Continuous outcomes with BART: Part 1

Robert McCulloch
Arizona State University

Rodney Sparapani
Medical College of Wisconsin

Abstract

This short article delves into the BART prior with respect to continuous outcomes and the **BART** R package implementation.

Keywords: Bayesian Additive Regression Trees.

1. Introduction to Bayesian Additive Regression Trees

Bayesian Additive Regression Trees (BART) arose out of earlier research on Bayesian model fitting of an outcome to a single tree (Chipman, George, and McCulloch 1998). In this era from 1996 to 2001, the excellent predictive performance of ensemble models became apparent (Breiman 1996; Krogh and Solich 1997; Freund and Schapire 1997; Breiman 2001; Friedman 2001; Baldi and Brunak 2001). Instead of making a single prediction from a complex model, ensemble models make a single prediction which is the summary of the predictions from many simple models. Generally, ensemble models have nice properties, e.g., they do not suffer from over-fitting (Kuhn and Johnson 2013). Like bagging (Breiman 1996), boosting (Freund and Schapire 1997; Friedman 2001) and random forests (Breiman 2001), BART relies on an ensemble of trees to predict the outcome; and, although, there are similarities, there are also differences between these approaches.

BART is a Bayesian nonparametric, sum of trees method for continuous, dichotomous, categorical and time-to-event outcomes. Furthermore, BART is a black-box, machine learning method which fits the outcome via an arbitrary random function, f, of the covariates. So-called black-box models generate functions of the covariates which are so complex that interpreting the internal details of the fitted model is generally abandoned in favor of assessment via evaluations of the fitted function, f, at chosen values of the covariates. As shown by Chipman, George, and McCulloch (2010), BART's out-of-sample predictive performance is generally equivalent to, or exceeds that, of alternatives like lasso with L1 regularization (Efron, Hastie, Johnstone, and Tibshirani 2004) or black-box models such as gradient boosting (Freund and Schapire 1997; Friedman 2001), neural nets with one hidden layer (Venables and Ripley 2013) and random forests (Breiman 2001). Over-fitting is the tendency to overly fit a model to an in-sample training data set at the expense of poor predictive performance for unseen out-of-sample data. Typically, BART does not over-fit to the training data due to the regularization tree-branching penalty of the BART prior, i.e., generally, each tree has few branches and plays a small part in the overall fit yielding desirable properties. Essentially, BART is a Bayesian nonlinear model with all the advantages of the Bayesian paradigm such as posterior inference including point and interval estimation. Conveniently, BART naturally

scales to large numbers of covariates and facilitates variable selection; it does not require the covariates to be rescaled; neither does it require the covariate functional relationship, nor the interactions considered, to be pre-specified.

2. Continuous outcomes with BART

In this section, we document the analysis of continuous outcomes with the **BART** R package. This requires delving into the details of the BART prior itself and the corresponding arguments to the `wbart` function.

2.1. The BART prior

Underlying this methodology is the BART prior. The BART prior specifies a flexible class of unknown functions, f, from which we can gather randomly generated fits to the given data via the posterior. Let the function $g(\boldsymbol{x}; T, M)$ assign a value based on the input $\boldsymbol{x}$; T representing the structure of a binary tree, including interior decision rules as branches and terminal nodes as leaves; and $M = \{\mu_1, \ldots, \mu_L\}$, the parameter values associated with the L leaves one of which will be the output. T is composed of decision rules of the form $x_j \leq c$ which means branch left and $x_j > c$, branch right; or terminal leaves where it stops. The function, $f(\boldsymbol{x})$, is a sum of H trees, i.e., $f(\boldsymbol{x}) = \sum_{h=1}^{H} g(\boldsymbol{x}; T_h, M_h)$ where H is "large", let's say, 50, 100 or 200.

For a continuous outcome, y_i, we have the following BART regression on the vector of covariates, $\boldsymbol{x}_i$: $y_i = \mu_0 + f(\boldsymbol{x}_i) + \epsilon_i$ where $\epsilon_i \overset{\text{iid}}{\sim} \mathrm{N}\big(0,\ w_i^2\sigma^2\big)$; the unknown random function, f, and the error variance, σ^2, follow the BART prior expressed notationally as $(f, \sigma^2) \overset{\text{prior}}{\sim}$ BART and μ_0 is a known constant which centers y. The w_i are known standard deviation weights which you can supply with the argument `w` that is only available for continuous outcomes; the unit vector is the default. The centering parameter, μ_0, can be specified via the `fmean` argument where the default is taken to be $\bar{y}$.

