Slow-Growing Trees

Philippe Goulet Coulombe [1]

Abstract

Random Forest's performance can be matched by a single slow-growing tree (SGT), which uses a learning rate to tame CART's greedy algorithm. SGT exploits the view that CART is an extreme case of an iterative weighted least square procedure. Moreover, a unifying view of Boosted Trees (BT) and Random Forests (RF) is presented. Greedy ML algorithms' outcomes can be improved using either "slow learning" or diversification. SGT applies the former to estimate a single deep tree, and *Booging* (bagging stochastic BT with a high learning rate) uses the latter with additive shallow trees. The performance of this tree ensemble quaternity (Booging, BT, SGT, RF) is assessed on simulated and real regression tasks.

1. Introduction

Suppose you are a terrible golfer invited to play a twisted form of golf. Rules are as follows. You are allowed 20 strokes. Performance is evaluated by a distance between a reported position and the hole. Two radically different strategies are available to the dismal golfer. The first is to try a hole a in one 20 times and report the average position of the ball. By unbiasedness (despite high variance), the average shot should be close to the hole. A less flamboyant strategy is to do 20 small yet precise strokes (perhaps with the putter) and report the final position of the ball. The hole is the conditional mean, the golfer a greedy algorithm, the first strategy is in the spirit of Random Forest (RF), while the second is based on Boosted Trees (BT).

This paper proposes a unified view of BT and RF by adopting a simple taxonomy. The two algorithms differ in two main aspects: model structure (additive shallow trees vs one deep tree) and how they tame the imperfect greedy optimization path (small learning rate vs randomization/di-

versification). The first means that the space of potential predictive functions depends on the underlying model-building algorithm (e.g., additive vs multiplicative). The second is picking a strategy to successfully explore that space.

The model/regularization pairs are well known. BT fits an additive model and uses slow learning via its small learning rate ν. RF performs fast but randomized learning of one deep tree, and gets it right on average. To re-emphasize, the model-building path of additive shallow trees is traditionally regularized in one way, and that of one deep tree in another. Yet, there is no ex-ante reason for these strict associations: additive shallow trees could well be regularized by randomization/diversification and the one deep tree's variance could be reduced by constructing it slowly. While the former was already proposed in (Goulet Coulombe, 2020b) for other purposes, the latter remained until now a conceptual curiosity. This paper makes it operational by developing the missing corner in Figure 1's quaternity: a Slow-Growing Tree (SGT).

SGT works by recognizing that CART (Breiman et al., 1984) can be seen as an extreme case of an iterative weighted least square procedure. More precisely, when splitting a sample into two subsamples, it gives a weight of one to observations that satisfy the splitting rule, and 0 to others. Then, calculations determining the next split for that particular node are carried using observations with a weight of 1. SGT introduces a learning rate η such that observations not satisfying the splitting rule are allocated a weight of $1 - \eta$ rather than 0. Thus, whenever $\eta < 1$, those are not completely discarded – or at least, not by a single sweep of the greedy algorithm. Trivially, SGT collapses to CART for $\eta = 1$. Furthermore, it is easy to see how naturally shrinkage operates. A decreasing learning rate means that the children nodes' current average will be shrunk to its parent node's — by allowing the other children's observations to enter subsequent calculations with a non-negligible weight.

The new algorithm is evaluated against standard alternatives on simulated and real data sets. SGT behaves very similarly to RF (Breiman, 2001), with sometimes marginally better/worse performance, supporting a **slo**w **t**ree **h**ypothesis (SLOTH) to explain RF. More generally, it is in line with the conjecture that the two regularization strategies (diversification vs slow learning) are largely interchangeable, and

Table 1. A Tree Ensemble Quaternity

		Model Structure	
		Additive Shallow Trees	One Deep Tree
Regularizer	Slow Learning	Boosting	Slow Growing Tree
	Randomization + Averaging	Booging	Random Forest

the best choice will depend on the data set. Hence, the development of SGT is conceptually interesting *per se* since it permits to single out the effect of different regularization strategies on the *same model* – as highlighted in Table 1.

Empirically backing the SLOTH, it is found that when the true DGP is a tree, SGT, and RF performances match. This observation helps link back RF to arguments that certain boosting procedures with infinitesimal learning rates approximate the Lasso path (Friedman et al., 2001; Rosset et al., 2004) – i.e., a global solution to a high-dimensional problem. When it comes to SGT (and RF), the high-dimensional space of generated regressors just got wider, and which corners of it that get to be explored depend on the previous greedy steps. Finally, Booging (bagging stochastic gradient boosted trees with a high learning rate) can deliver gains over BT, challenging slow-learning as the de facto regularization choice for additive trees.

This paper is organized as follows. In section 2, I detail the main conceptual insights, present the SGT algorithm and discuss practical aspects. Sections 3 and Section 4 report results on simulated and real data sets, respectively. Section 5 concludes.

2. Ensembling Trees Revisited

BT is a greedy algorithm sequentially combining trees to minimize a loss function. The resulting prediction is

$$\hat{y}_i = \sum_{s=1}^{S} \nu \mathcal{T}_s(X_i)$$

where S, the number of steps (and trees) is a tuning parameter governing early stopping. Intuitively, its range of candidate values depends directly on η – smaller values of η's typically necessitate higher values of S. In practice, it is customary to tune both (Friedman, 2002). Very often, the lower η gets, the better generalization is (Friedman, 2001).

Inspired by the above, I propose an algorithm estimating (slowly) a *single* tree that can match (and sometimes beat) RF's performance. While the utilization of a learning rate comes very naturally in a Boosting context (or in a deep learning one Goodfellow et al. 2016), it is less clear how it should materialize when fitting a single deep tree. A useful observation is that CART is an extreme form of an iterative

weighted least squares (WLS) procedure. The proposed implementation leverages the WLS analogy to develop a less "radical" tree-building algorithm.

