Weighted Splitting in Random Forests

Christian Hilscher

July 5, 2021

1 Introduction

2 Decision Trees and Random Forest

Unfortunately there is no one clear algorithm on how to build binary decision trees and random forests. Rather each paper has its own procedures and while mostly similar, they differ in some important aspects. For this reason we specify the exact tree building process and further aggregation into random forests.

2.1 Preliminaries

We start with our training data $\mathcal{D} = \{(\mathbf{X}_1, Y_1), ..., (\mathbf{X}_n, Y_n)\}$ where $\mathbf{X}_i \in [0,1]^d$ and $Y_i \in \mathbb{R}$ is a continuous response variable for $1 \leq i \leq n$. The j^{th} coordinate of the input matrix $\mathbf{X}$ is denoted by X_j. We assume $Y_i = m(\mathbf{X}_i) + \epsilon_i$ with $m(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ being an unknown regression function and ϵ_i is an $i.i.d$ error. The main goal is to estimate $m(\mathbf{x})$ and make predictions $\hat{Y}(\mathbf{x})$. The accuracy of the predictions will be determined by the mean squared error $\mathbb{E}[(\hat{Y}(\mathbf{X}) - m(\mathbf{X}))^2]$ which we seek to minimize.
Random forests are often used in setups with a large amount of input variables of which only a few actually determine the outcome variable Y. Therefore, we assume that $m(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ depends only on a small subset $\mathcal{S} < d$ features, which we also call strong features. On the other hand, all weak (noisy) variables $\{X_j : j \notin \mathcal{S}\}$ are independent of Y.

2.2 Decision Tree

A decision tree is a collection of recursive binary splits which aim to partition the data into more and more homogeneous subgroups. In other words, we apply the same process to smaller and smaller data until we arrive at a certain stopping criterion.
At each step, one variable and one split point are chosen which determine how the data will be partitioned. Consider the case where we arrive at node $\mathbf{t}$, splitting on variable X_j and s being the split point. The data will be separated into the two child nodes $\mathbf{t}_L = \{\mathbf{X} \in \mathbf{t} : X_j \leq s\}$ and $\mathbf{t}_R = \{\mathbf{X} \in \mathbf{t} : X_j > s\}$. This procedure is repeated until a specified stopping criterion.

The question arising now is how to choose the splitting dimension and the split point. Breiman et al. (1984) propose the CART (Classification and Regression Tree) algorithm which is the common way to construct decision trees. The ultimate goal is to partition the data into homogeneous subgroups, measured by the within-node variance. The CART algorithm therefore aims to reduce the variance of the response variable Y within a node by choosing the best splitting dimension and location.
The variance of a node $\mathbf{t}$ is given by

$$\hat{\Delta}(\mathbf{t}) = \frac{1}{N(\mathbf{t})} \sum_{\mathbf{X}_i \in \mathbf{t}} (Y_i - \hat{Y}_{\mathbf{t}})^2 \tag{1}$$

where $\hat{Y}_{\mathbf{t}}$ is the sample mean of the responses and $N(\mathbf{t})$ is the number of observations in node $\mathbf{t}$ respectively. To assess the quality of different split points as dimensions, we compare them using a

measure called *decrease in impurity*. The decrease in impurity for a generic variable X_j and split point s is given by

$$\hat{\Delta}(s;\mathbf{t}) = \hat{\Delta}(\mathbf{t}) - [\hat{P}(\mathbf{t}_L)\hat{\Delta}(\mathbf{t}_L) + \hat{P}(\mathbf{t}_R)\hat{\Delta}(\mathbf{t}_R)] \tag{2}$$

with $\hat{P}(\mathbf{t}_L) = \frac{N(\mathbf{t}_L)}{N(\mathbf{t})}$ and $\hat{P}(\mathbf{t}_R) = \frac{N(\mathbf{t}_R)}{N(\mathbf{t})}$ being the fractions of observations falling into the left and right child node. Maximizing (2) with respect to s then yields the split point which in turn partitions the data such that the resulting child nodes are as homogeneous as possible. Iterating over all input variables contained in $\mathbf{X}$ then allows to choose the splitting dimension X_j and respective s with the largest impurity reduction.
The estimator for a terminal node $\mathbf{t}$ is given by the sample mean of all observations falling into that node: $\hat{Y} = \bar{Y}_{\mathbf{t}}$.