BART is a Bayesian nonparametric prior, but it is best understood as a bundle of priors. Using the Gelfand-Smith shorthand bracket notation for the abstract specification of random variable distributions (Gelfand and Smith 1990), we represent the BART prior as $[T_h]\,[\mu_{hl}|T_h]\,\left[\sigma^2\right]$ where $[T_h]$ is the tree prior, $[\mu_{hl}|T_h]$ is the leaf prior and $\left[\sigma^2\right]$ is the error variance prior.

The tree prior: $[T_h]$. There are three priors for T_h which govern whether the tree branches grow or are pruned. The first is a regularity prior dictating the probability of an interior branch, event B, given the branch tier, t, is: $\mathrm{P}[B|t] = \mathrm{base}(1+t)^{-\mathrm{power}}$. You can specify these prior parameters with arguments, but the following defaults are highly recommended: `base=0.95` and `power=2`; for a detailed discussion of these parameters, see Chipman *et al.* (1998). Note that this prior penalizes branch growth, i.e., in prior probability, the default number of branches will likely be 1 or 2. Next, there is a prior dictating the choice of a splitting variable conditional on a branch event B which defaults to uniform probability $1/P$ where P is the number of covariates; however, you can specify a Dirichlet prior which is more appropriate if the number of covariates is large (Linero 2016) which we will return to below. Given a branch event, B, and a variable chosen, x_j, the last tree prior selects a cut point, c, within the range of observed values for x_j; this prior is uniform.

We represent the probability of variable selection via the sparse Dirichlet prior as

$[s_1, \ldots, s_P] \overset{\text{prior}}{\sim} \text{Dirichlet}(\theta/P, \ldots, \theta/P)$ which is specified by the argument `sparse=TRUE` while the default is `sparse=FALSE` for uniform $1/P$. The prior parameter θ can be fixed or random; the default `theta=0` means random, but fixed can be specified by supplying a positive number to the argument `theta`. Random θ is specified via $\frac{\theta}{\theta+\rho} \overset{\text{prior}}{\sim} \text{Beta}(a,\ b)$ where the parameter ρ can be specified by the argument `rho` (which defaults to `NULL` representing P; provide a value to over-ride), the parameter b defaults to 1 (which can be over-ridden by the argument `b`) and the parameter a defaults to 0.5 (which can be over-ridden by the argument `a`); `a=0.5` induces a sparse posture and `a=1`, non-sparse or dense.

If covariates are supplied via a data frame (rather than a matrix) with the argument `x.train`, then factors with more than two levels can be specified along with continuous variables. The factors are transformed into dummy variables with each factor in their own group while each continuous variable is a group all by itself (similarly, factors with two levels are a group with a single dummy variable). By specifying `sparse=TRUE`, these groups are handled with a sparse Dirichlet prior for grouped variables (Yuan and Lin 2006; Linero and Yang 2017). Suppose we have P groups each with K_j variables, then the probability of selecting a particular variable is $u_{jk} = s_j t_{jk}$ where $[s_1, \ldots, s_P] \overset{\text{prior}}{\sim} \text{Dirichlet}(\theta/P, \ldots, \theta/P)$ and $\left[t_{j1}, \ldots, t_{jK_j}\right] \overset{\text{prior}}{\sim} \text{Dirichlet}(\omega/K_j, \ldots, \omega/K_j)$. The specification of the θ prior is as above. The prior parameter ω is fixed and the default specification is set by the argument `omega=1`.

The leaf prior: $[\mu_{hl}|T_h]$. Given a tree, T_h, there is a prior on its leaf values, $\mu_{hl}|T_h$ and we denote the collection of all leaves in T_h by $M_h = [\mu_{h1}, \ldots, \mu_{hL_h}]$. Suppose that $y_i \in [y_{\min}, y_{\max}]$ for all i and $\left[\mu_{1(i)}, \ldots, \mu_{H(i)}\right]$ are the leaf output values from each tree corresponding to the vector of covariates, $\boldsymbol{x}_i$. If $\mu_{h(i)}|T_h \overset{\text{iid}}{\sim} \text{N}\left(\mu_\mu,\ \sigma_\mu^2\right)$, then $\text{E}[y_i|\boldsymbol{x}_i] = \mu_i \sim \text{N}\left(\mu_0 + H\mu_\mu,\ H\sigma_\mu^2\right)$. We choose values for μ_μ and σ_μ which are solutions to the system of equations created by the $1-\alpha/2$ confidence interval: $y_{\min} = \mu_0 + H\mu_\mu - |z_{\alpha/2}|\sqrt{H}\sigma_\mu$ and $y_{\max} = \mu_0 + H\mu_\mu + |z_{\alpha/2}|\sqrt{H}\sigma_\mu$, i.e., $\mu_\mu = \frac{y_{\max} - \mu_0 + y_{\min} - \mu_0}{2H}$ and $\sigma_\mu = \frac{y_{\max} - y_{\min}}{2|z_{\alpha/2}|\sqrt{H}}$. Since y is centered around μ_0, the solution for μ_μ will generally be near zero so we set it to zero. Therefore, we arrive at $\mu_{hl} \overset{\text{prior}}{\sim} \text{N}\left(0,\ \left[\frac{\tau}{2k\sqrt{H}}\right]^2\right)$ where $\tau = y_{\max} - y_{\min}$. So, the prior for μ_{hl} is informed by the data, y, but only weakly via the extrema, $y_{\min}$ and $y_{\max}$. The parameter k calibrates this prior as follows.