Consider the deceptively simple tree below, which is obtained after one recursion of CART.

Full Sample

$$A \equiv \{i|x_i < 0\} \quad B \equiv \{i|x_i \geq 0\}$$

The subsequent problem of finding k_A^* and c_A^* to further grow the tree on the A side is

$$\min_{k\in\mathcal{K},c\in\mathbb{R}} \left[\min_{\mu_1} \sum_{\{i|X_i^k\leq c\}} \omega_i^A (y_i - \mu_1)^2 \right. $$
$$\left. + \min_{\mu_2} \sum_{\{i|X_i^k>c\}} \omega_i^A (y_i - \mu_2)^2 \right] \quad (1)$$

where $\omega_i = I(i \in A)$. This can be generalized to

$$\omega_i = I(i \in A) + (1 - \eta)I(i \in B)$$

where $\eta \in (0, 1]$ is a learning rate. Trivially, those new weights collapse to those of CART when $\eta = 1$. Like ν in Boosting, $\eta < 1$ shrinks the new problem to the old one. This partial rejection of the greedy algorithm's proposed split desirably moderates the chosen optimization direction. This can also be seen as softly pruning the tree at each step s: with a decreasing η the mean in the current children node is shrunk back to full sample mean, partly annulling the earlier split. Of course, the idea here, like in Boosting, is that there will be many more such s's than when $\eta = 1$, allowing SGT to build a deeper tree where each specific "split" matters less on its own.

Algorithm 1 outlines SGT's recursive procedure. A few remarks are in order. First, a typical stopping rule for CART is either the minimal size of the node before attempting a last split, or a minimal size of the terminal nodes. Anyway this is set up, it is customary that trees are approximately fully grown with leaves containing between 1 and 5 observations. By construction, all nodes of SGT, no matter how deep within the tree they appear, all incorporate N observations.

Algorithm 1 Slow-Growing Tree

Input: Training data $[y_i \ X_i]$, test set predictors X_j, learning rate $\eta \in (0,1]$, maximal Gini coefficient $\bar{G}$

Initialize $\omega_i^0 = 1 \ \forall i$.

for l's such that $G_l < \bar{G}$ **do**

$$(k_l^*, c_l^*) = \underset{k \in \mathcal{K}, c \in \mathbb{R}}{\arg\min} \left[\min_{\mu_1} \sum_{\{i | X_i^k \leq c\}} \omega_i^l (y_i - \mu_1)^2 + \min_{\mu_2} \sum_{\{i | X_i^k > c\}} \omega_i^l (y_i - \mu_2)^2 \right]$$

Create 2 children nodes with $\omega_i^{l,+} = \omega_i^l (1 - \eta I(X_i^{k_l^*} \leq c_l^*))$ and $\omega_i^{l,-} = \omega_i^l (1 - \eta I(X_i^{k_l^*} > c_l^*))$.

end for

Return: $\hat{y}_j = \sum_{l=1}^{L} \left(w_j^l(X_j) \sum_{i=1}^{N} \omega_i^l y_i \right)$ where $w_j^l(X_j) = \frac{\omega_j^l(X_j)}{\sum_{l=1}^{L} \omega_j^l(X_j)}$

The barometer of in-sample learning is an increasingly unequal distribution of ω_i^l's. Thus, the stopping rule shall be formulated in terms of the imbalance of ω_i^l across i's for a given a leaf l. I use an upper bound $\bar{G}$ on the Gini *coefficient* $G_l \equiv \sum_{i=1}^{N} (\omega_i^l)^2$ (Gini, 1936). That bound is set between 0.2 or 0.3 in practice, which is extremely high (that is, the distribution is highly unequal with only a few observations having a noticeable weight) and is thought of approximating what a fully grown (overfitting) tree would be in this new environment. Considering smaller $\bar{G}$'s (like 0.05 or 0.1) is a form of early stopping which will be shown to pay off in low signal-to-noise environments in section 3.

Weights for a given terminal node is the product of all filters applied to get from the top of the tree to that particular leaf. This is, again, a generalization of the usual tree weights where filters (splitting rules) have the additional property of setting part of the weights directly to 0. In that spirit, the mean of that terminal node is a weighted average with weights ω_i^l. Finally, since the hard separation embedded in CART (observations go on to A or B but not both) is no more, a new data point j will almost surely fall in more than one leaf (depending on its X_j). Those within-leaf weighted averages ($\sum_{i=1}^{N} \omega_i^l y_i$) must be combined to get the final prediction. Taking naively the simple average would fail as leaves where j has a very small ω_j^l would get the same preponderance as those where ω_j^l dominates. $w_j^l(X_j) = \omega_j^l(X_j) / \sum_{l=1}^{L} \omega_j^l(X_j)$ adjusts for that. It is easy to see that is the $\eta = 1$ case where weights are dichotomous (and thus leaves do not overlap) that $w_j^l(X_j) = 1$ for a single leaf and zero for all others.

Lastly, here is a very technical, yet necessary, consideration. The fact that a split can be invoked twice (or as often as needed, really) gives rise to a particular situation that would not arise in CART: the distribution of ω_i in a node can revert back to the initial (flat) one. For instance, if we "split" using $x_i < 0$ in a node which unique previous split was based on $x_i < 0$, we get two new children nodes. The first is not problematic – we simply get twice the discriminating action

of $x_i < 0$, that is, a weight of 1 for observations that satisfy the rule, and $(1 - \eta)^2$ for those that do not. However, the second child node will have a flat distribution of weights because some observations got down weighted once by $(1 - \eta)$ in the first split, and all the remaining children get down weighted once $(1 - \eta)$ by the second split. Hence, they all have a weight of $(1 - \eta)$, which, after normalization, are the initialization weights. If left unattended, this creates an infinite loop of a tree within a tree. The solution is simple: such dead branches are trimmed out. Note that this is not problematic in SGT as the assigned observations are appearing elsewhere anyway (in CART, this would create a hole in the predicting function).

