Hierarchical Shrinkage: improving the accuracy and interpretability of tree-based methods

Abhineet Agarwal 1 Yan Shuo Tan 2 Omer Ronen 2 Chandan Singh 3 Bin Yu 2,3

Abstract

Tree-based models such as decision trees and random forests (RF) are a cornerstone of modern machine-learning practice. To mitigate overfitting, trees are typically regularized by a variety of techniques that modify their structure (e.g. pruning). We introduce Hierarchical Shrinkage (HS), a post-hoc algorithm that does not modify the tree structure, and instead regularizes the tree by shrinking the prediction over each node towards the sample means of its ancestors. The amount of shrinkage is controlled by a single regularization parameter and the number of data points in each ancestor. Since HS is a post-hoc method, it is extremely fast, compatible with any tree-growing algorithm, and can be used synergistically with other regularization techniques. Extensive experiments over a wide variety of real-world datasets show that HS substantially increases the predictive performance of decision trees, even when used in conjunction with other regularization techniques. Moreover, we find that applying HS to each tree in an RF often improves accuracy, as well as its interpretability by simplifying and stabilizing its decision boundaries and SHAP values. We further explain the success of HS in improving prediction performance by showing its equivalence to ridge regression on a (supervised) basis constructed of decision stumps associated with the internal nodes of a tree. All code and models are released in a full-fledged package available on Github.

1. Introduction

Decision tree models, used for supervised learning since the 1960s (Morgan & Sonquist, 1963; Messenger & Mandell, 1972; Quinlan, 1986), have recently attained renewed prominence because they embody key elements of interpretability: shallow trees are easily described and visualized, and can even be implemented by hand. While the precise definition and utility of interpretability have been a subject of much debate (Murdoch et al., 2019; Doshi-Velez & Kim, 2017; Rudin, 2019; Rudin et al., 2021), all agree that it is an important notion in high-stakes decision-making, such as medical-risk assessment and criminal justice. For this reason, decision trees have been widely applied in both areas (Steadman et al., 2000; Kuppermann et al., 2009; Letham et al., 2015; Angelino et al., 2017).

By far the most popular decision tree algorithm is Breiman et al. (1984)’s Classification and Regression Trees (CART). These can be ensembled to form a Random Forest (RF) (Breiman, 2001) or used as weak learners in Gradient Boosting (GB) (Friedman, 2001): both algorithms have achieved state-of-the-art performance over a wide class of prediction problems (Caruana & Niculescu-Mizil, 2006; Caruana et al., 2008; Fernandez-Delgado et al., 2014; Olson et al., 2018; Hooker & Mentch, 2021), and are implemented in popular machine learning packages such as ranger (Wright et al., 2017) and scikit-learn (Pedregosa et al., 2011). Variants of these algorithms, such as Basu et al. (2018)’s iterative random forest for finding stable interactions, have found use in scientific applications.

In view of the widespread use of tree-based methods, we seek to provide a new lens on their regularization. On the one hand, decision trees often obey traditional statistical wisdom in that they need to be regularized to prevent overfitting. In practice, this is carried out by specifying an early stopping condition for tree growth, such as a maximum depth, or alternatively, pruning the tree after it is grown (Friedman et al., 2001). These procedures, however, only regularize tree models via their tree structure, and it is usually taken for granted that the prediction over each leaf should be the average response of the training samples it contains. We show that this can be very limiting: shrinking these predictions in a hierarchical fashion can significantly...
reduce generalization error in both regression and classification settings (e.g. see Fig 1).

On the other hand, trees used in an RF are usually not explicitly regularized and interpolate the data by being grown to purity (e.g. see the default settings of scikit-learn and ranger). Instead, RF prevents overfitting by relying upon the randomness injected into the algorithm during tree growth, which acts as a form of implicit regularization (Breiman 2001; Mentch & Zhou 2019). We show that apart from this implicit regularization, more regularization, in the form of hierarchical shrinkage, does improve generalization even while using a smaller ensemble for many data sets.

Equally important, regularizing RFs also improves the quality of their post-hoc interpretations. RFs are usually interpreted via their feature and interaction importances, which have been used to provide scientific insight in areas such as remote sensing and genomics (Svetnik et al., 2003; Evans et al., 2011; Belgiu & Drăgut, 2016; Diaz-Uriarte & De Andres, 2006; Boulesteix et al., 2012; Chen & Ishwaran, 2012; Basu et al., 2018; Behr et al., 2020). The reproducibility and scientific meaning of such interpretations become questionable when the underlying RF model has poor predictive performance (Murdoch et al., 2019), or when they are highly sensitive to data perturbations (Yu, 2013). We show that HS improves the interpretability of RF by both simplifying and stabilizing its decision boundaries and SHAP values (Lundberg & Lee, 2017) on a number of real-world data sets.

Our proposed method, which we call Hierarchical Shrinkage (HS), is an extremely fast and simple yet effective algorithm for the post-hoc regularization of any tree-based model. It does not alter the tree structure, and instead replaces the average response (or prediction) over a leaf in the tree with a weighted average of the mean or average responses over the leaf and each of its ancestors. The weights depend on the number of samples in each leaf, and are controlled by a single regularization parameter $\lambda$ that can be tuned efficiently via generalized cross validation. HS is agnostic to the way the tree is constructed and can be applied post-hoc to trees constructed with greedy methods such as CART and C4.5 (Quinlan 2014), as well optimal decision trees grown via dynamic programming or integer optimization techniques (Lin et al., 2020).

A more naive form of shrinkage, which we call leaf-based shrinkage (LBS), appears as part of XGBoost (Chen & Guestrin, 2016): whenever a new tree is grown, the average response (prediction) over each leaf is shrunk directly towards the sample mean of the responses. LBS also occurs in Bayesian Additive Regression Trees (BART) (Chipman et al., 2010), which grows an ensemble of trees via a backfitting MCMC algorithm. Comparing LBS to HS on several real-world datasets shows that HS uniformly has better predictive performance than LBS.

We explain the advantages of HS by building on recent work by Klusowski (2021), who uses decision stumps associated to each interior node of a tree to construct a new (supervised) feature representation. The original tree model is recovered as the linear model obtained by regressing the responses on the supervised features. We show that HS is exactly the ridge regression solution in this supervised feature space, while LBS can also be viewed as ridge regression, but with a different (supervised) feature space (of the same dimension) that relies only on the leaf nodes. This allows us to use ridge regression calculations heuristically to partially explain both the reasonableness of the shrinkage scaling in HS, as well as our empirical evidence that HS achieves consistently better predictive accuracy than LBS (see Sec 3).