While single decision trees have been empirically shown to have rather low biases, they do exhibit high variance. This behavior is especially pronounced when the stopping criterion is set in such way, that each terminal node contains only one observation. In other words, if the tree is fully grown, each terminal node is made up of one observation. Splitting up to that point makes the tree very unstable and when facing different errors the tree would look markedly different. Thus, while being a really local predictor, the high variance of the estimator is one drawback of this approach. One possibility to mitigate this problem is to make the tree stop via a pre-specified stopping criterion. Imposing that each terminal node has to contain at least $k > 1$ observations, leads to multiple observations in the terminal nodes and reduces the variance of the tree. With an increasing k, the tree becomes more and more shallow. Naturally, one faces another trade-off with this imposition: The higher k, the more observations in a terminal node and this makes the estimator less local. Especially with small k however, the reduction in variance more than makes up for a little bit larger bias such that in practice single decision trees are not grown fully.

2.3 Random Forest

Another remedy to reduce the variance of a decision tree while trying to keep the favorable property of low bias was introduced in Breiman (2001)

2.4 Connection to kNN-Estimator

Lin and Yeon (2006) show that a random forest is adaptive nearest neighbor estimator. The neighborhood of a point x_0 for example are all points which are in the same terminal node as x_0. The adaptive component comes from the fact that the number of neighbors can differ depending on the neighborhood. This feature allows the random forest to adapt to the regression function and exploit local changes. Looking at a decision tree through the lens of a kNN estimator gives us the benefit that we can study certain properties in a more familiar setting. Especially when trying to understand on which variables the decision tree splits the data, the perspective from the kNN estimator proves

insightful. Lin and Yeon show that a decision tree splits more often on variables which are "important", meaning that they explain a large share of the variance of the outcome variable. To intuitively understand their argument, assume the following simple setup

$$Y = g(\mathbf{X}) + \epsilon$$
$$g(\mathbf{X}) = \sum_{j \in \mathcal{S}} a_j X_j$$

Here, Y is a simple additive model of our strong variables and ϵ and i.i.d error term. Assuming further that all $X_j \sim U[0,1]$, the importance of a particular X_j on Y is given by $|a_j|$. Variables with a large $|a_j|$ influence the outcome variable Y more than those with a low value.

Consider the case that we want to estimate $\hat{Y}_0$ for a given $\mathbf{X}_0$ for the actual outcome Y_0. Then the goal is to characterize the neighborhood aiming to best approximate $g(\mathbf{X})$ around $\mathbf{X}_0$. This is done by choosing intervals for each X_j such that the area within all those intervals consists of n points and whose average constitutes our estimate $\hat{Y}_0$. Let's call the interval lengths for each variable q_j. Lin and Yeon (2006) show that in equilibrium, $q_j|a_j| = C \ \forall j$ for any constant C. In other words, in the optimal case, all variables have the same importance.
Turning to our example with $X_j \sim \mathrm{U}[0,1]$, this implies that variables with a high $|a_j|$ have shorter intervals q_j. Intuitively, this means that for variables which have a large effect on Y, the interval around $\mathbf{X_0}$ should be rather small since being far away from $\mathbf{X_0}$ would push our estimate too far away from the actual outcome Y_0. On the other hand, for variables with a low importance, we can allow for a wider interval. The optimality condition from Leon and Yin (2006) means that more important variables have shorter interval lengths in optimum.

We can now take these insights and apply them to our random forest setup. The neighborhood around $\mathbf{X}_0$ are now all observations which are in the same terminal node as $\mathbf{X}_0$. The side lengths q_j correspond to the interval of X_j which defines the terminal node. One interesting aspect of this example is that the optimal q_j can vary from node to node, showcasing the local adaptivity of the random forest estimator.
The other, for this exposition more relevant insight however, is that variables which have a high importance have smaller side lengths. To get smaller side lengths, the algorithm must split more often on those particular variables. Put differently, strong variables will be chosen more often to split on in a random forest than weak variables. This ties into the findings of Klusowski (2019), who finds a negative relationship between the selection frequency of X_j and the respective importance of variable j measured by its MDI (mean decrease in impurity).

