was used in [25] for incremental learning of tree-based models. The training data were generated from $Y = m_2(X) + \varepsilon_2, \varepsilon_2 \sim N(0, 0.01^2), X_1, X_2 \overset{\text{i.i.d}}{\sim} \mathrm{U}(0, 1)$.

$$m_2(X) = \max\{\mathrm{e}^{-10X_1^2},\ \mathrm{e}^{-50X_2^2}, 1.25\mathrm{e}^{-5(X_1^2+X_2^2)}\}. \tag{3.9}$$

We compared the empirical performance of estimating $m_2(X)$ among the proposed SLRT and CART, as well as the derived ensemble methods LRF, wLRF and RF, wRF.

Figures 19–22 show the estimated surfaces based on one realized training data of sample size 1000, including the piecewise linear estimation generated by SLRT with the simple stopping rule, the piecewise constant estimation generated by CART, and the ensemble estimations by the wLRF and wRF with 100 resampled trees. Figures 19–22 show that the SLRT and wLRF can pick up the central peak of $m_2(X)$ better than the CART and its ensemble version wRF.

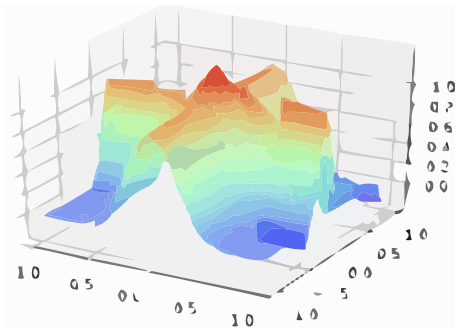

Figure 19 Greedy SLRT

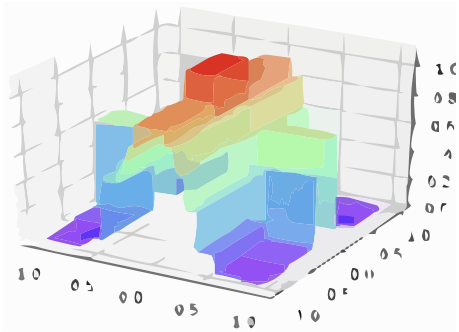

Figure 20 CART

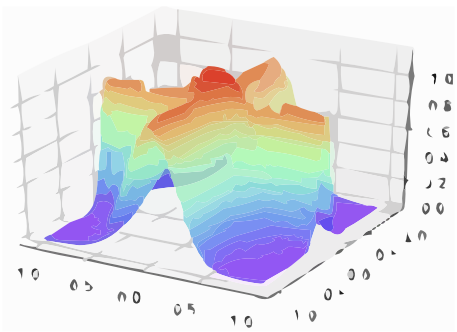

Figure 21 wLRF (ensemble of greedy SLRT)

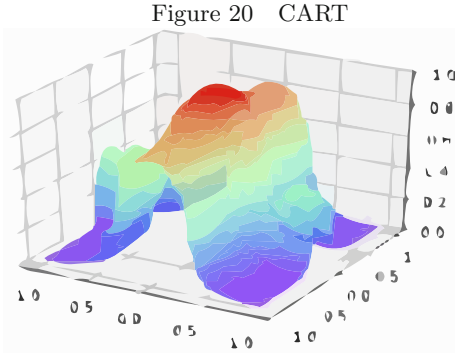

Figure 22 wRF (ensemble of CART)

Figure. Estimated surfaces based on training data of sample size 1000 for Model (3.9), generated by SLRT using Algorithm 2.1 and the simple stopping rule with $N_{\min}$=20 and $\mathrm{Dep}_{\max}$=10 (a), CART with the same stopping rule (b), ensemble estimators wLRF (c) and wRF (d) with ensemble size 100

In the simulation, training data of the sample sizes 500, 1000 and 2000 were considered. Table 1 reports the average and standard errors of the RMSPEs based on the single tree predic-

tors (SLRT and CART) and the ensemble predictors (LRF, wLRF and RF, wRF). The average and standard errors of the number of leaf nodes are reported for the single tree predictors.

