Bayesian Additive Regression Trees

Pietro Marini

May 2022

Abstract

A basic explaination of the main Bayesian ensemble model for regression analysis: Bayesian Additive Regression Trees, starting from the Bayesian objects that compose it and up to the Bayesian backfitting algorithm that informs the prior with the draws of the response variable. We compare its performance with boosting and bagging models, the most widely used ensembles of regression trees. In conclusion we explore an R package with improved efficiency thanks to parallelization and we use it to predict patients' length of stay, a fundamental variable for hospitals operational efficiency. We briefly highlight the solidity of the model from an Explainable AI perspective by outlining, as clear as possible, the foundations of the theoretical guidelines and empirical calibrations needed for specifying the model parameters. This is becoming essential considering the impact of automated systems in the XXI century and the lack of awareness of the decision-making processes underlying many important aspects of our lives.

Contents

1 Introduction

Nowadays there is a wide variety of regression models beyond linear regression for simple structure relationships or polynomial and treed regression for non-linear relationships. In particular, recent advances have brought the attention to ensemble models, consisting of weak learners combined with boosting, bagging or with the Bayesian approach used in the Bayesian Additive Regression Trees model by [Chipman et al., 2010] (BART by CGM10 henceforth).
Briefly, BART is a multivariate additive model built by a summation of weak learners. These submodels are dimensionally adaptive multivariate regression trees, and they are fitted through posterior sampling, with a Bayesian backfitting MCMC algorithm. Indeed, after the prior over the tree parameter space has been informed by the data, the posterior is explored with the Metropolis-Hastings algorithm. This avoids the computationally expensive exhaustive computation of the posterior for all the trees and makes it possible to select a model that is quite close to absolute optimality. Effectively, BART is a Bayesian nonparametric model because the cardinality of the set of regression tree submodels would be infinite if there existed an infinite dataset. The advantages of using this model are: (i) its robustness to high-dimensionality of the predictor space X; (ii) robustness to non-linear structure of the problem in some regions of X; (iii) its ability to account for interaction effects; (iv) very good interpretability-performance trade-off.
We are going to estimate the patients length of stay (LOS), very useful to enhance operational workload efficiency and quality of care in hospitals.

2 Ensembles of Regression Trees: A review

2.1 Literature

The first regression problem solved by Gauss and Legendre was prediction of orbits of astronomical objects around the sun. The first publication of the method of least squares is attributed to Legendre [Merriman, 1877], soon followed by Gauss. The term Regression was coined by Francis Galton [Pearson, 1930] and it dates back to the end of the 19^{th} century, when he created the Galton machine illustrated

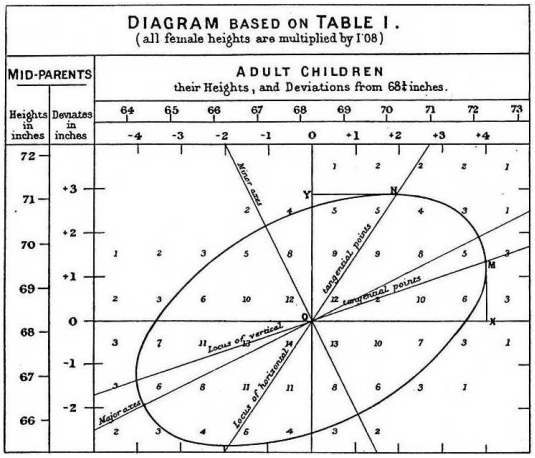


Figure 1: Galton correlation diagram

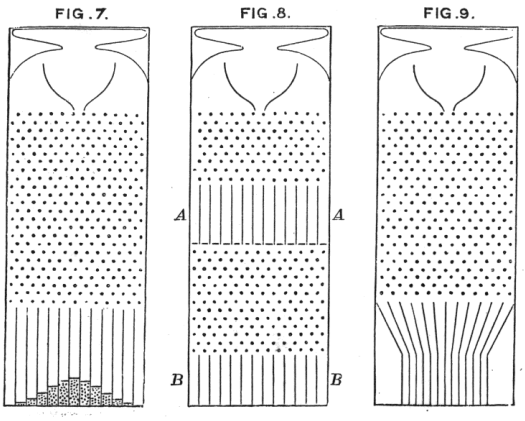


Figure 2: Galton Machine

in figure 2 above, where pellets falling from an entrance point formed a normal distribution. After decades, starting from the idea of regression towards the mean [Gal, 1877], statistical modeling framed regression analysis.

As a result, the fundamental problem of predicting a numeric dependent variable Y as the output of an unknown function f given some input independent variables $X = (X_1, X_2, \ldots, X_n)$ in the form:

$$Y = f(X) + \varepsilon \tag{1}$$

and this goes beyond the simple linear regression model where we can find the optimal model parameters by minimizing a Mean Square Error function, and obtain the optimal model described by parameter vector β with least squares

$$Y = X\beta + \varepsilon \tag{2}$$

$$\beta = \arg\min_{\beta} \sum_{\forall i} (y_i - x_i\beta)^2 \tag{3}$$

It goes beyond simple linear regression, univariate or multivariate, because of the function f. The specification of f, and the search of the optimal model weights in the parameter space β can reach very high levels of complexity with ensemble methods. Now we are going to take a look at the theoretical question that led to the development of ensemble methods.