$$
\begin{aligned}
\mu_i &\sim \text{N}\left(\mu_0,\ \left[\frac{\tau}{2k}\right]^2\right) \\
\text{P}[y_{\min} \le \mu_i \le y_{\max}] &= \Phi(k) - \Phi(-k) \\
\text{Since P}[\mu_i \le y_{\max}] &= \text{P}\left[z \le 2k\frac{y_{\max} - \mu_0}{\tau}\right] \approx \text{P}[z \le k] = \Phi(k) \\
\text{Similarly P}[\mu_i \le y_{\min}] &= \Phi(-k)
\end{aligned}
$$

The default value, $k = 2$, corresponds to μ_i falling within the extrema with 0.95 probability. Alternative choices of k can be supplied via the `k` argument. We have found that values of $k \in [1, 3]$ generally yield good results. Note that k is a good candidate parameter for choice via cross-validation.

The error variance prior: $\left[\sigma^2\right]$. The prior for σ^2 is the conjugate scaled inverse chi-square distribution, i.e., $\nu\lambda\chi^{-2}(\nu)$. We recommend that the degrees of freedom, ν, be from 3 to 10 and the default is 3 which can be over-ridden by the argument `sigdf`. The lambda parameter

can be specifed by the `lambda` argument which defaults to `NA`. If `lambda` is unspecified, then we determine a reasonable value for λ based on an estimate, $\hat{\sigma}$, (which can be specified by the argument `sigest` and defaults to `NA`). If `sigest` is unspecified, the default value of `sigest` is determined via linear regression or the sample standard deviation: if $P < N$, then $y_i \sim \mathrm{N}\left(\boldsymbol{x}_i'\hat{\boldsymbol{\beta}},\ \hat{\sigma}^2\right)$; otherwise, $\hat{\sigma} = s_y$. Now we solve for λ such that $\mathrm{P}\left[\sigma^2 \leq \hat{\sigma}^2\right] = q$. This quantity, q, can be specified by the argument `sigquant` and the default is 0.9 whereas we also recommend considering 0.75 and 0.99. Note that the pair (ν, q) are good candidate parameters for choice via cross-validation.

Other important arguments for the BART prior. We fix the number of trees at H which corresponds to the argument `ntree`. The default number of trees is 200; as shown by Bleich, Kapelner, George, and Jensen (2014), 50 is also a reasonable choice and cross-validation could be considered. The number of cutpoints is provided by the argument `numcut` and the default is 100. The default number of cutpoints is achieved for continuous covariates; however, discrete covariates which have fewer than 100 values will necessarily have fewer cutpoints. For continuous covariates, the cutpoints are uniformly distributed by default, or generated via uniform quantiles if the argument `usequants=TRUE` is provided.

2.2. Posterior samples returned

The number of MCMC samples discarded for burnin is specified by the `nskip` argument and the default is 100. The number of MCMC samples returned is specified by the `ndpost` argument and the default is 1000. Returning every l^{th} value, or thinning, can be specified by the `keepevery` argument which defaults to 1, i.e., no thinning. You can also thin some returned values, but not others. The following arguments default to `ndpost`, but can be over-ridden as needed.