	RMSPE		
	500	1000	2000
SLRT	0.0689 (0.0120)	0.0366 (0.0051)	0.0216 (0.0015)
LRF	0.0562 (0.0063)	0.0244 (0.0020)	0.0155 (5e-04)
wLRF	0.0471 (0.0070)	0.0200 (0.0019)	0.0134 (5e-04)
CART	0.1306 (0.0081)	0.0945 (0.0067)	0.0646 (0.0033)
RF	0.1652 (0.0070)	0.1277 (0.0037)	0.0953 (0.0030)
wRF	0.1443 (0.0063)	0.1074 (0.0043)	0.0733 (0.0040)

Table 1 The average (standard errors) of RMSPEs of the greedy SLRT and the ensemble versions (LRF, wLFR), compared with CART and its ensemble estimators RF and wRF. Both SLRT and CART used the same simple stopping rule with $N_{\min} = 20$ and $\mathrm{Dep}_{\max} = 10$ for all simulation settings with different sample sizes

As for the prediction accuracy, the RMSPEs of all methods decreased as the sample size was enlarged, where the smallest RMSPE 0.0134 was attained by wLRF based on training data with sample size 2000, which was close to the standard error of ε_2 in Model (3.9).

Predictive Performance on Public Datasets The predictive performances of the proposed methods were examined on the following nine datasets from the UCI Machine Learning Repository [6] and StatLib Datasets Archive.

• *Boston Housing*: The data concerns the median housing values (medv) on 506 census tracts around Boston, where 13 covariates are collected. We take the logarithm of medv as the response, to be consistent with the existing analysis on the dataset in [20].

• *Computer Hardware*: The target variable is the published relative CPU performance. There are 6 numeric attributes about the cycle time, memory and number of channels.

• *Auto-MPG*: The target variable mpg refers to city-cycle fuel consumption in miles per gallon. We discarded the 'car name' attribute and use the other 7 attributes for predictions of mpg.

• *Auto-mobile*: This dataset has 159 complete records of 26 attributes. We used the other 25 attributes to predict the logarithm of price, where the 'make' attribute is transformed to a binary variable.

• *Pyrimidine*: The goal is to explore the inhibition of dihydrofolate reductase by pyrimidines using the information of 74 pyrimidines. The predictive attributes include 26 numeric variables about the physiochemical and structural properties of pyrimidines.

• *Kinematics*: This dataset contains 8192 cases about the forward kinematics of an 8 link robot arm. The goal is to predict the distance of the end-effector from the target, using $\theta_1, \ldots, \theta_8$, the 8 angular positions of the joints as predictive attributes.

• *Abalone*: The relevant task is to predict the age of abalone from the nominal sex variable and seven other numeric variables of physical measurements.

• *Parkinson*: The dataset is composed of a range of biomedical voice measurements from people with early-stage Parkinson disease [18]. We use the 16 voice measures to predict the total UPDRS.

• *News Popularity*: The dataset collects 58 features of articles published by Mashable in two years [8], including 39644 complete cases. Aggregating the 13 variables about 'weekday' and 'channel' into two categorical variables, we use the 47 covariants to predict the log(shares).

The proposed SLRT with the least square estimates (SLRT_{LS}) and the LASSO method ($\text{SLRT}_{\text{LASSO}}$) were compared with three tree-based methods: CART [4], GUIDE [20] and MARS [10], which all used recursive partitioning. The tree structures in SLRT were constructed using the split selection Algorithm 3.1 and the simple stopping rule. To make results comparable, for each dataset, the same simple stopping rule with a common $N_{\min}$ was used for different methods. The specific values of $N_{\min}$ were shown in Table 2, which were specified according to the sample sizes of training data.

Ensemble predictors LRF and wLRF were the random forests (RF) and the proposed weighted RF equipped with SLRT_{LS} as the base predictor. The conventional RF based on CART (RF_{CART}) as well as the weighted version wRF were also implemented to serve as the benchmarks. To make the results of random forests and their weighted versions comparable, their predictions were based on the same ensembles of trees while the difference was only in the ways of aggregating, the simple average or the weighted average.

Table 2 provides the basic information about the characteristics of each dataset, where the sample size ranges from 74 to 39644, and five of the datasets include categorical variables.

Datasets	Sample Size	Number of Predictive Attributes (Numeric+ Categorical)	$N_{\min}$
Boston Housing	506	13 (12+1)	60
Computer Hardware	209	6 (6+0)	20
Auto-MPG	392	7 (4+3)	40
Auto-mobile	159	25 (15+10)	20
Pyrimidine	74	26 (26+0)	50
Kinematics	8192	8 (8+0)	180
Parkinson	5875	16 (16+0)	200
Abalone	4176	8 (7+1)	10
News Popularity	39644	47 (44+3)	1000

Table 2 Detailed aspects of the nine datasets: the sample sizes, the dimensions of the predictive attributes, and the parameters $N_{\min}$ of the simple stopping rule for each dataset

Table 3 reports the root mean squared prediction errors (RMSPE) by the 10-fold CV, where the integers in parentheses indicate the ranks within the single or the ensemble predictors, respectively.