2.2 Statistical and Computational Learning Theory

The question is the following: "can a set of weak learners combined create a single strong learner?" [Kearns and Valiant, 1989]. We know that the two things are equivalent thanks to the proof that a concept class is learnable (or strongly learnable) if there exists polynomial time algorithm that achieves low error with high confidence [Schapire, 1990]. A weak learner is a model consistently performing slightly better than random guessing. Ensemble models put weak models together to create strong learners: models with a consistently high performance, however usually there is a performance-interpretability trade-off. It is becoming a necessity to explain the decisions made by algorithms going beyond the typical "black-box" approach. According to [Blanchart, 2021] tree ensembles are the best for tracing back to the "cause" of the model making the output deicison. We are going to see that BART is an high performing model and it is more explainable than the other ensemble of trees thanks to its Bayesian "white-box" configuration.

2.3 Base models: Regression Trees

The base model for tree ensembles is a simple regression tree learner obtained through a consecutive binary splitting of the predictor space. This leads to a partitioning at each terminal node i with "homogeneous" distributions of $Y|x$ that can be modeled with a parameter $M = \{\mu_1, \mu_2, \ldots, \mu_b\}$. Usually, μ_i is associated with terminal node partitions based on mean and proportion hypothesis $H(A_i)$:

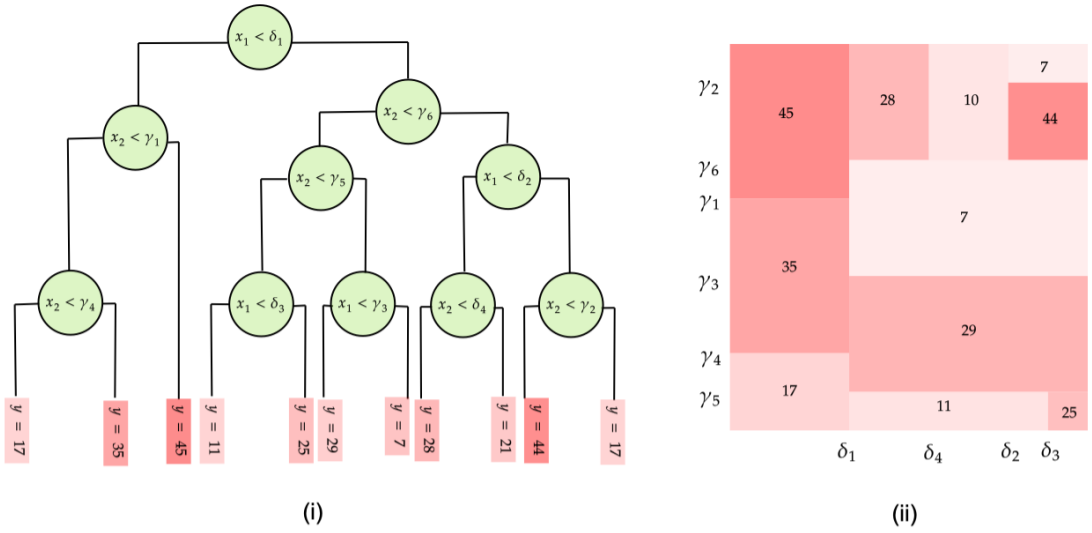
$$\begin{aligned} \text{Partition at depth } d : X &= \{\{A_1\}, \{A_2\}, \ldots, \{A_n\}\} \\ \Rightarrow Y &= \{\{H(A_1)\}, \{H(A_2)\}, \ldots, \{H(A_n)\}\} \end{aligned} \tag{4}$$

Regression tree models of this type can handle non-linearity and interaction effects.
An extensions of regression trees has been proposed by [Alexander and Grimshaw, 1996], through the addition of a affinity hypothesis:

$$H(A_i) = \mu_i x + \varepsilon, \quad \forall x \in A_i \tag{5}$$

because single regression might not fully exploit the tree partitioning at terminal nodes, being limited by the mean and proportion hypothesis. However, this

does not apply to tree ensembles, so we can stick to the old version regression trees. There are two representations for such models: the tree representation (i) is generalizable to more then two-dimensional predictor spaces. The rectangles representation (ii) is the 2D case of hyperrectangle partitioning of the predictor space, thus it is not a generalizable representation to more than 2D.



Here, given that the splits are binary, n is upper bounded by 2^d and it's increasing in the tree depth. The main problem of regression trees is their tendency to overfit. Tree depth can be set arbitrarily to reach even a null mean square error on the training data. Outside of the training sample, this will result in very bad generalization capabilities. In order to maintain some, the tree maximum depth must be set quite low, resulting in a model with a very high mean square error, but still a weak learner. At this point, we only have to combine many weak learners in a strong model and several ways have been explored in machine learning, in most cases focusing on performance, discarding more explainable statistical models on "non-mainstream" development tools (R), like Bayesian Additive Regression Trees.