2.1. What is SGT estimating?

I start by reviewing a popular interpretation of what a low learning rate does to a linear boosting (or forward stagewise regression) problem. Then, I leverage that knowledge to propose an explanation of what SGT is estimating

2.1.1. BOOSTING AS A l_1-REGULARIZED PROBLEM

(Friedman et al., 2001) observe that a simpler form of boosting (without trees and observations reweighting) deliver coefficients paths indistinguishable to that of Lasso (Tibshirani, 1996). More precisely, provided a small enough learning rate, plotting $\hat{\beta}_{\text{Boosting}}(s)$ against an increasing s and $\hat{\beta}_{\text{Lasso}}(\lambda)$ for a decreasing λ gives rise to two identical figures. This relationship is further explored in (Rosset et al., 2004) where boosting is argued to be a computationally convenient approach to solve a (very) high-dimensional regression with l_1 regularization.[1] This exact equivalence hinges on $\hat{\beta}_{\text{Lasso}}(\lambda)$ paths being monotone since Boosting solves a monotone Lasso problem where coefficients are penalized according to their "arc-length" (Hastie et al., 2007).[2]

---

[1] Closely related, (Zhang & Yu, 2005) provide a consistency result for Boosting provided a small step size and early stopping.

[2] Unlike traditional Lasso, the monotone version cannot be cast as a global optimization problem. Nonetheless, monotone Lasso has a less wiggly regularization path and can provide gains

(Schapire, 2013) nuances those claims by pointing out that the l_1 argument applies to a very specific variant of Adaboost, and that the Lasso analogy is hard to reconcile with the frequent observation that a very large S will not substantially damage out-of-sample performance – which would certainly not happen for Lasso in a classic setup when letting $\lambda \to 0$. On the other hand, (Kobak et al., 2020) show that the optimal λ (for ridge) can be either negative or 0 in extremely high dimensions – which is where BT operates. This provides an avenue to conciliate the l_1 "explanation" of boosting and the empirical observation that the generalization sometimes does not decrease in s.[3]

2.1.2. SGT ESTIMATES THE TRUE LATENT TREE

As a last building block behind the SLOTH, I now show that SGT can be cast as a peculiar Boosting problem. Then, we can exploit (Rosset et al., 2004) and (Hastie et al., 2007)'s arguments that Boosting, under certain conditions, performs a high-dimensional Lasso – i.e., solving a *global* problem.

A symmetric 2-layers tree's prediction can be written as

$$
\begin{aligned}
\hat{y}_i = \; & I(x_i > 0)\left[\alpha_1 I(z_i > 0) + \alpha_2 I(z_i \le 0)\right] \\
& + I(x_i \le 0)\left[\gamma_1 I(q_i > 0) + \gamma_2 I(q_i \le 0)\right]
\end{aligned} \quad (2)
$$

where γ's and α's are within-node mean parameters. x, z and q are predictors. Define $d_{x,i}^+ = I(x_i > 0)$ as a regressor and the rest accordingly. Note that SGT can be obtained by letting $d_{x,i}^+ = \tilde{\eta} I(x_i > 0) + (1 - \tilde{\eta}) I(x_i \le 0)$ where $\tilde{\eta}$ is some normalized version of η. (2) is also an additive model

$$
\hat{y}_i = \theta_1 d_{x,i}^+ d_{z,i}^+ + \theta_2 d_{x,i}^+ d_{z,i}^- + \theta_3 d_{x,i}^- d_{q,i}^+ + \theta_4 d_{q,i}^- d_{q,i}^-. \quad (3)
$$

with "polynomials" of order 2. A tree with 3-layers would extend (3) to 8 polynomials of order 3. Finally, one can rewrite (3) using the identity $d_{z,i}^+ = 1 - d_{z,i}^-$ (and similarly for q) to obtain

$$
\begin{aligned}
\hat{y}_i = \; & d_{x,i}^+ \left[(\alpha_1 - \alpha_2) d_{z,i}^+ + \alpha_2\right] \\
& + d_{x,i}^- \left[(\gamma_1 - \gamma_2) d_{q,i}^+ + \gamma_2\right] \\
= \; & \beta_1 d_{x,i}^+ + \beta_2 d_{x,i}^+ d_{z,i}^+ + \beta_3 d_{x,i}^- + \beta_4 d_{x,i}^- d_{q,i}^+.
\end{aligned} \quad (4)
$$

As in (3), (4) is additive but now, it is explicit that "earlier" basis functions are not ejected from the model. Hence, the deep tree can also be cast within (Rosset et al., 2004) and (Hastie et al., 2007)'s $\hat{y}_i = h(X_i)'\beta$ form where $h \in \mathcal{H}$. $\mathcal{H}$ is the fixed space of *all possible basis expansions* for the one deep tree, and it is significantly larger than what additive

---

over its famous brother in high-dimensional environments where regressors are highly collinear (Hastie et al., 2007).

[3]This is closely related to modern "interpolating regime" or "double descent" arguments (Wyner et al., 2017; Belkin et al., 2019; Bartlett et al., 2020).

shallow trees can allow for. The exploration of $\mathcal{H}$ is made possible by relying on the past steps of the greedy algorithm to guide its search through higher-order polynomials. That is, as an implication of the tree structure, some part of "later" basis expansions is already fixed.