The rest of the paper is organized as follows. Sec 2 gives...
Hierarchical Shrinkage

a formal statement of HS, and discusses several computational considerations. See \cite{varIOUS} for details on how HS improves the interpretability of RFs.

2. The Hierarchical Shrinkage (HS) algorithm

Throughout this paper, we work in the supervised learning setting where we are given a training set \( \mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n \), from which we learn a tree model \( \hat{f} \) for the regression function. Given a query point \( x \), let \( t_L \subset t_{L-1} \subset \cdots \subset t_0 \) denote its leaf-to-root path, with \( t_L \) and \( t_0 \) representing its leaf node and the root node respectively. For any node \( t \), let \( N(t) \) denote the number of samples it contains, and \( \hat{E}_t(y) \) the average response. The tree model prediction can be written as the telescoping sum:

\[
\hat{f}(x) = \hat{E}_{t_0}(y) + \sum_{l=1}^{L} (\hat{E}_{t_l}(y) - \hat{E}_{t_{l-1}}(y)).
\]

HS transforms \( \hat{f} \) into a shrunk model \( \hat{f}_\lambda \) via the formula:

\[
\hat{f}_\lambda(x) := \hat{E}_{t_0}(y) + \sum_{l=1}^{L} \frac{\hat{E}_{t_l}(y) - \hat{E}_{t_{l-1}}(y)}{1 + \lambda/N(t_{l-1})},
\]

where \( \lambda \) is a hyperparameter chosen by the user, for example by cross validation. We emphasize that HS maintains the tree structure, and only modifies the prediction over each leaf node.

Since HS continues to make a constant prediction over each leaf node, our method thus comprises a one-off modification of these values. This can be computed in \( O(m) \) time, where \( m \) is the total number of nodes in the tree. No other aspects of the underlying data structure are modified, with test time prediction occurring in exactly the same way as in the original tree. Moreover, our method HS does not even need to see the original training data, and only requires access to the fitted tree model. These features make it extremely lightweight and easy to implement, as we have done in the open-source package imodels \cite{Singh2021}.

By applying HS to each tree in an ensemble, it can be generalized to methods such as RF and gradient boosting.

While not typically done, it is possible to regularize RFs via other hyperparameters such as maximum tree depth. Tuning these hyperparameters, however, requires refitting the RF at every value in a grid. This quickly becomes computationally expensive, even for moderate dataset sizes\footnote{Many popular tree-building algorithms such as CART have a run time of \( O(pn^2) \) for constructing a binary tree.}. Over multiple folds in a cross-validation (CV) set up. In contrast, since HS is applied post-hoc, we only need to fit the RF once per CV fold, leading to potentially enormous time savings. In addition, due to the connection between our method and ridge regression, it is even possible to get away with fitting the RF only once by using generalized cross-validation \cite{Golub1979}.

We also note the formula for LBS:

\[
\hat{f}_L^2(x) := \hat{E}_{t_0}(y) + \hat{E}_{t_L}(y) - \hat{E}_{t_0}(y) + \frac{1}{1 + \lambda/N(t_L)}.
\]

Expanding this into a telescoping sum similar to (1), we see that the major difference between the two formulas is that whereas LBS shrinks each term by the same factor, HS shrinks each term by a different amount, with the amount of shrinkage controlled by the number of samples in the ancestor. This increased flexibility leads to better prediction performance for the final model, as evidenced by our results presented in the next section.

3. HS as ridge regression on supervised features

Recent work by Klusowski \cite{Klusowski2021} showed that decision trees are linear models on features obtained via supervised feature learning. To see this, consider a tree model \( \hat{f} \), with a fixed indexing of its interior nodes \( \{t_0, t_1, \ldots, t_{m-1}\} \). We first associate to each node \( t \) the decision stump

\[
\psi_t(x) = \frac{N(t_L)1\{x \in t_L\} - N(t_R)1\{x \in t_R\}}{\sqrt{N(t_L)N(t_R)}},
\]

where \( t_L \) and \( t_R \) denote the left and right children of \( t \) respectively. This is a tri-valued function that is positive on the left child, negative on the right child, and zero everywhere else. Concatenating the decision stumps together yields a supervised feature map via \( \Psi(x) = (\psi_{t_0}(x), \ldots, \psi_{t_{m-1}}(x)) \) and a transformed training set \( \Psi(\mathcal{D}_n) \in \mathbb{R}^{n \times m} \). One can easily check that these feature vectors are orthogonal in \( \mathbb{R}^n \), and furthermore that their squared \( \ell_2 \) norms are the number of samples contained in their corresponding nodes: \( \|\psi_t\|^2 = N(t_i) \).

More interestingly, Klusowski \cite{Klusowski2021} was able to show (see Lemma 3.2 therein) that we have functional equality between the tree model and the kernel regression model with respect to the supervised feature map \( \Psi \), or in other words, \( \hat{f}(x) = \hat{\beta}^T \Psi(x) \), where \( \hat{\beta} = \Psi(\mathcal{D}_n)^T y \). An easy extension of his proof yields the following result.

\textbf{Theorem 1.} Let \( \hat{\beta}_\lambda \) be the solution to the ridge regression
Hierarchical Shrinkage

\[
\min_{\beta} \left\{ \sum_{i=1}^{n} \left( \beta^T \Psi(x_i) - y_i \right)^2 + \lambda \|\beta\|^2 \right\}. \tag{4}
\]

We have the functional equality \( \hat{f}_L(x) = \beta_L^T \Psi(x) \).

Proof. See Appendix S5. \qed

Since the decision stumps \( 3 \) are orthogonal, we can decompose \( 4 \) into \( m \) independent univariate ridge regression problems, one with respect to each node \( t \):

\[
\min_{\beta} \left\{ \sum_{i=1}^{n} (\beta \psi_t(x_i) - y_i)^2 + \lambda \beta^2 \right\}. \tag{5}
\]

Next, we use this connection of HS to ridge regression to argue heuristically that the same \( \lambda \) works well for each regression subproblem \( 5 \). This helps to justify our choice of denominator for each term in the HS formula \( 1 \) (a different choice would have led to a rescaling of the features \( \psi_t, \))

Assume for the moment that the tree structure and hence the feature map is independent of the responses, which can be achieved via sample splitting. This is known in the literature as the “honesty condition”, and has been widely used to simplify the analysis of tree-based methods (Athey & Imbens 2016). Define \( \beta_s \in \mathbb{R}^m \) to have the value

\[
\beta_s(t) := \sqrt{\frac{N(t_L)N(t_R)}{N(t)}} (E\{y \mid t_L\} - E\{y \mid t_R\}), \tag{6}
\]

for the coordinate associated with each node \( t \). For any query point \( x \), \( \beta_s^T \Psi(x) \) gives the mean response over the leaf \( t(x) \) containing it. Furthermore, knowing \( \Psi(x) \) is equivalent to knowing the leaf containing \( x \). Putting these two facts together show that the population residuals \( r_{ij} := y_i - \beta_s^T \Psi(x) \) satisfy \( E\{r_{ij} \mid \Psi(x_i)\} = 0 \), so that we have a generative linear model, in which we can calculate that the optimal regularization parameter for \( 5 \) is equal to \( \lambda_{opt}(t) = \sigma^2(t)/\beta_{opt}(t)^2 \), where \( \sigma^2(t) \) is roughly equal to the conditional variance\(^5\) of the residual over \( t \).