2.4 Bootstrap Aggregation and Random Forest

A widely used tree-based ensemble model is random forest [Breiman, 2001], an extension of bootstrap aggregation, abbreviated with the acronym "bagging". Also

the bagging procedure was introduced by [Breiman, 1996] and it consisted in aggregating base predictors f_s by averaging their independent fitted values. Given a learning set X, Y, we consider every y as the realization of the random variable drawn from a probability distribution $P(Y)$ modeled by $randomForest(X)$. The base models are independent Regression Trees, since they are trained on independent subsets X_s, Y_s sampled with replacement from the data set X, Y, so that we can increase the number of bagged trees without limitations and with benefits.

$$randomForest(X) = \frac{1}{S}\sum_{s=1}^{S} f_s(X_s) \tag{6}$$

Each submodel $f_s(X_s)$ is weak since it is trained with undersampled data consisting only of the predictors of some data points and it has variance σ_s^2. On the contrary $randomForest$ being the average of randomly sampled independently fitted trees, by Central Limit Theorem, has variance decreasing in the number of trees S

$$\sigma_s^2 = \frac{1}{S}\sigma_s^2 \tag{7}$$

In addition to the data points subsampling performed by basic bootstrap aggregation methods, the random forest extension adds an ulterior method [Ho, 1998] to randomize the sample learning set X_s, Y_s of each submodel. In random forest we also sample the explanatory variables $X_{\alpha s}, \ldots, X_{\zeta s}$ from a subsample X_s.

2.5 Boosting and Gradient Boosting

The proof already mentioned in section 2.2 [Schapire, 1990] led to the development of boosted models. At each step t the partial fit of the t^{th} boosted tree is subtracted from the dependent variable, so that the base model F_{t+1} will be fitted to the residual of the previous submodel: $y_{t+1} = y_t - F_t(x)$. The submodels are weakened by a shrinkage (or learning rate) parameter, and the ensemble model gains strength through the fitting of the successive submodel on the error of the previous ones.
The extension of Boosting is Gradient Boosting, where at step t we fit the next model with a gradient descent correction: $F_{t+1}(x) = F_t(x) - \frac{\partial \mathcal{L}(F_t(x,y))}{\partial F_t(x)}$

2.6 CART model structure

Although similar in spirit to Boosting approaches, the Bayesian shrinkage for ensembles of Classification And Regression Trees (CART) models is obtained by wisely using priors on model parameters and by iteratively fitting new trees using a Bayesian adaptation of the Metropolis-Hastings as Gibbs Sampler.
A CART model [George and McCulloch, 1998] (CGM98 hereafter) has two main components: a binary tree T_j with a set of terminal nodes $b_i \in B$, and a parameter $\mu \in M = \{\mu_0, \mu_1, \dots\})$ associating a value μ_i to each terminal node b_i. The binary tree T_j, at each level subdivides the predictor space X with the most appropriate decision rule (henceforth splitting rule). For quantitative predictors the splitting rule creates two child nodes: $X_1 = \{x < s\}$, $X_2 = \{x > s\}$ based on a split-value s. We can also account for split-rules that cluster the predictors with a function $f(X)$: $X_i = \{f(x) > s\}$. For qualitative predictors, the splitting rule is based on categorical subsets: $X_1 = x \in C$, $X_2 = x \in C^C$. An additional assumption is that, conditionally on (T, M), the y values within a terminal node cluster are independently and identically distributed and y values across terminal nodes are independent $y_1, \dots, y_j | \mu_i \overset{\text{iid}}{\sim} N(\mu_k, \sigma_k^2)$, so the distribution of the data will be

$$p(y|X, M, T) = \prod_{i=1}^{b} f(Y_i|\mu_i) = \prod_{i=1}^{b} \prod_{j=1}^{J} f(y_{ij}|\mu_i) \tag{8}$$

where i is used for terminal nodes indexing $i = 1, 2, \dots, b$, and index j denotes the j^{th} observation of the dependent variable among the elements $j = 1, 2, \dots, J$ in the terminal node cluster i. For classification trees, where y belongs to one class C_k out of $C_1, \dots, C_K$, the parameter p_i associated with each terminal node i, is a multinomial probability distribution over the K classes: $p_i = P(y_{ij} \in C_k)$.

3 The BART model

Let (T_j, M_j) be a single regression tree, the base model building up BART as described in section 2.3. We define the base model in the same way as CGM10:

$$Y = g(x, T, M) + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2) \tag{9}$$

The conditional mean $E(Y|x)$ in (9) equals the terminal node parameter assigned by the tree terminal node b_i, namely μ_i. Thus, the additive model consisting of the sum of m trees can be explicitly expressed as

$$Y = \sum_{j=1}^{m} g(x, T_j, M_j) + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2) \tag{10}$$

with the normal error assumption tested in section 5.
Given a regression tree T_j with terminal node parameters M_j, every individual x will be evaluated into m treed regression models of the form $g(x, T_j, M_j)$ that will assign to x a value $\mu_{ij} \in M_j$ in math notation each tree will do: $g(x, T_j, M_j) = \mu_{ij}$. Overall, this results in $E(Y|x) = \sum_{j=1}^{m} \mu_{ij}$.
BART submodels will model interaction effects and main effects depending on the tree structure. If the tree T_j makes the splits on only one variable, the submodel $g(x, T_j, M_j)$ will represent a main effect, otherwise it will model interaction effects of varying order depending on the depth of the tree, usually pairwise as the tree rarely grows deeper than $d = 2$. As the number of trees m grows, the number of parameters will also grow quickly. Therefore we have to smooth down every submodel contribution. This will make the entire model flexible in the sense that different choices of regression trees $(T_1, M_1), \ldots, (T_m, M_m)$ can result in the same sum-of-trees ensemble model. Therefore, $\{g(x, T_1, M_1), \ldots, g(x, T_m, m)\}$ is an overcomplete basis, in the sense that even removing some base tree $g(x, T_j, M_j)$ we can obtain exactly the sum-of-trees model as above.