Using this representation, the intuition provided in (Rosset et al., 2004) directly applies, provided β's are monotone. This can be obtained by designing new basis functions using a rule which follows from a series of casual observations. First, it is noticed that one sweep of SGT's greedy algorithm updates two β's — that of the older basis expansion (like $d_{x,i}^+$) and that of the newly added one ($d_{x,i}^+ d_{z,i}^+$). Second, it does not matter for the span of the model whether we expand with $d_{x,i}^+ d_{z,i}^+$ or $d_{x,i}^+ d_{z,i}^-$. Third, given that those specific basis expansions amount to splitting a mean into two means, by construction, one of them will always be at least as high (in absolute value) as the parent node's mean. Thus, we can choose to use $d_{x,i}^+ d_{z,i}^+$ or $d_{x,i}^+ d_{z,i}^-$ so that the updated coefficient on $d_{x,i}^+$ (which is already in the active set of regressors) satisfies monotonicity.

As a result of the above, SGT is a high-dimensional l_1 regularized regression with an extremely large $\mathcal{H}$ which is explored at each s in a particular way. Naturally, that space is the space of trees of a certain dimension (bounded by the size of the data set). SGT is a computationally cheaper way of estimating the true latent tree, whose direct global optimization would be an extremely high-dimensional Lasso problem. Moreover, l_1 regularization in SGT's context can be straightforwardly interpreted as pruning: shrinking β_2 completely to 0 prunes the $x_i > 0$ side of the tree. Selecting both β_2 and β_4 to 0 prunes the 2-layers tree a single layer one. Additionally, early stopping takes a similar meaning.

Since RF is yet another way of taming CART, this understanding of SGT is transferable to RF. In (Goulet Coulombe, 2020b), it is argued, by an analogy to forecasting with nonlinear (recursive) time series models, that bagging brings an imperfect greedy procedure to extract the true conditional mean. The deeper reason is that taking an expectation over a series of *nonlinear* recursions gives a very different outcome from that of a series of recursions that take expectations as inputs. The former is what we hope CART would do, and the latter is what it actually does. Bagging allows simulating the right expectation (Goulet Coulombe, 2020b). The SLOTH is that this performs the *same* role as what we expect from the learning rate. A conjecture as to why a low η deliver similar results is that, by shrinking the children weights to those of the parents, this pushes the "updating" mapping towards the identity mapping, which is *linear*, making the expectation in recursion problem much milder. Simulations will back the SLOTH (i.e., the bagging-η equivalence) by showing that when the true DGP is a tree, SGT and RF provide near-identical performance.

2.2. Interpretation?

In RF, the visual inspection of underlying trees is doomed because there are usually 500 of them and they are all of substantial size. The former problem remains for BT, whereas the latter vanishes since trees are typically shallow. However, unlike RF, all those trees are fitted to a different target (the pseudo-residuals at step s), a new complication. Thus, it is customary to attempt the understanding of such models using variable importance measure, partial dependence plots and surrogate models (Molnar, 2019). Yet, none of those options provide the enviable clarity of inspecting a single tree. SGT provides a single tree. However, the utilization of the learning rate means giving up hard-thresolding, which implies that a single observation can belong at varying degrees to different leaves. Also, $\eta < 1$ allows for splitting rules to be invoked more than once. This gives rise to "hierarchy" of splits: some on which the greedy algorithm really insists, and those that it may have merely stumbled upon along the way. None of those complications prevents from plotting the corresponding tree. For instance, the fact that some observations are shared across leaves can be simply thought as tree obtained on an artificially augmented data set (N-wise). Rather, the core roadblock is that the resulting SGT is very large, making human inspection daunting. In sum, SGT opens new avenues for the interpretation of a highly performing model (matching in many ways RF), but exploring those is non-trivial and is kept for future work.

2.3. Hyperparameters and Computational Aspects

Like any tree-based model, computational demand increases in N and K. It depends highly on η: a smaller η necessarily increase the depth of SGTs, the same way ν necessitate a higher S. Hence, computational burden behaves similarly to that of fitting a deep tree on a very large data set (N-wise). Additionally, since we are recursively solving WLS problems rather than assigning 0-1 weights (à la CART), the size of samples used to pick (k_l^*, c_l^*) at the lowest layers of the SGT is the same as those of upper layers. Finally, with a lower $\bar{G}$ acting as an early-stopping rule, computations decrease in $\bar{G}$. To accelerate computations – which can be demanding for large N data sets, I let η increase marginally with depth so the algorithm avoid languishing with many leaves with a G_l slightly below $\bar{G}$. The rule of thumb is starting at $\eta = 0.1$ and increasing it by 0.01 with depth (that is, how many splits led to the current node) until an imposed plateau of 0.5 is reached. To further increase speed and add mild regularization à la stochastic gradient descent (Friedman, 2002), `mtry` =0.75 is used throughout.

2.4. Related Works

Related to SGTs are soft trees (Irsoy et al., 2012). They differ from SGT in many ways – goals and results. They consist in changing the hard-threshold rule of CART by a sigmoid function which will still allocate 0 and 1 weights to observations in the tails, but will have less drastic weight change around c^*. Hence, some observations will still be completely discarded on the basis of one greedy step, and as a result, soft trees still suffer from most of the CART ailments – like attaining a global optimum. The latter concern has been the subject of recent work (Bertsimas & Dunn, 2017; Hu et al., 2019; Norouzi et al., 2015; Blanquero et al., 2020) where recursive partitioning is retired in favor of global optimization techniques. While those methods will almost always beat CART, their performance usually lag behind that of RF in a sizable way (see (Bertsimas & Dunn, 2017) results on classification and those of (Blanquero et al., 2020) for regression).

2.5. *Booging*

Booging (for Bagging Boosting) is the application diversification-based regularization to additive shallow tree structure. It was originally proposed in (Goulet Coulombe, 2020b) to show that any randomized greedy algorithm (beyond RF) can be self-regularized, that is, perform implicit early stopping. In this paper's simulations and empirical work, it will be used again to show how Booging can rival BT with tuned learning rates – further demonstrating the virtues of the championed tree ensemble quaternity.