Given the connection between impurity and residual variance, if the tree model \( \hat{f} \) considered in this section is grown using an impurity decrease stopping condition, we should expect the residual variance to be relatively similar over all leaves, so that \( \sigma^2(t) \) also does not vary too much over different nodes. Meanwhile (see Appendix S5.2)

\[
(\frac{1}{N(t_L)} \text{Var} \{r \mid t_L\} - \frac{1}{N(t_R)} \text{Var} \{r \mid t_R\})^2 \approx \text{diam}(t)^2 \approx 2^{-2 \text{depth}(t) / p} \tag{7}
\]

where \( p \) is the dimension of the original feature space. Since the maximum depth is typically \( O(\log n) \), \( \lambda_{opt}(t) \) also does not vary too much across different nodes, and we do not lose too much by using a common value of \( \lambda \) across all the univariate subproblems \( 5 \) corresponding to different decision stump features.

A more naive (supervised) tree-based feature map is the one-hot encoding of an original feature vector obtained by treating the leaf index as a categorical variable. We denote this using \( \Xi \). While \( \Xi \) can be obtained from \( \Psi \) via an invertible linear transformation, the two maps result in different kernels, and thus different ridge regression problems. Indeed, the ridge regression solution with respect to \( \Xi \) is equivalent to performing LBS on the tree model. The leaf indicator features are also orthogonal, so we may similarly decompose this ridge regression problem into independent univariate subproblems, one for each leaf. However, in this case, the population regression vector \( \beta^*_s \), for which \( \beta^*_s^T \Xi(x) \) gives the expected response over each leaf, has coordinates equal to the population expectation over the leaves: \( \beta^*_s(t) = E\{y \mid t\} \). As such, the optimal regularization parameters for different leaf nodes could be very different, and we lose more by having to use a common value of \( \lambda \).

In practice, sample splitting is rarely done, and the feature map depends on the responses. Nonetheless, we believe that the heuristics detailed above continue to hold to a certain extent as shown in our experimental results.

4. HS improves predictive performance on real-world datasets

4.1. Data overview

In this section, we study the performance of HS on a collection of classification and regression datasets selected as follows. For classification, we consider a number of datasets used in the classic Random Forest paper (Breiman 2001; Asuncion & Newman 2007), as well as two that are commonly used to evaluate rule-based models (Wang 2019). For regression, we consider all regression datasets used by (Breiman 2001) with at least 200 samples, as well as a variety of data-sets from the PMLB benchmark (Romano et al. 2020) ranging from small to large sample sizes. Table 1 displays the number of samples and features present in each dataset, with more details provided in Appendix S1.

In all cases, \( 2/3 \) of the data is used for training (hyperparameters are selected via 3-fold cross-validation on this set) and \( 1/3 \) of the data is used for testing.
Hierarchical Shrinkage

| Name                        | Samples | Features |
|-----------------------------|---------|----------|
| Heart                       | 270     | 15       |
| Breast cancer               | 277     | 17       |
| Haberman                    | 306     | 3        |
| Ionosphere                  | 351     | 34       |
| Diabetes                    | 768     | 8        |
| German credit               | 1000    | 20       |
| Juvenile                    | 3640    | 286      |
| Recidivism                  | 6172    | 20       |

| Name                        | Samples | Features |
|-----------------------------|---------|----------|
| Friedman                    | 200     | 10       |
| Friedman3                   | 200     | 4        |
| Diabetes                    | 442     | 10       |
| Geographical music          | 1059    | 117      |
| Red wine                    | 1599    | 11       |
| Abalone                     | 4177    | 8        |
| Satellite image             | 6435    | 36       |
| CA housing                  | 20640   | 8        |

Table 1. Real-world datasets analyzed here for classification (top panel) and regression (bottom panel).

4.2. HS improves prediction performance for commonly used tree methods

The prediction performance results for classification and regression are plotted in Fig 2A and Fig 2B respectively, with the number of leaves, as a proxy for the model complexity, plotted on the x-axis. We consider trees grown using four different techniques: CART, CART with cost-complexity pruning (CCP), C4.5, and GOSDT (Lin et al., 2020), a method that grows optimal trees in terms of the cost-complexity penalized misclassification loss. To reduce clutter, we only display the classification results for CART and CART with CCP in Fig 2A/B and defer the results for C4.5 (Fig S3) and GOSDT (Appendix S3.2) to the appendix.

For each of the four tree-growing methods, we grow a tree to a fixed number of leaves $m$ for several different choices of $m \in \{2, 4, 8, 12, 15, 20, 24, 28, 30, 32\}$ (in practice, $m$ would be pre-specified by a user or selected via cross-validation). For each tree, we compute its prediction performance before and after applying HS, where the regularization parameter for HS is selected from the set $\lambda \in \{0.1, 1.0, 10.0, 25.0, 50.0, 100.0\}$ via cross-validation. Results for each experiment are averaged over 10 random data splits. We observe that HS (solid lines in Fig 3A,B) does not hurt prediction in any of our data sets, and often leads to substantial performance gains. For example, taking $m = 15$, we observe an average increase in relative predictive performance (measured by AUC) of 6.2%, 6.5%, 8% for HS applied to CART and CART with CCP, and C4.5 respectively for the classification data sets. For the regression data sets with $m = 15$, we observe an average relative increase in $R^2$ performance of 9.8%, 10.1% for CART and CART with CCP respectively.

As expected, the improvements tend to be larger when $m$ increases and for smaller datasets (e.g. the top row of Fig 3A and Fig 3B), although for larger datasets we see substantial improvements using HS once the number of leaves in the model is increased (Appendix S3.3).