Datasets	Single Predictors					Ensemble Predictors			
	$SLRT_{LS}$	$SLRT_{LASSO}$	GUIDE	CART	MARS	LRF	wLRF	RF	wRF
Boston Housing (506)	0.174(2)	***0.170***(1)	0.187(4)	0.262(5)	0.179(3)	0.162(2)	***0.158***(1)	0.218(4)	0.200(3)
ComputerHardware (209)	47.89(2)	***47.40***(1)	48.06(3)	62.60(5)	54.17(4)	38.49(2)	***36.77***(1)	64.91(4)	39.08(3)
Auto-MPG (392)	2.831(2)	***2.791***(1)	3.545(4)	3.680(5)	2.942(3)	2.633(2)	***2.614***(1)	3.273(3)	3.240(4)
Auto-mobile (159)	0.154(2)	***0.140***(1)	0.231(5)	0.192(4)	0.184(3)	0.160(1)	***0.172***(4)	0.165(3)	0.162(2)
Kinematics (8192)	0.139(2)	***0.138***(1)	0.140(3)	0.257(5)	0.198(4)	0.117(2)	***0.115***(1)	0.249(4)	0.248(3)
Abalone (4176)	2.162(4)	***2.143***(1)	2.151(2)	2.497(5)	2.161(3)	2.116(2)	***2.113***(1)	2.458(4)	2.456(3)
Parkinson (5875)	9.374(3)	9.327(2)	***9.300***(1)	10.534(5)	9.660(4)	8.691(2)	***8.679***(1)	10.326(4)	10.317(3)
Pyrimidine (74)	0.088(2)	0.093(3)	0.096(5)	0.094(4)	***0.074***(1)	0.078(4)	0.076(3)	0.073(2)	***0.049***(1)
News Popularity (39644)	0.877(4)	0.872(2)	0.873(3)	0.903(5)	***0.865***(1)	***0.868***(1)	***0.868***(1)	0.901(3)	0.901(3)
Average Rank	2.6	***1.4***	3.3	4.8	2.9	2.0	***1.6***	3.4	2.8

Table 3 RMSPE (rank) of 10-fold CV on 9 data sets: data name (sample size). The best performance is marked in ***bold italic***, within each group of single predictors and ensemble predictors. Average Rank is the average of ranks on all datasets for each method

Among the five single tree predictors, $SLRT_{LASSO}$ attained the best prediction in six datasets, MARS in two, and $SLRT_{LS}$ and GUIDE in one, respectively. The average ranks of $SLRT_{LS}$ and $SLRT_{LASSO}$ were ahead of CART, GUIDE, and MARS. This demonstrates the advantages of the proposed SLRT. Directly comparing SLRT with GUIDE, LRT_{LS} had better prediction in 6 datasets and $SLRT_{LASSO}$ in 8 datasets. SLRT also compared favorably to MARS, where $SLRT_{LS}$ had better performance in 6 datasets and $SLRT_{LASSO}$ was better in 7 datasets. CART appeared to be the worst predictor in seven datasets. The better performance of $SLRT_{LASSO}$ over $SLRT_{LS}$ showed the benefits of conducting the variables selection on the leaf nodes.

The ensemble predictors LRF and wLRF showed better performance than the conventional RF equipped with CART in 8 out of 9 datasets. Meanwhile, the ensemble predictors tended to outperform the single predictors, which suggests the effects of the bagging operation. The proposed wLRF also showed improved predictions over the RF, which benefited from the weighting procedure that reduced the contribution of those underperforming trees. On average, wLRF attained the best average rank at 1.6. It was somehow more striking to see that the single predictor LRT even outperforms the ensemble predictors of RF and wRF based on CART over all datasets but the one about Pyrimidine.

4 Conclusions

In this paper, we propose the Segmented Linear Regression Tree (SLRT), which adopts the same recursive partitioning procedure as the traditional CART regression tree but extends the leaf model from constant fitting to linear regression models.

On the one hand, SLRT can be regarded as a new technique to identify the split boundaries for the Segmented Linear Regression (SLR) models. The SLR models inherit the easy interpretation of the linear models while offering more flexibility as they reflect the changing regimes of the regression over different regions of the function domain. SLRT connects segmented models with the regression trees by fitting linear models over the leaf nodes, which

provides a computationally efficient and completely data-driven way for estimating the segmented linear regression models. As for the split selection under the model specification of SLR, we propose the Algorithm 2.4 to select the split boundary based on the pattern of conditional Kendall's τ coefficients and a sequential changepoint-in-mean detection method. This algorithm is supported by theoretical analyses and validated by simulation results.