3.1 Regularization Prior: base model shrinkage

Since we want to preserve model flexibility and the excellent predictive capabilities of the sum-of-smoothed-trees model, the prior specification must be made in such a way that the fit is regularized, and each submodel influence is small, to avoid a single model to overwhelm the others. Optimal regression tree submodels are found by sampling from the posterior distribution over the parameter space Φ:

$$p(\Phi|data) \propto p(data|\Phi) \cdot p(\Phi) \tag{11}$$

To define the prior over $\Phi = T, M, \sigma$, CGM10 resort to hyperparameter calibration. Even though the prior is not really uninformed by the data due to some hyperparameters calibration, the only concern that led CGM10 to use observed variations in y and sacrifice Bayesian coherence to define the prior was guided by a Bayesian principle: specifying a prior that can be properly updated by the data.

3.1.1 Prior independence and symmetry

The prior specification problem can become complex if the priors are not independent, it is easier to focus on those satisfying it:

$$\begin{aligned} p((T_1, M_1), \ldots, (T_m, M_m), \sigma) &= \left[\prod_j p(T_j, M_j)\right] p(\sigma) \\ &= \left[\prod_j p(M_j|T_j) p(T_j)\right] p(\sigma) \end{aligned} \tag{12}$$

and

$$p(M_j|T_j)p(T_j) = \prod_i p(\mu_{ij}|T_j) \tag{13}$$

where $\mu_{ij} \in M_j$. With the prior independence assumption, we reduce the prior specification problem to the formulation of just $p(\sigma)$, $p(T_j)$ and $p(M_j)$. The forms proposed for Bayesian CART in CGM98 are thought for this problem and can be used for BART, but they slightly differ in parametrization choices.

3.1.2 The $p(\sigma^2)$ prior

Also for the variance we use a conjugate prior, still for computational complexity reasons. The two most common parametrizations for variance priors in Bayesian statistics are: inverse-gamma and inverse-χ^2, we want them to be non-informative. Following the approach of [Bernardo, 1979] we can think of non-informative priors as starting points to get posteriors that would have been obtained with the use of properly informed priors. Since we do not dispose of properly informed priors we can set:

$$\sigma^2|T_j \sim \text{IG}(\nu|2, \nu\lambda/2) \tag{14}$$

This parametrization yields posteriors that are very sensitive to ν according to [Gelman, 2006]. An alternative parametrization for the same distribution is the scaled inverse Chi-Square prior that was picked for BART specification in CGM10:

$$\sigma^2 \sim \frac{\nu\lambda}{\chi^2_\nu} \tag{15}$$

Even though inverse gamma priors are pointed out to be sensitive to parametrization (ν, λ), also for the prior (15) it is fundamental to set prior hyperparameters so that the prior is updatable by data, in line with Empirical Bayesian methods. This is why a data-informed calibration of the hyperparameter ν and scaling λ is the recommended tuning to avoid unproper dispersion or overconcentration of $p(\sigma^2)$. So, also in this case, the approach used is a mix of data-informed considerations and prior beliefs. Thus, CGM10 use a "rough overestimate" $\hat{\sigma}$ of σ based on data as prior. This random variable can be interpreted either as the sample variance of Y or as the residual variance of an ordinary least squares linear regression of $Y \sim f(X) + \varepsilon$. We are going to finish this prior specification by describing how to set ν and λ in section 3.1.

3.1.3 The T_j prior as a stochastic process

The initial uncertainty in the trees model space T_j is described by a prior distribution $p(T_j)$. This prior must contain information on (i) the probability of a node at depth d to be nonterminal, that we set low for $d > 3$ by specifying

$$\alpha(1+d)^{-\beta}, \qquad \alpha \in (0,1),\ \beta \in [0,\infty) \tag{16}$$

and the splitting rule assignment on two levels: (ii) the variable chosen for the split and (iii) the decision rule for the split.
However, it is very hard to write a closed form expression for $p(T_j)$, this can be tackled by using the tree generating stochastic process introduced by CGM98. The tree propensity to grow is controlled by the two functions determining whether a terminal node is split $p_{SPLIT}(b_i, T_j)$ and if so, which splitting rule to assign to the node $p_{RULE}(\rho|b_i, T_j)$ in the stochastic process structured as we illustrate in the following pseudo-code:

Tree-Generating Stochastic Process

```
1: procedure TREE( )
2:     Tree ← {root node}                          ▷ Tree is a set of node objects
3:     Terminal Nodes ← {root node}                ▷ Terminal Nodes ⊂ Tree
4:     for Node ∈ Terminal Nodes do
5:         with p_SPLIT(Node,Tree)
6:             C_Left, C_Right ← Children(Node)    ▷ Left and Right Children
7:             C_Left[ρ]← p_RULE(ρ|Node,Tree)      ▷ splitting rule ρ assignment
8:             C_Right[ρ] ← p_RULE(ρ|Node,Tree)    ▷ ρ for both child nodes
9:             Tree ← Tree ⋃ {C_Right,C_Left}
10:            Terminal Nodes ← Terminal Nodes \ Node
11:            Terminal Nodes ← Terminal Nodes ⋃ Children(Node)
12:    end for
13: end procedure
```

The internal structure of a Regression Tree object T_j is built by the stochastic process modifying two attributes: (i) the binary tree $(nodes, edges)$ and (ii) an assignment of splitting rules to the *nodes*. To understand how it is structured we have to analyze p_{RULE}, the distribution over the possible splitting rules. For every terminal node of the tree in the stochastic process, with probability p_{SPLIT} the terminal node will become internal by being split and generating two child terminal nodes.