The fact that improvements hold for CART (CCP) shows that the effect of HS is not entirely replicated by tree structure regularization, and instead, the two regularization methods can be used synergistically. Indeed, applying HS can lead to the selection of a larger tree. Since tree models are sometimes used for subgroup search, larger trees from HS could allow for the discovery of otherwise undetected subgroups.

Fig 2 shows a simulation result analyzing the bias-variance tradeoff for CART with and without HS. Here, data is generated from a linear model with Gaussian noise added during training (see Appendix S2 for experimental details, and other simulations). While predictive performance curves are often U-shaped because of the bias-variance tradeoff, those for HS are monotonic since HS is able to effectively reduce variance. The optimal regularization parameter $\lambda$ decreases with the total number of leaves; this is corroborated by our calculations in Sec 3.

![Figure 2](image)

4.3. HS outperforms LBS

We next compare the performance of HS to that of leaf-based shrinkage (LBS), which is used in XGBoost. Fig 3C shows that hsCART tends to outperform CART (LBS), when repeating the same experiments as in Fig 3A; regression results are pushed to Appendix S3.3 due to space constraints.

For CART (CCP), we grow the tree to maximum depth, and tune the regularization parameter to yield $m$ leaves.
Hierarchical Shrinkage

Figure 3. Hierarchical Shrinkage (solid lines) often improves predictive performance across various datasets, particularly for small datasets. (A) Top two rows show results for classification datasets (measured by AUC of the ROC curve) and (B) the next two rows show results for regression datasets (measured by $R^2$). HS often significantly improves the performance over CART, CART with CCP, and (C) leaf-based shrinkage. (D) HS even improves results for Random Forests as a function of the number of trees. Across all panels, errors bars show standard error of the mean computed over 10 random data splits. Note that the y-axis scales differ across plots.
4.4. HS improves prediction performance for RF

As mentioned earlier, trees in an RF are typically grown to purity without any constraints on depth or node size. Nonetheless, Mentch & Zhou (2019) argues that the \textit{mtry} parameter, which controls the degree of feature sub-selection, “serves much the same purpose as the shrinkage penalty in explicitly regularized regression procedures like lasso and ridge regression.” This parameter is typically set to a default value and is not tuned. We compare the performance of regularizing RF via HS against maximum tree depth and \textit{mtry}, tuning the hyperparameter for each method via cross validation. We repeat this for several different choices of \( B \), the number of trees in the RF, and like before, average the results over 10 random data splits.

The results, displayed in Fig[3D] show that HS significantly improves the prediction accuracy of RF across the datasets we considered. Moreover, HS clearly outperforms the two RF-regularization methods (using \textit{depth} and \textit{mtry}) in all but one dataset (breast-cancer). This is especially promising because HS is also the fastest and easiest method to implement, as it does not require refitting the RF. Moreover, \textit{hsRF tends to achieve its maximum performance with fewer trees} than RF without regularization; as a consequence, RF with HS is often able to achieve the same performance with an ensemble that is five times smaller, allowing us to achieve large savings in computational resources.

We also compare \textit{hsRF} to the predictive performance of BART, and observe that \textit{hsRF} and BART are comparable in terms of prediction performance. However, \textit{hsRF} is much faster to fit than BART (typically 10-15 times faster) and we can also apply HS to BART, see Appendix S3.5.

5. HS improves RF interpretations by simplifying and stabilizing them

In addition to improving predictive performance, HS reduces variance and removes sampling artifacts, resulting in (i) simplified boundaries, (ii) stabilized feature importance scores, and (iii) making it easier to interpret interactions in the model.

---

\[7\]This parameter is denoted \textit{mtry} in \textit{ranger} and \textit{max_features} in \textit{scikit-learn}.

\[8\]Typically \( \sqrt{p} \) for classification and \( p/3 \) for regression, where \( p \) is the number of features.

---

Fig[4] shows an example of simplification: smoothing decision boundaries. On the diabetes dataset (Smith et al., 1988), RF can achieve strong performance (AUC 0.733) even when fitted to only two features. When HS is applied to this RF, the performance increases (to an AUC of 0.787), but the decision boundary also becomes considerably smoother and less fragmented. Since these two features are the only inputs to the model, these smooth boundaries enable a user to identify much clearer regions for high-risk predictions. Appendix S4.1 shows boundary maps for all 8 classification datasets, showing that HS consistently makes boundaries smoother (these maps visualize the boundary with respect to the two features with the highest mean-decrease-impurity (MDI) score).

In models with many features, post-hoc interpretations, such as SHAP scores [Lundberg & Lee, 2017], can help a practitioner understand how a model makes its predictions. Fig[5] shows that HS improves the stability of SHAP scores. Stability is measured using the variance of SHAP values when models are fit to 100 random train-test splits of the breast-cancer dataset. This makes the model interpretations less sensitive to minor data perturbations and thus more trustworthy. Moreover, these improvements in stability persist even for datasets such as Heart, Diabetes, and Ionosphere, for which HS does not greatly improve prediction performance (see SHAP stability plots for all datasets.
Hierarchical Shrinkage

in Fig S12 and Fig S13. Hence, HS can improve the stability and interpretability of RF, even when it does not improve its predictive performance.

Figure 5. Comparison of SHAP plots learnt by Random Forests on the breast-cancer dataset before / after applying HS. HS displays lower variability across different data perturbations (without dropping in predictive performance), indicating enhanced stability.

Fig 6 shows an example investigating the SHAP scores across the breast-cancer dataset for a trained RF. After applying HS, the SHAP values for each feature often cluster much more nicely. Each cluster corresponds to a group of samples for which a feature contributes a similar amount to the predicted response, regardless of the value of other features. As a result, each cluster can be interpreted without taking into account feature interactions, simplifying the user’s interpretation. Since HS improves the model’s predictive performance, the clustered SHAP scores suggest that HS improves performance by regularizing some unnecessary interactions in the model, and makes the fitted function closer to being additive, which allows for simpler interpretations. Appendix S4.2 shows the SHAP plots for all 8 classification datasets, showing that HS consistently leads to more clustered SHAP values.

6. Discussion

HS is a fast yet powerful regularization procedure that can be applied to any tree-based model without changing its structure. In our experiments, HS never hurts prediction performance, and often leads to substantial gains in both predictive performance and interpretability. HS is partly motivated by [Tan et al., 2021]’s non-minimax-optimal generalization lower bounds for decision trees that predict using average responses over each leaf, pointing to a possible limitation of averaging. HS allows us to break this inferential barrier by pooling information from multiple leaves.