The simple additive model above showcases intuitively why strong variables are more often chosen as splitting dimension. Klusowski's finding holds for a far more general class of regression functions, making it clear that the relative contribution of the variable j to the total variance of Y plays an important role for the number of times it is selected as splitting dimension.

2.5 Literature Overview

The theoretical properties of random forests have been studied quite intensively in the last couple of years. Decision trees and methods building on them are among the most widely used algorithms in the fields of supervised machine learning. Despite their broad usage, the exact theoretical characteristics have proved to be difficult to analyze.

It is especially the data-dependency of the splitting points which complicates the analysis of random forest estimators. For this reason, most of the literature has made adaptions to the original CART algorithm and to the way tree within a forest are constructed.

Conditions for consistency of data-independent estimators have been proposed by Stone (1977). Decision trees being partitioning estimators, the first attempts to characterize the theoretical properties aimed to describe the behavior of data-independent algorithms. Cutler and Zhao (2001) achieve this by first randomly choosing the splitting variable and afterwards randomly selecting the split point. These alterations make the estimator independent of the data insofar as that the probability of a split happening on a particular variable for example is a not influenced by the underlying data. Another way to abstract from the data was analyzed by Breiman (2004) where the splitting dimension is chosen randomly and the split point is always the median observation. This way the structure of the tree is again data-independent and consistency can be shown relatively easily.

Biau (2012) was the first to show consistency results for an algorithm which more closely resembles the original random forest procedure and is data-dependent. The split points are determined by their respective decrease in impurity. In Biau's setup, the split points do not maximize the empirical decrease in impurity but rather the asymptotic one. For the consistent estimation of these asymptotic values a second dataset is needed however, since using the same data for estimation and locating the split points would lead to inconsistencies. Building on these results, Scornet et al. (2015) are able to prove consistency for a data-dependent random forest, albeit only for additive models. The restriction of allowing only an additive structure can be traced back to the influential paper by Lin and Jeon (2006) who reformulate the random forest as an adaptive nearest neighbor estimator. This way they can show that the variable selection frequency, that is the number of times a specific variable was selected to split on, is related to its "importance". The more of variation in Y in a specific neighborhood X_j can explain, the more frequently will the split happen on variable j. In an additive model the influence of one variable only depends on that variable itself since there are not interactions terms, which in turn render the analysis of the process more amenable.

Klusowski (2019) adds to the literature by proving consistency for a more general class of response surfaces which go beyond the additive structure used in Scornet et al. (2015). Overall the study of theoretical properties of the random forest algorithm has proved challenging and all the adaptions to the original algorithm prevents simple comparisons of approaches.

3 Concepts

3.1 MDI

3.2 Partial Dependence Function

It is useful to have a coherent method of measuring the importance a specific variable X_j has on the output Y. Since partial dependence plots usually only depict the relationship between two variables, we follow Klusowski (2019) by introducing a conditional partial dependence function. Recall that we assume $Y_i = m(\mathbf{X}_i) + \epsilon_i$. The idea behind the conditional partial dependence function is to look at the influence of one specific X_j while ignoring the others. For this let

$$\bar{F}_j(x_j; \mathbf{t}) = E[Y \mid \mathbf{X} \in \mathbf{t}, X_j = x_j] \tag{3}$$

where $\bar{F}_j(x_j; \mathbf{t})$ is the partial dependence function for variable j conditional on being in node $\mathbf{t}$. One can also think of () as a solution to a least squares approximation of $m(\mathbf{X})$ as a function of only X_j within node $\mathbf{t}$. For later purposes it will be beneficial to additionally define the mean centered partial dependence function

$$\begin{aligned} \bar{G}_j(x_j; \mathbf{t}) &= \bar{F}_j(x_j; \mathbf{t}) - E[Y \mid \mathbf{X} \in \mathbf{t}] \\ &= E[Y \mid \mathbf{X} \in \mathbf{t}, X_j = x_j] - E[Y \mid \mathbf{X} \in \mathbf{t}] \end{aligned} \tag{4}$$

which is the partial dependence function demeaned by the average of all observation within the respective node.