3.1.4 Splitting rule assignments according to p_{RULE}

The assignment of splitting rules will build the tree and it will structure the clustering of the predictors. We have already anticipated the form of splitting rules for quantitative and qualitative variables in section 2.6. Now, we have to define the set of splitting rules ρ that we can assign to the tree terminal nodes. Note that these trees are only terminal in this step of the stochastic process, after splitting rule assignment they become internal. Terminal nodes will be assigned another parameter μ_{ij}.

In any finite data set, observations of variable X_i are $x_{i\alpha} \in \{x_{i1}, x_{i2}, \ldots, x_{in}\}$. Therefore, every possible splitting rule for categorical variables will be of the form:

$$x \in S, \quad \forall S \in \mathcal{P}(\{x_{i1}, x_{i2}, \ldots, x_{in}\}) \tag{17}$$

where $\mathcal{P}(X_i)$ is the power set of the realizations of X_i: $\{x_{i1}, x_{i2}, \ldots, x_{in}\}$. The first splitting rule will be chosen among 2^n possbilities, so large that even with a few hundreds of distinct split classes, the number of splitting rules would be greater than the estimated number of atoms in the universe, Bayesian Nonparametrics.
For numerical variables, the splitting rules building different trees are defined by the choice of a threshold value $s \in R$

$$x < s, \quad \forall s \in (X_i) \tag{18}$$

Although $s \in R$, in a finite data set s can only assume a finite number of values leading to distinct tree clustering models, so the model space will be finite. However, this "finiteness" is only due to the finiteness of the data set, and the BART model hypothetically works even in infinite spaces. Therefore it can be considered a Bayesian nonparametric model.
Having defined the set of possible splitting rules, we can proceed with the prior $p_{RULE}(\rho|\eta, T)$ specification. The trivial choice is the uniform specification: choose a splitting predictor uniformly among the independent variables and choose a splitting threshold value uniformly from the set of available split values for variable X_i. This naive specification is the most appropriate if we have no idea of what predictors are the most influential on the outcome, so it will be appropriate for my LOS prediction problem in section 5. If a decision rule has been used on higher levels of tree, it will not be available to avoid the tree to generate empty terminal nodes. The uniform specification robustness to monotone transformations is also a very appealing feature. As you might have already noticed, due to the sequential uniform sampling, these nested uniforms do not give the same probability to each category, unless the number of splits (distinct observations) of x is constant across every predictor.
The alternative procedure would be to split uniformly across all variables splits, the resulting problem is variables with less potential splits being ignored. We might want to use non-uniform p_{RULE} distributions for variable selection (see section 5.4) and put greater mass on variables that are thought to be more important.
From a Bayesian point of view, it is not easy to set a prior that is sensitive to data and that updates after the likelihood is computed. We have to carefully bal-

ance between: (i) too informative priors, that are very squeezed, thus too tight to be informed by the likelihood, and (ii) uninformative priors that are dispersed resulting in the posterior being even more stretched.

3.1.5 The $M|T_j$ prior

When defining the prior distribution $p(M|T_j)$ over the terminal node parameter Space $\mu_{ij} \in M$, it must be considered that the computational cost of posterior calculation can be heavily reduced by choosing a conjugate normal prior [Chipman and McCulloch, 2000]. Therefore, in order to marginalize out M and obtain the likelihood

$$p(Y|X, T_j) = \int p(Y|X, M_j, T_j)p(M_j|T)dM_j \tag{19}$$

we preserve conditional independence of parameters across terminal nodes with the model parameter random variable $M_j|T_j, \sigma$ is distributed as

$$\mu_{1j}, \ \dots \ , \mu_{bj}|\sigma, T_j \overset{\text{iid}}{\sim} N(\mu_\mu, \sigma_\mu^2) \tag{20}$$

To parametrize this Normal with μ_μ and σ_μ, we first take into account that the additive model conditional expectation is $E(Y|x) = \sum_{j=1}^{m} \mu_{ij}$ and since we assume that μ_{ij} are iid, the induced prior on $E(Y|x)$ is the scaled normal $N(m\mu_\mu, m\sigma_\mu^2)$. Consequently, the hyperparameters μ_μ, σ_μ are chosen so that the $E(Y|x)$ prior is centered on the region of plausible values of $E(Y|x)$ and with the right shrinkage. Even Though BART is quite robust to changes in the exact specification, in order to smooth down the submodels $g_1, \dots, g_m$, we min-max-scale Y into the interval $(-0.5, 0.5)$ and then we center it at 0 to make it very likely that $E(Y|x) \in (y_{min}, y_{max})$. This can be done simply by choosing the probability of μ_{ij} to fall in the right "ballpark" by setting $k = 2$ in