The work here naturally suggests many exciting future directions regarding the regularization of trees and RFs. First, replacing ridge regression with more sophisticated linear methods such as lasso or elastic net can result in other promising regularization methods. In addition, HS could improve other structured rule-based models, such as rule lists and tree sums (Tan et al., 2022).

Meanwhile, we have only scratched the surface of the relationships between regularization, robustness, and interpretability. Indeed, the connection between HS and ridge regression suggests the relevance of Donhauser et al. (2021)’s observations that ridge regression helps with robust generalization. Furthermore, we have seen that the simpler and smoother decision boundaries resulting from HS are more likely to generalize well. Hence, we conjecture that HS could improve the predictive performance of RF with respect to covariate shift. Moreover, the improved clustering and stability of SHAP scores after applying HS suggest that regularization via HS could improve the identification of important features, even when using alternative methods such as MDI.

Figure 6. Comparison of SHAP values for RF on the breast-cancer dataset before/after applying HS with features arranged by importance. Each point represents a single sample in the dataset. HS leads to more clustered SHAP values for each feature, reflecting less heterogeneity in the SHAP values of each feature.
Hierarchical Shrinkage

7. Acknowledgements

We gratefully acknowledge partial support from NSF TRIPODS Grant 1740855, DMS-1613002, 1953191, 2015341, IIS 1741340, ONR grant N00014-17-1-2176, the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370, NSF grant 2023505 on Collaborative Research: Foundations of Data Science Institute (FODSI), the NSF and the Simons Foundation for the Collaboration on the Theoretical Foundations of Deep Learning through awards DMS-2031883 and 814639, and a grant from the Weill Neurohub.

References

Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., and Rudin, C. Learning certifiably optimal rule lists for categorical data. arXiv preprint arXiv:1704.01701, 2017.

Asuncion, A. and Newman, D. Uci machine learning repository, 2007.

Athey, S. and Imbens, G. Recursive partitioning for heterogeneous causal effects. Proceedings of the National Academy of Sciences, 113(27):7353–7360, 2016.

Basu, S., Kumbier, K., Brown, J. B., and Yu, B. Iterative random forests to discover predictive and stable high-order interactions. Proceedings of the National Academy of Sciences, pp. 201711236, 2018.

Behr, M., Kumbier, K., Cordova-Palomera, A., Aguirre, M., Ashley, E., Butte, A., Arnaout, R., Brown, J. B., Preist, J., and Yu, B. Learning epistatic polygenic phenotypes with boolean interactions. bioRxiv, 2020.

Belgiu, M. and Drăguț, L. Random forest in remote sensing: A review of applications and future directions. ISPRS Journal of Photogrammetry and Remote Sensing, 114:24–31, 2016.

Boulesteix, A.-L., Janitza, S., Kruppa, J., and König, I. R. Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2(6):493–507, 2012.

Breiman, L. Random forests. Machine learning, 45(1):5–32, 2001.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. J. Classification and regression trees. Chapman and Hall/CRC, 1984.

Caruana, R. and Niculescu-Mizil, A. An empirical comparison of supervised learning algorithms. In Proceedings of the 23rd International Conference on Machine learning, pp. 161–168, 2006.

Caruana, R., Karampatziakis, N., and Yessenalina, A. An empirical evaluation of supervised learning in high dimensions. In Proceedings of the 25th International Conference on Machine learning, pp. 96–103, 2008.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785–794, 2016.

Chen, X. and Ishwaran, H. Random forests for genomic data analysis. Genomics, 99(6):323–329, 2012.

Chipman, H. A., George, E. I., and McCulloch, R. E. Bart: Bayesian additive regression trees. The Annals of Applied Statistics, 4(1):266–298, 2010.

Díaz-Uriarte, R. and De Andres, S. A. Gene selection and classification of microarray data using random forest. BMC bioinformatics, 7(1):3, 2006.

Donhauser, K., Tifrea, A., Aerni, M., Heckel, R., and Yang, F. Interpolation can hurt robust generalization even when there is no noise. Advances in Neural Information Processing Systems, 34, 2021.

Doshi-Velez, F. and Kim, B. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608, 2017.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. Least angle regression. The Annals of statistics, 32(2):407–499, 2004.

Evans, J. S., Murphy, M. A., Holden, Z. A., and Cushman, S. A. Modeling species distribution and change using random forest. In Predictive Species and Habitat Modeling in Landscape Ecology, pp. 139–159, Springer, 2011.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. Do we need hundreds of classifiers to solve real world classification problems? The Journal of Machine Learning Research, 15(1):3133–3181, 2014.

Friedman, J., Hastie, T., Tibshirani, R., et al. The Elements of Statistical Learning, volume 1. Springer Series in Statistics New York, 2001.

Friedman, J. H. Multivariate adaptive regression splines. The annals of statistics, pp. 1–67, 1991.

Friedman, J. H. Greedy function approximation: a gradient boosting machine. Annals of statistics, pp. 1189–1232, 2001.

Golub, G. H., Heath, M., and Wahba, G. Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics, 21(2):215–223, 1979. doi: 10.1080/00401706.1979.10489751. URL https://www.tandfonline.com/doi/abs/10.1080/00401706.1979.10489751.

Hooker, G. and Mentch, L. Bridging Breiman’s brook: From algorithmic modeling to statistical learning. Observational Studies, 7(1):107–125, 2021.

Klusowski, J. M. Universal consistency of decision trees in high dimensions. arXiv preprint arXiv:2104.13881, 2021.

Kuppermann, N., Holmes, J. F., Dayan, P. S., Hoyle, J. D., Atabaki, S. M., Holubkov, R., Nadel, F. M., Monroe, D., Stanley, R. M., Borgialli, D. A., et al. Identification of children at very low risk of clinically-important brain injuries after head trauma: a prospective cohort study. The Lancet, 374(9696):1160–1170, 2009.
Hierarchical Shrinkage

Letham, B., Rudin, C., McCormick, T. H., Madigan, D., et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *Annals of Applied Statistics*, 9(3):1350–1371, 2015. doi: 10.1214/15-aos848.

Lin, J., Zhong, C., Hu, D., Rudin, C., and Seltzer, M. Generalized and scalable optimal sparse decision trees. In *International Conference on Machine Learning*, pp. 6150–6160. PMLR, 2020.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pp. 4768–4777, 2017.

Mentch, L. and Zhou, S. Randomization as regularization: a degree of freedom explanation for random forest success. *arXiv preprint arXiv:1911.00190*, 2019.