On the other hand, SLRT can serve as an extension of CART regression trees that provides a more accurate approximation to the general regression functions. From this point of view, we propose the heuristic split selection algorithm that selects the split such that the information contained in the OLS residuals can be further extracted in the partitioned data subsets. With this intuition, we propose the criterion function based on cumulative Kendall's τ coefficients and select the split that maximizes the criterion function (Algorithm 3.1). Compared to the split selection for the SLR models, this method does not require the model specifications to be correct and hence can be applied to the general case to improve the predictive accuracy. Besides, we implant the SLRT as the base predictor in RF and wRF to create more accurate breeds of ensemble predictors, LRF and wLRF. The proposed procedures are evaluated by both numerical simulations and case studies, which show advantageous predictive accuracy over other tree-based methods, and in creating more powerful breeds of ensemble predictors.

Conflict of Interest The authors declare no conflict of interest.

References

[1] Bishop, Y. M., Fienberg, S. E., Holland, P. W.: Discrete Multivariate Analysis: Theory and Practice, Cambridge, Massachusetts, 1975

[2] Breiman, L.: Bagging predictors. *Machine Learning*, **24**, 123–140 (1996)

[3] Breiman, L.: Random forests. *Machine Learning*, **45**, 5–32 (2001)

[4] Breiman, L., Friedman, J. H., Olshen, R. A., et al.: Classification And Regression Trees, Wadsworth International Group, New York, 1984

[5] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM International Conference on Knowledge Discovery and Data Mining, ACM, 2016, 785–794

[6] Dheeru, D., Karra Taniskidou, E.: UCI Machine Learning Repository, http://archive.ics.uci. edu/ml. Irvine, CA: University of California, School of Information and Computer Science (2019)

[7] Einhorn, H. J.: Alchemy in the behavioral sciences. *Public Opinion Quarterly*, **36**(3), 367–378 (1972)

[8] Fernandes, K., Vinagre, P., Cortez, P.: A proactive intelligent decision support system for predicting the popularity of online news. In: Portuguese Conference on Artificial Intelligence, Springer International Publishing, 2015, 535–546

[9] Freund, Y., Schapire, R. E.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**(1), 119–139 (1997)

[10] Friedman, J. H.: Multivariate adaptive regression splines. *The Annals of Statistics*, **19**, 1–67 (1991)

[11] Friedman, J., Hastie, T., Tibshirani, R.: The Elements of Statistical Learning, Springer Series in Statistics, New York, USA, 2001

[12] Gonzaloa, J., Pitarakisb, J.-Y.: Estimation and model selection based inference in single and multiple threshold models, *Journal of Econometrics*, **3**, 1–34 (2002)

[13] Ke, G., Meng, Q., Finley, T., et al.: Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, **30**, 3146–3154 (2017)

[14] Kendall, M. G.: A new measure of rank correlation. *Biometrika*, **30**, 81–93 (1983)

[15] Killick, R., Eckley, I.: Changepoint: An R package for changepoint analysis. *Journal of Statistical Software*, **58**(3), 1–19 (2014)

[16] Kim, H. J., Yu, B., Feuer, E. J.: Selecting the number of change-points in segmented line regression. *Statistica Sinica*, **19**(2), 597–609 (2009)

[17] Kim, J., Kim, H. J.: Asymptotic results in segmented multiple regression, *Journal of Multi-variate Analysis*, **99**, 2016–2038 (2008)

[18] Little, M. A., McSharry, P. E., Hunter, E. J., et al.: Accurate telemoni toring of parkinson's disease progression by non-invasive speech tests, *IEEE: Transactions on Biomedical Engineering*, **56**, 1015–1022 (2009)

[19] Liu, J., Wu, S. Y., Zidek, et al.: On segmented multivariate regression. *Statistica Sinica*, **7**, 497–525 (1997)

[20] Loh, W.-Y.: Regression trees with unbiased variable selection and interaction detection. *Statistica Sinica*, **12**(2), 1–26 (2002)

[21] Loh, W.-Y. and Zheng, W.: Regression trees for longitudinall and muti-response data. *The Annals of Applied Statistics*, **7**, 495–522 (2013)

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

[23] Oiwa, H. and Fujimaki, R.: Partition-wise linear models. *Advances in Neural Information Pro-cessing Systems*, **27**, (2014)

[24] Perron, P., Qu, Z.: Estimating restricted structural change models. *Journal of Econometrics*, **134**, 373–399 (2006)

[25] Potts, D., Sammut, C.: Incremental learning of linear model trees. *Machine Learning*, **61**, 5–48 (2002)

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