Booging consists of running BT with a relatively high learning rate (like $\nu = 0.25$) on many bootstrap samples. Clearly, Bagging is only useful here because each BT run becomes increasingly unstable with high ν's. Moreover, each run is perturbed by using Stochastic Gradient Boosting (Friedman, 2002) with a certain fraction (often 0.3 or 0.5) of randomly selected observations being used to construct trees. To reinforce diversification potential (especially when regressors are scarce), the Booging also uses data augmentation (i.e., include noisy carbon copies of X). For all implementation details, see (Goulet Coulombe, 2020b).

3. Simulations

I use simulated data from five standard data generating processes (DGPs) to evaluate when SGTs and Booging outperforms their counterparts relying on their traditionally associated regularization scheme.

Five versions of the two classic tree models (one deep tree, additive shallow trees) are considered. In Figure 1, each model is a row facet. Both are constructed in five different ways, each with their own associated color. First, we have the bagged and perturbed version, which corresponds to RF for one deep tree, and Booging for additive shallow trees. For RF, both `mtry` and the minimal node size are tuned by cross-validation. The "plain" versions of the two
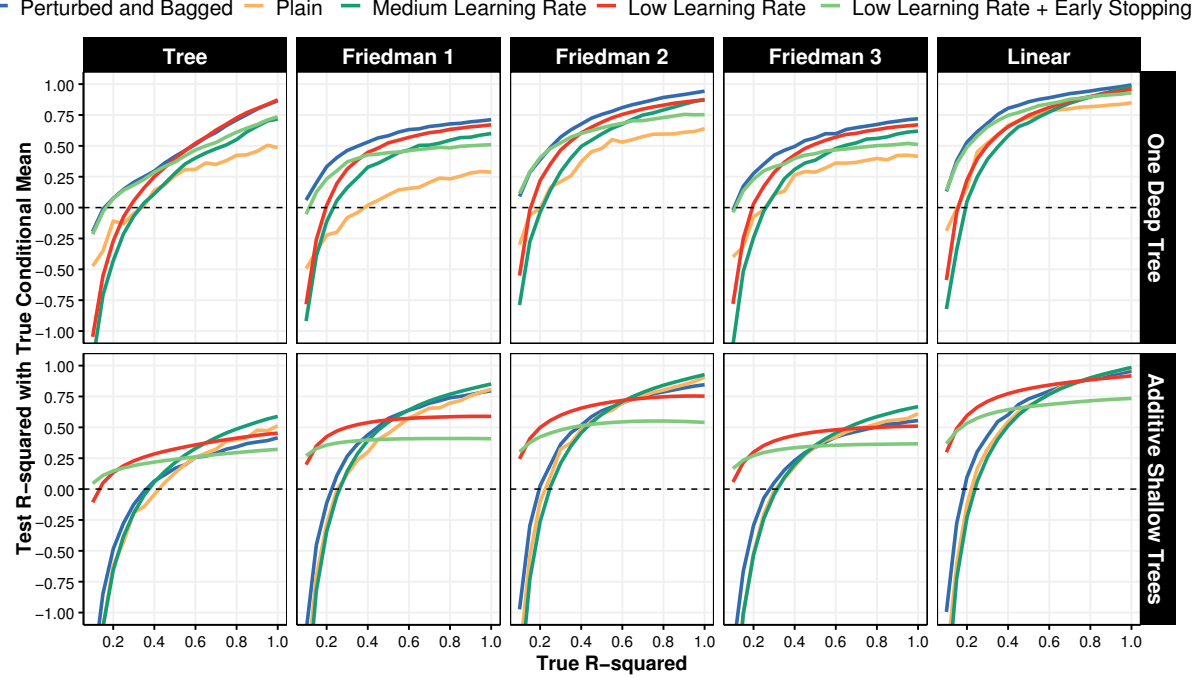
Figure 1. This plots the hold-out sample R^2 between the prediction **and the true conditional mean**. The level of noise is decreasing along the x-axis. Column facets are DGPs and row facets are "models". The y-axis is cut at -1 to favor readability because a few models go largely below it for the lowest true R^2 case.

structures are CART (with a tuned depth) and BT with a high ν (0.25) and a tuned number of trees. Third, I consider 3 versions with varying learning rates with one of them additionally incorporating early stopping. For the "one deep tree", these are SGTs with $\eta \in \{0.5, 0.1, 0.1\}$ and $\bar{G} \in \{0.25, 0.25, 0.05\}$. For additive trees, this is standard BT with $\nu \in \{0.1, 0.001, 0.001\}$ and $S \in \{1500, 1500, 750\}$.[4]

The five DGPs are identical to that of (Goulet Coulombe, 2020b). First, we have the "Tree" DGP which is generated using a CART algorithm's prediction function as a "new" conditional mean function from which to simulate. The depth of the tree is set by fixing the minimal node size so that true terminal nodes contain about $1/8$ of the sample on average. Then, we have Friedman 1, 2 and 3 (Friedman, 1991) as well as a linear model. The latter's conditional mean is the sum of five mutually orthogonal and normally distributed regressors. Each model is fed 10 predictors with the fraction of truly useful ones varying across DGPs (but is usually around 50%). In Figure 1, DGPs are column facets.

Finally, it is expected that the performance differential between the two regularization schemes depends not only on the DGP but also the signal-to-noise ratio. To explore such dependence, the performance is reported for "true" R^2's ranging (on Figure 1's x-axis) between 0.1 and 0.99. For

all simulations, $N = 100$ and the test set also has 100 observations.[5] The reported performance statistic is the mean R^2 between hold-out sample predictions and the true conditional mean for 30 simulations.

First, let us focus on the "Tree" column of Figure 1. The blue line of RF is visually absent, because it is perfectly matched by SGT($\eta = 0.1, \bar{G} = 0.05$) (light green) up until the true R^2 is about 0.45, and by SGT($\eta = 0.1, \bar{G} = 0.25$) (red) afterwards.[6] This simply means that when it comes to estimating a conditional mean corresponding to a true tree, SGT and RF coincides – as the SLOTH suggests. That is, slow learning and diversification are equally valid methods to tame CART's greedy algorithm.