Messenger, R. and Mandell, L. A modal search technique for predictive nominal scale multivariate analysis. *Journal of the American Statistical Association*, 67(340):768–772, 1972.

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

Murdoch, W. J., Singh, C., Kambier, K., Abbasi-Asl, R., and Yu, B. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, 2019. doi: 10.1073/pnas.1900654116.

Nash, W. J., Sellers, T. L., Talbot, S. R., Cawthorn, A. J., and Ford, W. B. The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. *Seu Fisheries Division, Technical Report*, 48:p411, 1994.

Olson, R. S., Cava, W. L., Mustashan, Z., Varik, A., and Moore, J. H. Data-driven advice for applying machine learning to bioinformatics problems. In *Biocomputing 2018: Proceedings of the Pacific Symposium*, pp. 192–203. World Scientific, 2018.

Oosfisky, J. D. The effects of exposure to violence on young children (1995). *Carnegie Corporation of New York Task Force on the Needs of Young Children; An earlier version of this article was presented as a position paper for the aforementioned corporation.*, 1997.

Pace, R. K. and Barry, R. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.

Pedregosa, F., Varoquaux, G. é. l., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011. URL [http://jmlr.org/papers/v12/pedregosa11a.html](http://jmlr.org/papers/v12/pedregosa11a.html)

Quinlan, J. R. Induction of decision trees. *Machine learning*, 1 (1):81–106, 1986.

Quinlan, J. R. *C4.5: programs for machine learning*. Elsevier, 2014.

Romano, J. D., Le, T. T., La Cava, W., Gregg, J. T., Goldberg, D. J., Ray, N. L., Chakraborty, P., Himmelstein, D., Fu, W., and Moore, J. H. Pmlb v1.0: an open source dataset collection for benchmarking machine learning methods. *arXiv preprint arXiv:2012.00058*, 2020.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. doi: 10.1038/s42256-019-0048-x.

Rudin, C., Chen, C., Chen, Z., Huang, H., Demenova, L., and Zhong, C. Interpretable machine learning: Fundamental principles and 10 grand challenges. *arXiv preprint arXiv:2103.11251*, 2021.

Sigillito, V. G., Wing, S. P., Hutton, L. V., and Baker, K. B. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, 10(3):262–266, 1989.

Singh, C., Nasseri, K., Tan, Y. S., Tang, T., and Yu, B. imodels: a python package for fitting interpretable models. *Journal of Open Source Software*, 6(61):3192, 2021. doi: 10.21105/joss.03192. URL [https://doi.org/10.21105/joss.03192](https://doi.org/10.21105/joss.03192).

Smith, J. W., Everhart, J. E., Dickson, W., Knowler, W. C., and Johannes, R. S. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the annual symposium on computer application in medical care*, pp. 261. American Medical Informatics Association, 1988.

Steadman, H. J., Silver, E., Monahan, J., Appelbaum, P. Robbins, P. C., Mulvey, E. P., Grisso, T., Roth, L. H., and Banks, S. A classification tree approach to the development of actuarial violence risk assessment tools. *Law and Human Behavior*, 24 (1):83–100, 2000.

Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P., and Feuston, B. P. Random forest: a classification and regression tool for compound classification and qsar modeling. *Journal of Chemical Information and Computer Sciences*, 43(6):1947–1958, 2003.

Tan, Y. S., Agarwal, A., and Yu, B. A cautionary tale on fitting decision trees to data from additive models: generalization lower bounds. *arXiv preprint arXiv:2110.09626*, 2021.

Tan, Y. S., Singh, C., Nasseri, K., Agarwal, A., and Yu, B. Fast interpretable greedy-tree sums (figs). *arXiv preprint arXiv:2201.11931*, 2022.

Wang, T. Gaining free or low-cost interpretability with interpretable partial substitute. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6505–6514. PMLR, 09–15 Jun 2019. URL [http://proceedings.mlr.press/v97/wang19a.html](http://proceedings.mlr.press/v97/wang19a.html)

Wright, M. N., Ziegler, A., et al. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software, 77*(i01), 2017.

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

S1. Data details

| Name               | Samples | Features | Class 0 | Class 1 | Majority class % |
|--------------------|---------|----------|---------|---------|------------------|
| Heart              | 270     | 15       | 150     | 120     | 55.6             |
| Breast cancer      | 277     | 17       | 196     | 81      | 70.8             |
| Haberman           | 306     | 3        | 81      | 225     | 73.5             |
| Ionosphere (Sigillito et al., 1989) | 351 | 34       | 126     | 225     | 64.1             |
| Diabetes (Smith et al., 1988) | 768 | 8        | 500     | 268     | 65.1             |
| German credit      |         |          |         |         |                  |
| Juvenile (Osofsky, 1997) | 3640 | 286      | 3153    | 487     | 86.6             |
| Recidivism         | 6172    | 20       | 3182    | 2990    | 51.6             |

Table S2. Classification datasets (extended).

| Name               | Samples | Features | Mean  | Std  | Min  | Max  |
|--------------------|---------|----------|-------|------|------|------|
| Friedman1 (Friedman, 1991) | 200     | 10       | 14.3  | 4.6  | 2.1  | 25.2 |
| Friedman3 (Friedman, 1991)  | 200     | 4        | 1.3   | 0.4  | 0.0  | 1.6  |
| Diabetes (Efron et al., 2004) | 442 | 10       | 152.1 | 77.0 | 25.0 | 346.0 |
| Geographical music    | 1059    | 117      | 0.0   | 1.0  | -1.5 | 5.9  |
| Red wine             | 1599    | 11       | 5.6   | 0.8  | 3.0  | 8.0  |
| Abalone (Nash et al., 1994) | 4177     | 8        | 9.9   | 3.2  | 1.0  | 29.0 |
| Satellite image (Romano et al., 2020) | 6435 | 36      | 3.7   | 2.2  | 1.0  | 7.0  |
| California housing (Pace & Barry, 1997) | 20640 | 8      | 2.1   | 1.2  | 0.1  | 5.0  |

Table S3. Regression datasets (extended).

S2. Additional simulations

In this section, we provide experimental details for our bias-variance trade-off simulation in Fig 2 as well as other simulations settings that we display below. Note that we reproduce Fig 2 in Fig S1 for ease of the reader.

