Note on Random Forests

In this note I provide a mathematical formulation of Random Forests (RF). I follow the notation of Hastie and Tibshirani's textbook.

Random Forest is simply a group of trees. Hence, I start with trees and then show how one can build a RF. I focus on regression trees.

Fitting a tree, in its simplest form, is a form of fitting a piecewise constant function. The way Classification and Regression Tree (CART) algorithm does that is by partitioning the covariate space into rectangle regions and estimating a simple average of the variable of interest in each region.

Suppose we are given a dataset (y_i, x_i) for $i = 1, ..., N$, where each $x_i \in \mathbb{R}^P$. Ideally, we would like to partition the covariate space into M regions $R_1, ..., R_M$ and fit a constant $\hat{c}_m$ to each region, which (for squared-error loss) would simply be given by the average of y_i's in that region, that is,

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i, \tag{1}$$

where $N_m = \#\{x_i \in R_m\}$. Hence, our fit would simply be the sum of regional averages:

$$\hat{f}(x) = \sum_{m=1}^{M} \hat{c}_m I(x \in R_m). \tag{2}$$

However, this is not computationally feasible for interesting problems. The CART algorithm makes three simplifications to make this problem computationally tractable. First, the partitioning will proceed in a greedy fashion, that is, we partition the covariate space step by step. Second, in each of these steps, only binary partitions are considered. Finally, it is allowed to partition only at one of the P covariates at a time.

The algorithm works as follows. At each step, we focus on one of the regions, and decide which of the variables to select for binary splitting and where to split it, known as splitting variables and split points (threshold). Specifically, we search for the splitting variable j and split point s that would optimize some objective function, for example the sum of standard deviations in each of the produced subregions, that is

$$\min_{j,s} \left(\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right), \tag{3}$$

where $R_1(j, s)$ and $R_2(j, s)$ are the half-planes produced as a result of binary splitting,

$$R_1(j, s) = X | X_j \leq s \quad \text{and} \quad R_2(j, s) = X | X_j > s. \tag{4}$$

Again, under squared-error loss, the inner minimization problem of (2) is solved by

$$\hat{c}_1 = average(y_i | x_i \in R_1(j, s)) \quad \text{and} \quad \hat{c}_2 = average(y_i | x_i \in R_2(j, s)). \tag{5}$$

Given a variable, it is simple to identify the best split point. By going over all variables, we will then have to choose a variable-(best)split pair which minimizes the above objective function. Hence, the above simplifications allow us to obtain splitting variables and corresponding split points. This is done for each of the produced regions, usually until maximum allowed depth of the tree is not reached or until some threshold for the objective function is not met. Alternatively, it is possible to regularize the depth of the tree with the regularization strength typically chosen by cross-validation.

A simple example is plotted bellow with two regression trees of length 2 and 5 fitted to one-dimensional artificial data.

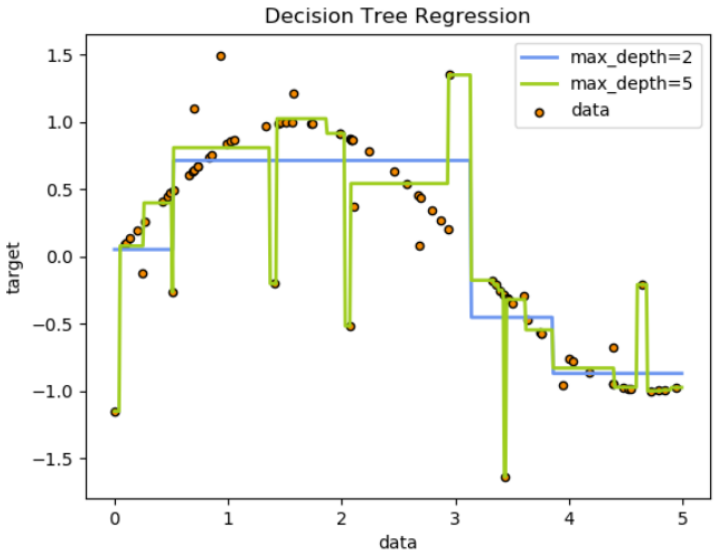

Figure 1: Regression Trees

The fitted functions do not look impressive. However, it is possible to greatly improve the fit by introducing randomness into the model, and this is how a random forest works. The randomness is twofold:

1. **Randomness in data** (bagging): instead of fitting a single regression tree to whole dataset, we bootstrap samples and fit trees to those samples.

2. **Randomness in variables**: when fitting a decision tree to a bootstrap sample, at each node we randomly select a set of possible variables to choose a split variable from. This might seem very counterintuitive at first. The goal of such restriction is to diminish the correlation between regression trees. The lower the correlation between the trees, the stronger are the benefits of aggregation.

RF fit is simply the average of many regression trees, each built on a bootstrap sample with the set of variables for selection restricted randomly at each node. That is, denoting an individual tree model as T_b, RF's prediction is given by the bagged average of trees,

$$\hat{f}_{RF}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \tag{6}$$