In line with common wisdom for ν in BT (Friedman, 2001; Rosset et al., 2004), the SGT($\eta = 0.1, \bar{G} = 0.25$) line is strictly above that of ($\eta = 0.5, \bar{G} = 0.25$) (dark green). The strict dominance of a lower η is also verified for the remaining four DGPs. Finally, CART (light orange) is noticeably inferior to any SGTs, provided $\bar{G}$ would be tuned appropriately. In fact, it is interesting to notice how parallel are the light green and orange lines for all DGPs, with the latter always below the former by a sizable constant margin.

---

[4]As for other standard hyperparameters of Boosting-based models, the fraction of randomly selected observations to build trees at each step is 0.5, and the interaction depth of those trees is 5. ν is set to 0.25 for Booging.

[5]Relative performance results are close to identical when bumping N to 400.

[6]It is understood that tuning $\bar{G}$ would simply generate a line close to the upper envelope of SGT($\eta = 0.1, \bar{G} = 0.05$) and SGT($\eta = 0.1, \bar{G} = 0.25$), and thus (tuned) RF.

For DGPs Friedman 1,2 and 3, RF has a small edge over SGTs which is mostly apparent for high signal-to-noise ratio environments – RF performance is generally matched by $\text{SGT}(\eta = 0.1, \bar{G} = 0.05)$ for noisier setups. In the case of the linear DGP, RF performance is almost exactly matched by $\text{SGT}(\eta = 0.1, \bar{G} = 0.05)$ for all noise levels.

For the additive tree row, it is clearly visible that Booging implicitly performs early stopping, as the blue line closely follows the orange one, being marginally better or worse at times. Bringing ν down to 0.01 helps marginally, especially in the high true R^2 scenarios, whereas decreasing it even further to 0.001 pays off mostly when the true R^2 is low. As expected, the additive shallow trees, mostly underperforms with respect to the one deep tree when the true DGP is a tree. Another takeaway is that Booging, by using diversification with a high learning rate, is often comparable to a lower learning rate version of BT.

4. Empirics

I consider data sets that are standard in the literature (all from UCI repository or Kaggle) as well as 12 data sets representing different macroeconomic forecasting tasks.[7] The latter contrast sharply with the traditionally considered regression examples: it is time series data, the number of features is extremely high, the number of observations is low, and the signal-to-noise ratio is low when the data is properly transformed to induce stationarity. In short, it is a hostile environment where alternative regularization may yield much needed improvements.

The first 6 of those economic datasets are quarterly US data based on (McCracken & Ng, 2020) and the popular FRED database from the Federal Reserve Bank of St-Louis. The 3 targeted macroeconomic variables are GDP growth, unemployment rate change (UR), and CPI inflation (INF), which are predicted at horizons of 1 quarter and 2 quarters. The predictors matrix is based on (Goulet Coulombe, 2020a) and (Goulet Coulombe et al., 2020)'s recommendations for ML algorithms applied on stationary (see (McCracken & Ng, 2020) on that) macro data. Each data set has 212 observations (ending in 2014) and around 600 predictors.[8]

The last 6 data sets are monthly data from the UK using the newly built database from (Goulet Coulombe et al., 2021). The 4 variables are employment growth (EMP), hours worked change (HRS), and CPI inflation (INF). Those are predicted at horizons of 1 month and 3 months. For the latter, the target is the average growth rate (or change) over the next 3 months. The underlying data set is similar in nature to (McCracken & Ng, 2016) and transformations similar to what described above are performed. Each data

---

[7]For *California Housing*, a subset of 2000 observations is used.
[8]The features set varies because of a mild screening rule.

Table 2. Test set R^2's for all data sets and models

	Benchmarks			One Deep Tree		Additive Shallow Trees	
	AR	Lasso	Tree	RF	SGT	BT	Booging
Boston		0.67	0.58	0.88	0.86***	**0.90**	0.88
Abalone		0.51	0.31	**0.57**	0.55**	0.56	0.55
Auto		0.67	0.55	**0.71**	0.65***	0.68	0.66
Red Wine		0.35	0.25	**0.47**	0.42***	0.41	0.37***
Concrete		0.59	0.41	0.90	0.86***	**0.92**	0.90***
Fish Toxicity		0.55	0.36	**0.64**	0.64	0.63	0.62
NBA salary		0.49	0.22	**0.64**	0.60	0.54	0.52
CA Housing		0.69	0.51	**0.81**	0.77***	0.78	0.80
Ohio Housing		0.96	0.79	0.97	0.96	**0.98**	0.97
Airfoil Noise		0.51	0.36	0.92	0.85***	**0.93**	0.87***
US-GDP-1Q	0.27	0.27	0.09	0.36	0.34	0.35	**0.38**
US-GDP-2Q	0.17	−0.02	−0.25	0.17	0.02**	**0.19**	0.19
US-UR-1Q	0.53	0.45	−0.07	0.58	0.45**	0.62	**0.64**
US-UR-2Q	0.29	0.21	−0.25	0.41	0.36	**0.48**	0.48
US-INF-1Q	0.33	0.43	0.33	0.39	0.30	0.37	**0.39**
US-INF-2Q	0.22	0.10	0.45	0.28	**0.45****	0.12	0.27***
UK-EMP-1M	−0.05	0.03	−0.51	−0.07	−0.02	−0.04	**0.04**
UK-EMP-3M	0.06	0.15	−0.49	0.23	0.18	0.17	**0.27****
UK-HRS-1M	0.05	0.00	−0.14	0.03	−0.05	0.00	**0.08***
UK-HRS-3M	0.01	0.16	0.01	0.22	**0.27**	0.24	0.22
UK-INF-1M	0.73	0.13	0.32	0.08	0.25***	**0.26**	0.26
UK-INF-3M	0.74	−0.27	−0.03	0.05	0.14	0.03	**0.24****