**Experimental Design:** For Fig S1, we simulate data via a linear model \( y = \sum_{i=1}^{10} x_i + \epsilon \) with \( x \sim \text{Unif}(0, 1)^{10} \) and \( \epsilon \) being drawn from a gaussian and laplacian distribution for the left and right panel respectively with \( \sigma^2 = 0.01 \). In Fig S2, we simulate responses from a linear model with pairwise interactions \( y = \sum_{i=1}^{10} x_i + x_1x_2 + x_3x_6 + x_11x_{12} + \epsilon \). We used a training and test set of 500 samples, and computed the bias and variance over 100 independent draws of the training set. In all of our experiments, we fit CART and applied HS post-hoc while varying the number of leaves. The regularization parameter \( \lambda \) was chosen on the training set via 3-fold cross-validation.

![Figure S1. CART (HTS) test error rate for a linear model with gaussian (left)/laplacian (right) noise stays low as the number of leaves increases, whereas CART test error displays U-shaped bias-variance tradeoff as model complexity increases.](image-url)
Figure S2. CART (HTS) test error rate for a linear model with pairwise interactions and gaussian (left)/laplacian (right) noise stays low as the number of leaves increases, whereas CART test error displays U-shaped bias-variance tradeoff as model complexity increases.

S3. Experimental results

S3.1. C4.5 classification performance improves after adding HS

In this section, we display results on the classification data sets Table 1 for C4.5 before/after applying HS. The experimental details are provided in Sec 4.2.

Figure S3. Classification results for C4.5. HS significantly improves the performance even with very few leaves. Errors bars show standard error of the mean computed over 10 random data splits.

S3.2. GOSDT classification performance after adding HS

Here, we compare HS to GOSDT. We use the best tree found by GOSDT within one hour, since GOSDT fails to complete for many datasets within a reasonable timeframe (24 hours). To improve performance, we use a single pre-processing step of selecting the 5 most important features for each dataset (using RF feature importance). This step ensures that the resulting tree, fitted within a time limit of one hour, is not extremely shallow (number of leaves \(\leq 3\)) and that GOSDT does not fail to due to a memory error.

For GOSDT, we are unable to tune \(m\) exactly, and instead tune its cost-complexity regularization parameter. S4 shows the results, where we omit all the datasets for which all the resulting trees are extremely shallow. We note that GOSDT only outputs the root node for many data sets with regularization parameter larger than 0.
Hierarchical Shrinkage

Figure S4. HS performance across selected datasets. HS is as good as and sometimes slightly improves GOSDT.

S3.3. Leaf Based Results for Regression Datasets

We compare the performance of XG-Boost’s LBS to HS on the regression datasets in Table 1 with experimental details provided in Sec 4.3.

Figure S5. HS performs better than LBS for regression datasets in Table 1. Error bars show the standard error of the mean computed over 10 random data splits.

S3.4. Experimental Results for Deeper Trees

In this section, we investigate the performance of decision tree algorithms: (i) CART, (ii) C4.5, (iii) CART with CCP before/after applying HS while growing deeper trees for all the data sets in Table 1 and Table 1. We observe that the
performance of the baselines drop dramatically while the performance of HS continuously improves as the ratio of number of leaves to samples increases. This is explained by our connection of HS with ridge regression discussed on a supervised basis in [A].

![Image](image-url)

**Figure S6.** HS is able to improve (A) Classification and (B) regression predictive performance for both baselines: CART and CART with CCP for larger data-sets when using deeper trees by preventing overfitting. As shown by our connection to Ridge regression in Sec 3, HS is beneficial when the ratio of number of leaves to samples is large. Error bars show standard error of the mean computer over 10 random data splits.

### S3.5. BART

BART is a Bayesian “sum of trees” model, in particular the tree structure and leaf values are considered to be random variables, following a prior distribution. Conditioned on these random variables and the query point $x$, the response variable $y$ is assumed to follow a Gaussian distribution. The posterior distribution of the tree structures and leaf values is used for inference, where samples are generated via backfitting MCMC algorithm. Every single sample is a sum of trees function, and the final prediction is typically the average prediction over many such samples.

The tree growing algorithm in BART is fundamentally different from CART, in that it considers the mean response function, to be a random variable. In BART, the tree functions used for inference are samples from a posterior distribution, whereas in CART a tree is learned to minimize a certain objective. Another key difference is that BART shrinks the predictions to the grand mean, by centering the response variable and selecting a Gaussian prior with mean 0 for the leaf predictions. Therefore BART provides an interesting study case, both in terms a different tree growing method and a an alternative shrinkage method.

Our experimental setup is as follows:

First, a BART model is fitted using a single MCMC chain and otherwise default parameters using bartpy package. Then, to control for model complexity (in terms of the number of leaves), a single sample from the posterior distribution is obtained, from which the BART predicted values and tree structure/s are extracted. Using the tree structure/s we impute leaf values and then apply HS, to obtain the hSBART prediction.
Figure S7. Hierarchical Shrinkage (solid lines) applied post-hoc to BART trees, shows similar performance for classification (A), and regression (B) datasets in terms of AUC and $R^2$ respectively.

We hypothesize that the remarkably similar performance, for both regression and classification datasets, is due to BART’s tendency to construct relatively small trees, when the recommended parameters are used (Chipman et al., 2010).

S4. Experimental results for interpretability

S4.1. Decision boundaries

In this section, we plot the different decision boundaries learnt by a Random Forest model with 50 trees before/after applying HS for all the data sets built on the two most important covariates as measured by random forest feature importance. For instance, Fig S9(D) visualizes the decision boundary for the recidivism data set using the covariates: age and prior counts.
Figure S8. Comparison of decision boundary learnt by Random Forests on the smaller data sets before / after applying HS. Even though HS does not result in an increase in predictive performance, it learns a smoother, simpler decision boundary, resulting in improved interpretability and simplicity.
Hierarchical Shrinkage

Figure S9. Comparison of decision boundary learnt by Random Forests on the larger data sets before / after applying HS. Even though HS does not result in an increase in predictive performance, it learns a smoother, simpler decision boundary, resulting in improved interpretability and simplicity.
### S4.2. SHAP Plots

In this section, we investigate SHAP plots (Lundberg & Lee, 2017) for RF with and without HS. SHAP plots are a popular tool to explain black-box models such as RFs, and provide explanations by computing a shapley value for every sample and feature. We provide SHAP plots for RF with and without HS for every dataset in Table 1.