$$\sigma_\mu = 0.5/k\sqrt{m} \tag{21}$$

to get $p(\mu_{ij} \in (y_{min}, y_{max})) = 0.95$, but the ensemble model is robust to the exact specification. Despite this rescaling, predictors are robust to monotone transfor-

mation because they are only used for the decision rules.
Rescaling would have been necessary for a linear model, regression trees are non-linear $g(X, T, M)$, in the sense that the dependent variable is not modeled by a linear combination of the predictors. In addition to that, BART is an Additive ensemble, so the dependent variable is modeled by a linear combination of m shrinked non-linear functions of the features X. This sounds far from linearity at first, but [Hastie and Tibshirani, 2000] showed that their backfitting algorithm always converges for Generalized Additive Models [Hastie and Tibshirani, 1986]. In our case Bayesian backfitting is an adaptation of the backfitting algorithm, and it converges on the posterior of the Bayesian Additive model.

3.2 Other BART hyperparameters

Hyperparameters are required for hierarchical models. BART model parameters are specified with a Bayesian approach: setting a prior on them. So we consider the model parameters to be random variables distributed under some prior distribution that have to be informed by data. These priors are parametrized with hyperparameters. Hyperparameters must be set in order to make the model parameter prior updatable in a Bayesian fashion. We have already described the specification of some hyperparameters: the parameters of the terminal nodes prior $p(M) \sim N(\mu_\mu, \sigma_\mu^2)$, now we are going to discuss the parameters of the variance prior: ν and λ (see section 3.1.2).

3.2.1 ν and λ

The hyperparameters ν and λ are the parameters of the scaled inverse Chi-Square distribution for the variance prior described in section 3.1.2.

$$p(\sigma^2|\nu, \lambda(y)) = \frac{(\lambda/2)^2}{\Gamma(\nu/2)} \frac{\exp \frac{-\lambda(y)}{2\sigma^2}}{\sigma^{2(1+\nu/2)}} \tag{22}$$

In order to understand the choice of ν and λ, we have to exploit the fact that the scaled inverse-Chi-Square belongs to the exponential family. This allows us to

rewrite the prior in the [Diaconis and Ylvisaker, 1979] form

$$p(\sigma|\nu_0, \lambda_0) = \kappa(\nu_0, \lambda_0)c(\sigma)^{\nu} \tag{23}$$

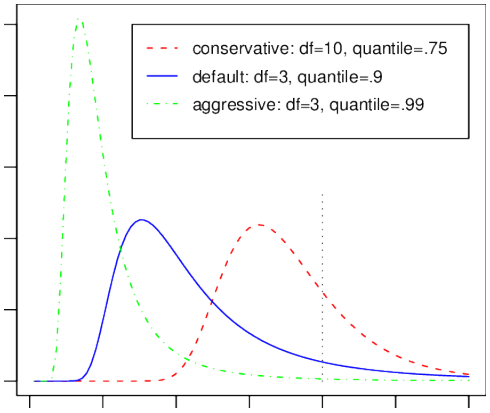
And for these conjugate priors, we can evaluate the posterior as in [Hoff, 2009]

$$\begin{aligned} p(\sigma^2|y_1, \ldots, y_n, (\nu_0, \lambda_0)) &\propto p(\sigma^2|\nu_0, \lambda_0)p(y_1, \ldots, y_n|\sigma^2, (\nu_0, \lambda_0)) \\ &= c(\sigma)^{\nu_0+\nu_n} \exp[\sigma(\nu_0\lambda_0 + \sum_{k=1}^{\nu_n} \lambda(y_k))] \\ &= p(\sigma|\nu_0 + \nu, \nu_0\lambda_0 + \nu_n \sum_{k=1}^{\nu_n} \lambda(y_k)) \end{aligned} \tag{24}$$

Where the conjugate prior informed by n observations suggests that it is possible to interpret ν_n as number of observations and ν_0 as "prior sample size", and ν_0 corresponds to the BART model hyperparameter ν: sample size. Therefore, it means our sample-variance prior gives information about σ^2 the same weight as if it was based upon 3 observations, making this a reasonably uninformative prior.
The choice of λ is data-informed. The parameter λ is the sufficient statistic of the $p(\sigma^2)$ prior model and we scale it using $\hat{\sigma}^2$, the unbiased estimate of σ^2 which is the residual variance of a full linear regression $Y \sim \beta X + \varepsilon$.
CGM10 refer to $\hat{\sigma}^2$ as a "rough data based overestimate" to express the expectation that the additive regression will have a lower standard error than the full linear regression model.
However, we still allow for $\hat{\sigma}^2 < \sigma^2$, with three settings of these hyperparameters with a different quantile q: (i) default with $q = 0.9$, (ii) aggressive with $q = 0.99$, (iii) conservative with $q = 0.75$, where $P(\hat{\sigma}^2 < \sigma) = q$, and the value of λ is implied by the chosen q [Chipman et al., 2002]

$$\lambda = \sigma^2 \Psi_{\nu}(1 - q)\nu \tag{25}$$

where Ψ_{ν} is the *c.d.f* of the scaled inverse-χ^2 with ν degrees of freedom.
In the results section we can see that performances with these three different settings of BART variance prior are pretty similar. In the graph below we can observe the different shape of the variance prior under these three settings.