This concept now allows us to reformulate our definition of strong and weak variables. Since weak variables $\{X_j : j \notin \mathcal{S}\}$ are independent of Y, by law of iterated expectations, (3) is equal to $E[Y \mid \mathbf{X} \in \mathbf{t}]$. This in turn implies that the demeaned partial dependence function $\bar{G}_j(x_j; \mathbf{t})$ is zero for all weak variables. In contrast, strong variables, have an effect on Y and thus the demeaned partial dependence function is not necessarily 0. Note however, that here we concern ourselves with the population parameters. In a finite sample analysis, it is indeed the case that because of the error term, we can have $\bar{G}_j(x_j; \mathbf{t}) \neq 0$ even for weak variables. This will fact play an important part in the later analysis when constructing a decision tree.

3.3 End-cut-property

Central to the CART algorithm and thus decision trees and random forests is the splitting rule. The default way of determining split points is by maximising the decrease in impurity as described in section XYZ.

When looking for the optimal split point on a noisy variable, the CART algorithm has the tendency to split at the extreme of the feature space. *End-cut-property* or *end-cut-preference* (ECP) was first mentioned and documented by Breiman et al. (1984). When a variable is noisy, it does not have an effect on Y and therefore the decrease in impurity is asymptotically zero everywhere. When looking for a split point in practise however, the largest decrease in impurity can often be found at the edges by cutting off the first or last couple of observations. The reasoning behind this is that the more observations in either node, the closer will the sample variance of that child-node become to the variance in the parent node and thus lowering the decrease in impurity. Theorem 11.1 in Breiman et al. (1984) book formally describes this mechanism.

3.4 Drawbacks of ECP

The literature up to now regarded ECP mostly as a negative effect of the CART algorithm. Splitting at the edges of the feature space produces two very unbalanced nodes: a tiny amount of observations in one node and the rest in the other. The main problem with this behavior is that the observations in the small node do not constitute a good local predictor. Even in an asymptotic setting where $n \to \infty$, it is not guaranteed, that the number of observations in *every* node also approaches infinity. Since splitting on the edge leaves only a couple of observations in one of the resulting child-nodes, the number of observations in that node can stay fairly small and the predictor then does not converge to its true value.

Researchers have found several ways to mitigate this problem. The most straight forward approach is the one also adopted in Athey and Wager (2018). They impose a fraction α which is the minimum fraction of leaves that need to be in any child-node. The resulting trees are then called α-regular and as n grows, so does the number of observations in each node. Scornet (2015) imposes that at least k observations have to be in each node and this way forces prevents nodes at the top of the tree from becoming too small. Denil, Matheson and De Freitas (2014) employ slightly different procedure. For a dimension j they randomly draw five observations and then only consider those points between the smallest and the largest of the five observations as possible split points. Compared to the first approach, here it does not seem as if the number of observations in each node goes to infinity. The restrictions however force the CART algorithm to search for the largest decrease of impurity in the middle and both papers show that this leads to the overall number of observations in every node to increase as n grows.

All of these approaches have the same goal: making sure that splits do not happen at the extremes of the feature space while keeping as much from the original CART algorithm as possible. From a theoretical point of view, ECP is seen as a problem because it hinders the analysis in an asymptotic setting. In applied work it is mainly unwanted because it leads to bad predictors for those observations

which end up in those small nodes. All major software packages have therefore implementations of the approaches described above and the default values of all of them are such that splits at the edges are discouraged.

3.5 Potential Benefits

The verdict on end-cut-preference is not unanimous however, as Ishwaran (2014) highlights multiple benefits of the end-cut-preference exhibited by the CART algorithm. The main argument is built upon the local adaptivity of random forests. The ECP leads the child-nodes to be very imbalanced in size with the majority of the sample ending up in one child-node just a few observations in the other one. The advantage now comes from the preservation of most of the sample. Since the split on the noisy variable is not informative, only a few observations are "lost" while almost all other observations are used in splits further down the tree. This way the decision tree can recover from a bad split along a noisy variable. Ishwaran also mentions that ECP can occur for strong variables in regions where the parent-node is situated in a region of the feature space where the signal is relatively low. Especially in random forests where for each node only a subset of variables is considered as splitting dimension, the possibility of considering only variables with low signal increases, making the ability to recover from bad splits even more important. In simulation studies as well as with real world datasets, Ishwaran finds that algorithms possessing the ECP are in almost all setups superior to the ones without end-cut-preference.
Another fairly trivial situation where ECP is beneficial is when one faces outliers in the data. In such a case cutting at the edges is naturally what one would like the algorithm to do. A priori it is however unclear whether outliers in the data constitute genuine outliers or whether they happen to have large error terms. Ideally one has an algorithm which can pick up these differences and adapt accordingly.