- `nkeeptrain` : number of f draws to return corresponding to `x.train`
- `nkeeptest` : number of f draws to return corresponding to `x.test`
- `nkeeptestmeam` : number of f draws to use in computing `yhat.test.mean`
- `nkeeptreedraws` : number of tree ensembles to return

2.3. Typical use case

Typically, when calling the `wbart` function to analyze some data, many of the arguments can be omitted since the default values are adequate for most purposes. However, there are certain common arguments which are either always needed or frequently provided. The `wbart` (`mc.wbart`) function is for serial (parallel) computation. The outcome `y.train` is a vector of numeric values. The covariates for training (validation, if any) are `x.train` (`x.test`) which can be matrices or data frames containing factors; in the display below, we assume matrices for simplicity.

```
set.seed(99)
post <- wbart(x.train, y.train, x.test, ..., ndpost=M) or
post <- mc.pbart(x.train, y.train, x.test, ..., ndpost=M, mc.cores=2, seed=99)
```

Input matrices: `x.train` and, optionally, `x.test`: $\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix}$ or $\boldsymbol{x}_i$

`post`, of type `wbart`, which is essentially a list

`post$yhat.train` and `post$yhat.test`: $\begin{bmatrix} \hat{y}_{11} & \cdots & \hat{y}_{N1} \\ \vdots & \cdots & \vdots \\ \hat{y}_{1M} & \cdots & \hat{y}_{NM} \end{bmatrix}$ where $\hat{y}_{im} = f_m(\boldsymbol{x}_i)$

The columns of `post$yhat.train` and `post$yhat.test` represent different covariate settings and the rows, the `M` draws from the posterior.

Often it is impractical to provide `x.test` in the call to `wbart` due to the number of predictions considered or all the settings to evaluate are simply not known at that time. To allow for this common problem, the **BART** package returns the trees encoded in an ASCII string, `treedraws$trees`, and provides a `predict` function to generate any predictions needed. Note that if you need to perform the prediction in some later R instance, then you can save the `wbart` object returned and reload it when needed, e.g., save with `saveRDS(post, 'post.rds')` and reload, `post <- readRDS('post.rds')` . The `x.test` input can be a matrix or a data frame; for simplicity, we assume a matrix below.

```
pred <- predict(post, x.test, mc.cores=1, ...)
```

Input: `x.test`: $\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_Q \end{bmatrix}$ or $\boldsymbol{x}_h$

`pred` is a matrix: $\begin{bmatrix} \hat{y}_{11} & \cdots & \hat{y}_{Q1} \\ \vdots & \vdots & \vdots \\ \hat{y}_{1M} & \cdots & \hat{y}_{QM} \end{bmatrix}$ where $\hat{y}_{hm} = f_m(\boldsymbol{x}_h)$

References

Baldi P, Brunak S (2001). *Bioinformatics: the machine learning approach.* MIT Press, Cambridge, MA.

Bleich J, Kapelner A, George EI, Jensen ST (2014). "Variable selection for BART: an application to gene regulation." *The Annals of Applied Statistics*, **8**(3), 1750–1781.

Breiman L (1996). "Bagging predictors." *Machine learning*, **24**(2), 123–140.

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

Chipman HA, George EI, McCulloch RE (1998). "Bayesian CART Model Search." *Journal of the American Statistical Association*, **93**(443), 935–948.

Chipman HA, George EI, McCulloch RE (2010). "BART: Bayesian Additive Regression Trees." *Annals of Applied Statistics*, **4**, 266–98.

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

Freund Y, Schapire RE (1997). "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of computer and system sciences*, **55**(1), 119–139.

Friedman JH (2001). "Greedy function approximation: a gradient boosting machine." *Annals of Statistics*, **29**, 1189–1232.

Gelfand AE, Smith AF (1990). "Sampling-based approaches to calculating marginal densities." *Journal of the American statistical association*, **85**(410), 398–409.

Krogh A, Solich P (1997). "Statistical mechanics of ensemble learning." *Physical Review E*, **55**, 811–25.

Kuhn M, Johnson K (2013). *Applied Predictive Modeling*. Springer, New York, NY.

Linero AR (2016). "Bayesian regression trees for high dimensional prediction and variable selection." *Journal of the American Statistical Association*, (<doi:10.1080/01621459.2016.1264957>).

Linero AR, Yang Y (2017). "Bayesian regression tree ensembles that adapt to smoothness and sparsity." *arXiv preprint arXiv:1707.09461*.

Venables WN, Ripley BD (2013). *Modern applied statistics with S-PLUS*. Springer Science & Business Media.

Yuan M, Lin Y (2006). "Model selection and estimation in regression with grouped variables." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **68**(1), 49–67.

Affiliation:

Rodney Sparapani rsparapa@mcw.edu
Division of Biostatistics, Institute for Health and Equity
Medical College of Wisconsin, Milwaukee campus