3.2.2 The choice of hyperparameter m

The level of shrinkage needed for BART will depend on the number of trees that compose the model according to equation (21).
Also, in the spirit of hierarchical Bayesian models it might be reasonable to treat m as an unknown parameter and use a prior $p(m)$, or I could have selected m by cross-validation.
The only issue with these methods is computational complexity, CGM10 suggest to set $m = 200$ by default, since the 200-trees-BART yielded excellent predictive performance on several problems. However, there is an alternative BART implementation in R that runs faster than the one developed by CGM10, we are going to look at its specifics in section 5.

4 Bayesian backfitting

We are talking about Bayesian backfitting since the algorithm we are going to see is a stochastic generalization of the backfitting algorithm [Buja et al., 1989]. The ideal approach for finding the model with the greatest posterior probability in a very rigorous Bayesian setting would be to find the full posterior density and find the maximum.
Bayesian backfitting [Hastie and Tibshirani, 2000] does not evaluate the full posterior but it is the most appropriate stochastic approach for nonparametric models parameter space search. We have set non-informative priors to get a data-informed

posterior distribution (26) and infer the optimal model from the posterior

$$p((T_1, M_1), \ldots, (T_m, M_m), \sigma | y) \tag{26}$$

In practice, the computational cost of full posterior density calculation makes it unfeasible in terms of time complexity, except for trivially small problems, where BART ensemble method does not particularly exploit its power.
As a matter of fact, despite the restrictions made in section 3.1.3 regarding the model space finiteness, finding the full density would result in a combinatorial explosion in terms of computational complexity, due to the hierarchical specification of the parameter space in section 3. Therefore in most cases it will be nearly impossible to find the absolute optimum a posteriori in the model space.
We have to proceed with a stochastic version of the backfitting algorithm for Bayesian additive models. The following procedure to estimate the additive model $f = \sum_{j=1}^{m} g(X, T, M)$ iterates over j - for notational convenience $(T_{(j)}, M_{(j)})$ is going to be the set of all m tree models except j - in the following way:

BART Backfitting -
Adaptation from [Hastie and Tibshirani, 2000] and [Chipman et al., 2010]

input: Priors, Hyperparameters ▷ defined in section 3
T, M, σ are initialized ▷ single node tree models
for $t = 0, 1, \ldots, r$ **do** ▷ r = number of restarts
    for uniform sampling $j \in \{0, 1, \ldots, m\}$ **do**:
        Calculate the partial residual only excluding Tree j: $R_{(j)}$
        $R_{(j)} \leftarrow y - \sum_{k \neq j} g(x, T_k, M_k)$
        if convergence:
            return output model
        else
            update $\sigma | T, M, y \sim$ IG ▷ conjugate IG prior
            update $T_j | R_{(j)}, \sigma$ with MH algorithm ▷ generate next tree
            update $M_j | T_j, \sigma \overset{\text{iid}}{\sim} N(\mu_t^j, \sigma))$ ▷ conjugate prior
            continue iterating
        Until the joint distribution of $(f_0^t, f_1, t, \ldots, f_p^t)$ converges to stationary

Where at each step t we restart the backfitting and we re-initialize the single rooted-tree models T, the parameters M associated with their terminal nodes

and $\sigma^2|T_j \overset{\text{iid}}{\sim} \text{IG}(\nu|2, \nu\lambda/2)$. Then, for each tree j: we exclude the regression tree $g(x, T_j, M_j)$ from the m base models in the ensemble and we calculate the sum-of-trees model made by the other $m-1$ base learners. We get the the partial residual. This random variable is used for Bayesian backfitting and for generating the next tree in the sequence: $T_j|R_{(j)}$ with Metropolis-Hastings.
When backfitting the T_j tree to $R_{(j)}$, the dependent variable will fit this tree on the part of y unexplained by the other trees, somewhat similar to boosting. In addition to that, the multiple restarts given by iterations over $t = 1, 2, \ldots, r$ give this model a remarkably similar result even in difficult problems.
However, as shown in the BART-backfitting pseudo-code, all the tree model parameter updates are conditional on the Tree T_j that is generated with Metropolis-Hastings, so now we are going to see how this procedure works.

4.1 Metropolis-Hastings algorithm

In the Metropolis Hastings algorithm we set a starting tree T^0 and then at each step t we simulate the transition from T^t to T^{t+1} with the following steps:

Metropolis-Hastings Algorithm Steps

1: **Sample** T^* from $q(T^t, T^*)$ ▷ q specified in the next section
2:

$$\alpha(T^t, T^*) \leftarrow min\left[\frac{q(T^t, T^*)}{q(T^*, T^t)}\frac{p(Y|X, T^*) \cdot p(T^t)}{p(Y|X, T^t) \cdot p(T^t)}, 1\right] \tag{27}$$

3: **with** probability $\alpha(T^1, T^*)$ ▷ $\alpha(T^t, T^*)$ is acceptance probability
4: $T^{t+1} \leftarrow T^*$
5: **otherwise**
6: $T^{t+1} \leftarrow T^t$ =0

Metropolis-Hastings would not be Bayesian per se, we make it Bayesian by setting the acceptance probability $\alpha(T^t, T^*)$ dependent on the ratio of posterior probability of the proposed tree T^* and the current tree T^t.
If [Tierney, 1994] weak conditions are satisfied, the sequence output by this algorithm will be a Markov chain with limiting distribution $p(T|Y, X)$. Note that the norming constant for $p(T|Y, X)$ is not needed to compute.