4 Weighted Splitting

As early as in Breiman et al. (1984) a weighted splitting rule was proposed to mitigate the ECP of the standard CART procedure. The idea is to modify the splitting rule such that it takes the location of the split points into account and penalizes splits towards the edges of the feature space. Klusowski (2019) mentions that one could adapt the splitting rule and instead maximize

$$\begin{aligned} \Delta_\alpha(s;\mathbf{t}) &= [4P(\mathbf{t}_L)P(\mathbf{t}_R)]^\alpha \ \Delta(s;\mathbf{t}) \\ &= \omega \ \Delta(s;\mathbf{t}) \end{aligned} \tag{5}$$

where $\Delta(s;\mathbf{t})$ is the decrease in impurity which is maximized in the standard CART algorithm. $P(\mathbf{t}_L)$ and $P(\mathbf{t}_R)$ are the probabilities that an observation ends up in the left and right node and $\omega \leq 1$. These quantities naturally depend on the location of the split point. In case of a split in at the median of the observations, they are equal and the weight is maximized at $\omega = 1$. The more imbalanced the nodes become, the smaller the weight will be. The parameter α acts as a regularizer which governs how severely imbalances and thus split points at the edges are penalized.

Interestingly, (5) reveals a trade-off between the decrease in impurity and the node balancedness. When maximizing Δ_α, a split near the edges is likelier if the associated decrease in impurity is large. For small decreases in impurity on the other hand, the weighted splitting favors splits more in the middle.

5 Theoretical Results

Most of the theoretical results build upon the framework laid out by Klusowski (2019) since its results are the most general ones up to now. Adding the weight into the maximization problem makes some alterations to his proofs necessary but the overall framework is still valid.

5.1 Importance of MDI

References

[1] Random forests and adaptive nearest neighbors. *Quarterly Publications of the American Statistical Association 101*, 474 (June 2006), 578–590.

[2] Biau, G. Analysis of a random forests model. *Journal of Machine Learning Research 13*, 38 (2012), 1063–1095.

[3] Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.

[4] Breiman, L. Consistency for a simple model of random forests. Tech. rep., 2004.

[5] Breiman, L., Friedman, J., Stone, C., and Olshen, R. *Classification and Regression Trees*. Taylor & Francis, 1984.

[6] Cutler, A., and Zhao, G. Pert - perfect random tree ensembles. *Computing Science and Statistics* (2001), 497.

[7] Denil, M., Matheson, D., and de Freitas, N. Narrowing the gap: Random forests in theory and in practice. In *International Conference on Machine Learning* (2014).

[8] Ishwaran, H. The effect of splitting on random forests. *Machine Learning 99*, 1 (Apr. 2015), 75–118.

[9] Klusowski, J. Analyzing CART. *arXiv e-prints* (June 2019), arXiv:1906.10086.

[10] Klusowski, J. Sparse learning with cart. In *Advances in Neural Information Processing Systems* (2020), pp. 11612–11622.

[11] Scornet, E. On the asymptotics of random forests. *Journal of Multivariate Analysis 146* (2016), 72–83. Special Issue on Statistical Models and Methods for High or Infinite Dimensional Spaces.

[12] Scornet, E., Biau, G., and Vert, J.-P. Consistency of random forests. *The Annals of Statistics 43*, 4 (Aug 2015).

[13] Stone, C. J. Consistent Nonparametric Regression. *The Annals of Statistics 5*, 4 (1977), 595 – 620.

[14] Wager, S., and Athey, S. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association 113*, 523 (2018), 1228–1242.