**Figure S10.** Comparison of SHAP plots learnt by Random Forests on the smaller data sets before / after applying HS. HS displays more clustered SHAP values for each feature, reflecting less heterogeneity in the effect of each feature on predicted response.
Figure S11. Comparison of SHAP plots learnt by Random Forests on the larger data sets before / after applying HS. HS displays more clustered SHAP values for each feature, reflecting less heterogeneity in the effect of each feature on predicted response.
S4.3. SHAP Variability plots

We investigate the stability of SHAP scores to data perturbations arising from different train/test splits for all classification data sets in Table 1. In order to do this, we randomly choose 50 held-out samples in each data set and measure the variance of their SHAP scores per feature across 100 different train-test splits for RF with and without HS. We then average the variance per feature across all 50 held-out samples and visualize them in the figures below. As seen in the figures below, we see that SHAP values for RF with HS are more stable to perturbations, thereby verifying their increased interpretability.

Figure S12. Comparison of SHAP plots learnt by Random Forests on different (small) datasets before / after applying HS. HS displays lower variability across different data perturbations, indicating enhanced stability.
Figure S13. Comparison of SHAP plots learnt by Random Forests on different (large) datasets before / after applying HS. HS displays lower variability across different data perturbations, indicating enhanced stability.

S5. Theory
In this section, we provide a proof of Theorem 1 and the heuristic arguments discussed in Sec 3.

S5.1. Proof of Theorem 1
Throughout this proof, we denote the left and right child of a node \( t_i \) by \( t_{i,L} \), \( t_{i,R} \). Further, we assume WLOG that the sample mean \( \hat{E}_{t_0} \{ y \} = 0 \). Then using the well known solution to ridge regression, we have that

\[
\hat{\beta}_\lambda = (\Psi(D_n)\Psi(D_n)^T + \lambda I)^{-1}\Psi(D_n)^Ty
\]

Using the orthogonality of the decision stumps and the norm of the features, we have that the transformed covariance matrix \( \Psi(D_n)\Psi(D_n)^T \) is a diagonal matrix with entries \( (\Psi(D_n)\Psi(D_n)^T)_{ii} = \mathcal{N}(t_i) \). Therefore, we have the following
Hierarchical Shrinkage

expansion for the $i$-th coordinate of $\hat{\beta}_\lambda$

$$\hat{\beta}_{\lambda,i} = \frac{\langle \psi_{t_0}(D_n), y \rangle}{N(t_i) + \lambda} = \frac{N(t_{i,L}) \sum_{y_i \in t_{i,L}} y_i - N(t_{i,R}) \sum_{y_i \in t_{i,R}} y_i}{\sqrt{N(t_{i,L}) N(t_{i,R})} (N(t_i) + \lambda)}$$

$$= \frac{\sqrt{N(t_{i,L}) N(t_{i,R})} \left( \hat{\mathbb{E}}_{t_{i,L}} \{ y \} - \hat{\mathbb{E}}_{t_{i,R}} \{ y \} \right)}{N(t_i) + \lambda}$$

$$= \frac{\sqrt{N(t_{i,L}) N(t_{i,R})} \left( \hat{\mathbb{E}}_{t_{i,L}} \{ y \} - \frac{N(t_i) \hat{\mathbb{E}}_{t_i} \{ y \}}{N(t_{i,R})} + \frac{\hat{\mathbb{E}}_{t_{i,L}} \{ y \} N(t_{i,L})}{N(t_{i,R})} \right)}{N(t_i) + \lambda}$$

$$= \frac{\sqrt{N(t_{i,L})^2 N(t_{i,R})}}{N(t_{i,R}) (N(t_i) + \lambda)^2} \left( \hat{\mathbb{E}}_{t_{i,L}} \{ y \} - \hat{\mathbb{E}}_{t_i} \{ y \} \right) \quad (8)$$

Then given a query point $x$ with leaf-to-root path $t_L \subset t_{L-1} \subset \cdots \subset t_0$, assume WLOG that each descendant is always the left child of its ancestor. Then using (8) and , the prediction at $x$ can then be expanded as follows

$$\hat{f}_\lambda(x) = \Psi(x)^T \beta_\lambda$$

$$= \sum_{l=0}^{L-1} \frac{N(t_{i,R}) \langle \psi_{t_i}(D_n), y \rangle}{\sqrt{N(t_{i,L}) N(t_{i,R})} (N(t_i) + \lambda)}$$

$$= \sum_{l=0}^{L-1} \frac{\sqrt{N(t_{i,L}) N(t_{i,R})} \left( \hat{\mathbb{E}}_{t_{i,L}} \{ y \} - \hat{\mathbb{E}}_{t_i} \{ y \} \right)}{N(t_i) + \lambda}$$

$$= \sum_{l=0}^{L-1} \frac{\hat{\mathbb{E}}_{t_{i,L}} \{ y \} - \hat{\mathbb{E}}_{t_i} \{ y \}}{1 + \lambda/N(t_i)}$$

We see that the equation above is precisely the same as that proposed for HS in [1].

S5.2. Heuristics for equation [7]

Consider the generative model

$$y = f(x) + \epsilon.$$  \hspace{1cm} (9)

where $\mathbb{E}\{\epsilon \mid x\} = 0$. Suppose for the moment that $f(x) = \beta^T x$ is linear, so that

$$\mathbb{E}\{y \mid t_L\} - \mathbb{E}\{y \mid t_R\} = \beta^T (\mathbb{E}\{x \mid t_L\} - \mathbb{E}\{x \mid t_R\}).$$  \hspace{1cm} (10)

Now further assume that for some $\beta_{min}$ and $\beta_{max}$, we have

$$\beta_{min} \leq |\beta_i| \leq \beta_{min}$$

for all $i$. Plugging these into (10) allows us to compute

$$\beta_{min}^2 \text{diam}(t)^2 \leq (\mathbb{E}\{y \mid t_L\} - \mathbb{E}\{y \mid t_R\})^2 \leq \beta_{max}^2 \text{diam}(t)^2.$$  \hspace{1cm} (11)

Next, for a given node $t$, let its side lengths be denoted by $l_j$, with corresponding measure $\mu(t) = \prod_{j=1}^d l_j$. Assuming that all the side lengths of $t$ are similar, we have that the diameter of the node $\text{diam}(t) \approx \mu(t)^{1/d}$. Furthermore, assuming that when a node $t$ is split into left and right children $t_L, t_R$, we split the node fairly evenly, such that the measure of $t_L, t_R$ are roughly equivalent. Then we have that the measure of the node $t$ is $\approx 2^{-\text{depth}(t)/d}$, which implies that $\text{diam}(t) \approx 2^{-\text{depth}(t)/d}$. Substituting this into (11) gives us the heuristic claimed in [7].

For a more general $C^1$ regression function $f$, note that the $C^1$ assumption implies that $f$ is approximately linear locally, i.e. when the nodes are small enough.