Notes: This table reports out-of-sample R^2's for 24 data sets and different models, either standard or introduced in the text. For macroeconomic targets (the last 14 data sets), the set of benchmark models additionally includes an autoregressive model (AR) of order 2 for quarterly data and of order 6 for monthly data. Numbers in **bold** identify the best predictive performance among tree ensembles. Numbers in blue are the best performances over all models. For tree ensembles, t-test (and (Diebold & Mariano, 2002) tests for time series data) are performed to evaluate the statistical significance of the accuracy differential between the "standard" versions (either RF or BT) and the alternative regularization versions (SGT or Booging) . '*', '**' and '***' respectively refer to p-values below 10%, 5% and 1%.

set has about 260 observations (ending in 2019) and a number of predictors ranging from 250 to about 400.

The models of interest are the four members of the tree ensemble quaternity, i.e., BT, RF, Booging and $\text{SGT}(\eta = 0.1, \bar{G} = 0.25)$. BT, CART, Lasso, and RF are tuned via k-fold cross-validation. "Benchmarks" include Lasso and CART. For the macro data sets, they additionally include an autoregressive model of order 2 (AR), which is often very hard to beat (Kotchoni et al., 2019). I keep 70% of observations for training (and optimizing hyperparameters if needed) and use the remaining 30% to evaluate performance. For cross-sectional data sets, those observations are chosen randomly. For time series applications, the first $70\% \times N$ observations are the training set.

Table 2 report results. First, for standard data sets, RF and BT clearly dominates, which lend empirical support for the usual regularization/model pairs. A natural first question is whether SGT can closely match the RF stellar record. The answer is yes, SGT – with a single greedily optimized tree – trails behind by 0 to 7 points of R^2 percentages. Sometimes that difference is statistically significant (like for Red Wine,

a 5% gap significant at the 1% level), sometimes it is not (like for Boston, Fish Toxicity and Ohio Housing). Note that SGT is miles ahead of CART for every of those 10 data sets. Hence, unlike "optimal" trees (Bertsimas & Dunn, 2017; Blanquero et al., 2020), SGT results are in the ballpark of RF rather than merely improving upon CART.

Another interesting question is whether successes are mostly driven by the choice of regularizer or rather by that of the inherent model. The answer largely depends on the case under study. Results of NBA Salary, Red Wine, and Fish Toxicity are indicative of the one deep tree being more appropriate than shallow additive trees for those data sets (RF≻Booging and SGT≻BT). In contrast, results for California Housing suggest that opting for diversification over slow learning is the game changer (RF≻SGT and Booging≻BT).

The difficulties evoked earlier for macro data sets are clearly visible, with most R^2's being rather low. A first observation is that RF is *never* the best among all models, nor tree ensembles. Nevertheless, it still offers highly competitive results. When it comes to US-GDP at 1 quarter ahead, all tree ensembles are alike with a small edge for those based on diversification. When it comes to unemployment rate (UR) at horizon 1, Booging supplants all models. At horizon 2, it still dominates, but is now ex-aequo with BT. For UR at horizon 1, it is interesting to notice how diversification-based tree ensembles supplant all slow learning-based variants. For inflation (INF) at horizon 1, all tree ensembles give similar performance, with SGT lagging behind in a non-statistically significant fashion. However, those performances are narrowly eclipsed by Lasso. At horizon 2, Booging improves over BT by 15%, which is significant at the 1% level. SGT strongly improves over RF for the 2-quarter ahead forecast (17%, which is significant at the 5% level) and is the best by far among tree ensembles. In terms of MSE, it is matched by CART, but SGT has a mean absolute error that is lower by 7%. Thus, SGT generates the best 2-quarters ahead inflation forecasts overall.

Figure 2 investigate where SGT inflation forecasting gains originate from. At a horizon of half a year ahead, it is understood that many economic shocks cannot be captured by learning algorithms — one example is the 2008 downward spike attributed to an unprecedented collapse of oil prices. Rather, a good inflation forecast at such a range is a reliable estimate of the lower frequency movements of inflation, which is in itself a notoriously hard problem (Stock & Watson, 2008). SGT clearly provides that. While RF mostly overshoots, SGT matches nicely the slowly decreasing then increasing trend of inflation up to 2008. Moreover, the forecast is immune to the "missing disinflation puzzle" following the Great Recession. That is, unlike many macroeconomic and econometric models (Coibion & Gorodnichenko, 2015; Lindé & Trabandt, 2019), SGT does not wrongfully predict

an overly large decrease in inflation following 2008. Finally, SGT is more resilient than RF when it comes to the "missing inflation puzzle" occurring in the last years of the sample.
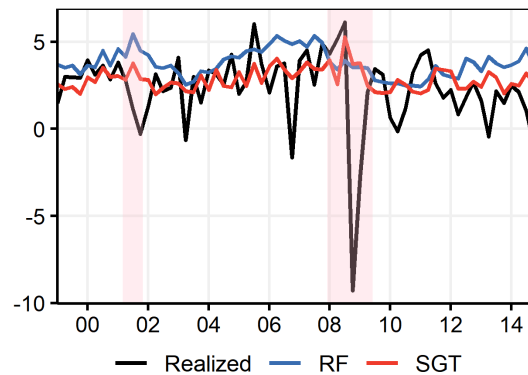

Figure 2. This plot compares the forecasts from RF and SGT for US inflation (from one quarter to the next, annualized) at a horizon of 2 quarter. Pink-shaded regions are recessions.

In the noisy environment of monthly time series, many gains are not statistically significant. There are exceptions. SGT's (R^2=25%) gain over RF (R^2=08%) for inflation one month ahead is statistically significant at the 5% level. However, this is far from being enough to capture UK inflation dynamics: SGT, BT and Booging all trail behind the more appropriate AR by a large margin.[9] There are less dismal cases – like that of hours worked 3 months ahead where SGT is the best model overall. This points out that SGT may be a potent tool to extract the faint signal out notoriously noisy series depicting real economic activity. Finally, Booging provides the best forecast for employment at both horizons, and hours worked at horizon 1. For EMP at the 3 months horizon, diversification clearly does better than slow learning with RF ≻ SGT and Booging ≻ BT. The edge of Booging over BT is statistically significant at the 1% level.

5. Conclusion

This paper proposes a tree ensemble taxonomy which ultimately helps understand what RF is doing. To make that claim, Slow-Growing Trees are introduced. It is shown that learning a single SGT delivers results often similar to that of RF. SGT allows linking back RF to boosting theory where the latter is argued to be similar to a very high-dimensional l_1-regularized regression problem. Simulations confirm that SGT and RF coincide when the true DGP is a tree. Applications to real data sets show that both SGT and Booging (additive trees regularized via diversification rather than a low learning rate) can provide gains over both BT and RF in difficult learning environments. Thus, when it comes to trees ensemble, regularization swapping can be beneficial.

---

[9]This motivates (Goulet Coulombe, 2020a) to develop a type of RF which can accommodate more easily (among other things) for the persistence of most macroeconomic time series.

References

Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 2020.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Bertsimas, D. and Dunn, J. Optimal classification trees. *Machine Learning*, 106(7):1039–1082, 2017.

Blanquero, R., Carrizosa, E., Molero-Rıo, C., and Morales, D. R. On sparse optimal regression trees. Technical report, Technical report, IMUS, Sevilla, Spain, https://www. researchgate. net . . . , 2020.

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

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. *Classification and regression trees*. CRC press, 1984.

Bühlmann, P. and Yu, B. Boosting with the l 2 loss: regression and classification. *Journal of the American Statistical Association*, 98(462):324–339, 2003.

Bühlmann, P., Hothorn, T., et al. Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4):477–505, 2007.

Coibion, O. and Gorodnichenko, Y. Is the phillips curve alive and well after all? inflation expectations and the missing disinflation. *American Economic Journal: Macroeconomics*, 7(1):197–232, 2015.

Diebold, F. X. and Mariano, R. S. Comparing predictive accuracy. *Journal of Business & economic statistics*, 20(1):134–144, 2002.

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

Freund, Y., Schapire, R. E., et al. Experiments with a new boosting algorithm. In *icml*, volume 96, pp. 148–156. Citeseer, 1996.

Friedman, J., Hastie, T., and Tibshirani, R. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:, 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.

Friedman, J. H. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.

Gini, C. On the measure of concentration with special reference to income and statistics. *Colorado College Publication, General Series*, 208(1):73–79, 1936.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Goulet Coulombe, P. The macroeconomy as a random forest. *arXiv preprint arXiv:2006.12724*, 2020a.

Goulet Coulombe, P. To bag is to prune. *arXiv preprint arXiv:2008.07063*, 2020b.

Goulet Coulombe, P., Leroux, M., Stevanovic, D., Surprenant, S., et al. Macroeconomic data transformations matter. Technical report, CIRANO, 2020.

Goulet Coulombe, P., Marcellino, M., and Stevanovic, D. Can machine learning catch the covid-19 recession? *CEPR Discussion Paper No. DP15867*, 2021.

Hastie, T., Taylor, J., Tibshirani, R., Walther, G., et al. Forward stagewise regression and the monotone lasso. *Electronic Journal of Statistics*, 1:1–29, 2007.

Hu, X., Rudin, C., and Seltzer, M. Optimal sparse decision trees. In *Advances in Neural Information Processing Systems*, pp. 7267–7275, 2019.

Irsoy, O., Yıldız, O. T., and Alpaydın, E. Soft decision trees. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 1819–1822. IEEE, 2012.

Kobak, D., Lomond, J., and Sanchez, B. The optimal ridge penalty for real-world high-dimensional data can be zero or negative due to the implicit ridge regularization. *Journal of Machine Learning Research*, 21(169):1–16, 2020.

Kotchoni, R., Leroux, M., and Stevanovic, D. Macroeconomic forecast accuracy in a data-rich environment. *Journal of Applied Econometrics*, 34(7):1050–1072, 2019.

Lindé, J. and Trabandt, M. Resolving the missing deflation puzzle. 2019.

Mason, L., Baxter, J., Bartlett, P. L., and Frean, M. R. Boosting algorithms as gradient descent. In *Advances in neural information processing systems*, pp. 512–518, 2000.

McCracken, M. and Ng, S. Fred-qd: A quarterly database for macroeconomic research. Technical report, National Bureau of Economic Research, 2020.

McCracken, M. W. and Ng, S. Fred-md: A monthly database for macroeconomic research. *Journal of Business & Economic Statistics*, 34(4):574–589, 2016.

Molnar, C. *Interpretable machine learning*. Lulu.com, 2019.

Norouzi, M., Collins, M., Johnson, M. A., Fleet, D. J., and Kohli, P. Efficient non-greedy optimization of decision trees. In *Advances in neural information processing systems*, pp. 1729–1737, 2015.

Rosset, S., Zhu, J., and Hastie, T. Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, 5(Aug):941–973, 2004.

Schapire, R. E. Explaining adaboost. In *Empirical inference*, pp. 37–52. Springer, 2013.

Stock, J. H. and Watson, M. W. Phillips curve inflation forecasts. Technical report, National Bureau of Economic Research, 2008.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Wyner, A. J., Olson, M., Bleich, J., and Mease, D. Explaining the success of adaboost and random forests as interpolating classifiers. *The Journal of Machine Learning Research*, 18(1):1558–1590, 2017.

Zhang, T. and Yu, B. Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579, 2005.