To understand the steps of the algorithmic search over the model space, we need to look at the possible the transition proposals for T^{t+1}.

4.1.1 Tree proposal moves

We consider proposals $q(T, T^*)$ which generate T^* from T by randomly choosing among four steps:

- GROW: Increase the depth of the tree by randomly picking a terminal node and generate child nodes by spliting it into two new ones though a random splitting rule assignment, sampled from p_{RULE} used for the prior (see section 3.1.2).
- PRUNE: Decrease the depth of the tree by randomly selecting a parent of two terminal nodes and by making it terminal, by simply collapsing its child nodes.
- CHANGE: Random update of splitting rules by randomly picking a non-terminal node, and reassign it a splitting rule sampled from the p_{RULE} from the prior.
- SWAP: Randomly pick a parent-child pair which are both internal nodes, swap their splitting rules unless the other child has the identical rule. In that case, swap the splitting rule of the parent with that of both children.

The proposal above has some appealing features. To begin with, it adds some randomization to prevent the the posterior search from getting stuck in some regions of the model space. Then, it yields a reversible Markov chain because every step from T^t to T^{t+1} must have a counterpart that can move from T^{t+1} to T. Indeed, the GROW and PRUNE steps are counterparts of one another, and the CHANGE and SWAP steps are their own counterparts.
As a result, also this will contribute to model flexibilty and to overcompleteness of the basis $\{g(x, T_1, M_1), \ldots, g(x, T_m, m)\}$, that was already overcomplete, as mentioned in section 3, since the base learners are shrinked.

4.2 Metropolis-Hastings Markov Chain as a Gibbs sampler

The Metropolis-Hastings algorithm is used to draw observations from the posterior. Whereas the model full posterior is hard to sample from directly, the conditional distributions of each model component given the others are relatively easy to calculate step by step. Consequently, at a general level the backfitting MCMC algorithm is a Gibbs sampler.

The result is that the Markov Chain sequence of samples generated by Metropolis-Hastings

$$T^0, T^1, T^2, \dots \tag{28}$$

converges to stationary into the posterior distribution and we can use it to estimate MAP. Since this posterior exploration is not exhaustive, we do not find the posterior density to find the maximum.

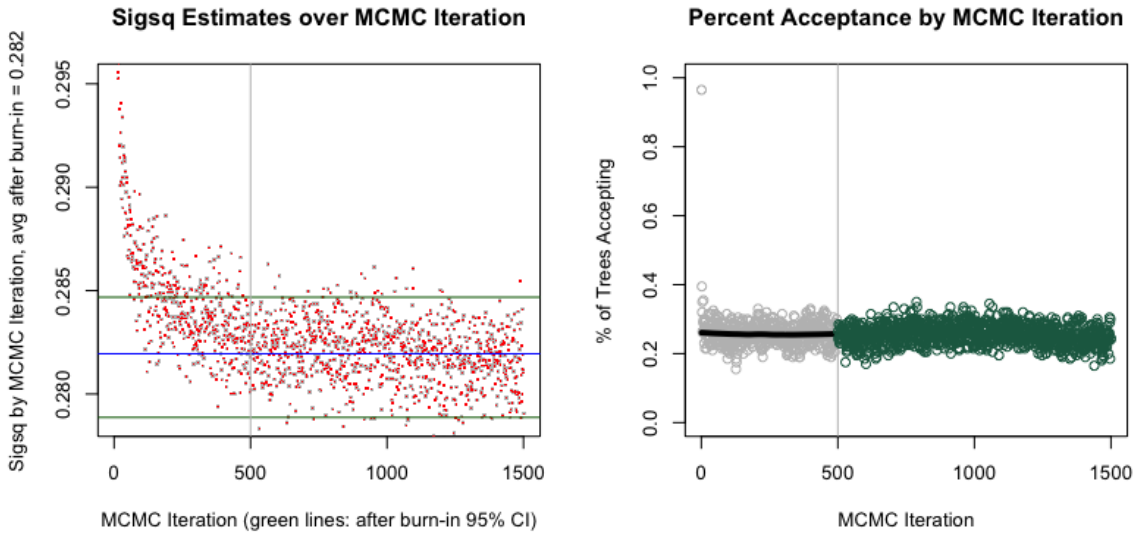
However, Bayesian CART model search search is still way more comprehensive than greedy search and more "directed" than grid-search. It is comprehensive for three reasons: (i) it moves past local maxima with a probability greater than 0, greedy search would stop at such points, therefore the model subspace sampled with the MCMC procedure is way larger.

(ii) It generates many variants of tree proposals T^* since it does not just grow or prune the tree but also modifies it by swapping or changing splitting rule assignment to the nodes, and finally (iii) with the multiple restarts of the tree growing process it provides an additional level of randomization that helps finding a more diverse set of models.

It is directed because the Bayesian setting shapes an acceptance rule $\alpha(T^*, T^t)$ that makes the search gravitate around the posterior models that better approximate Y.

In machine learning usually we do this manually, i.e. with grid-search hyperparameter tuning we would have to search the model space by training it multiple times, and then proceed by selecting manually selecting a good region in the hyperparameter posterior. We have to do it by hand because the search is not recursively self-informed in a Bayesian fashion.

In the figures above we can see that sampled trees generated by Bayesian model search converge.



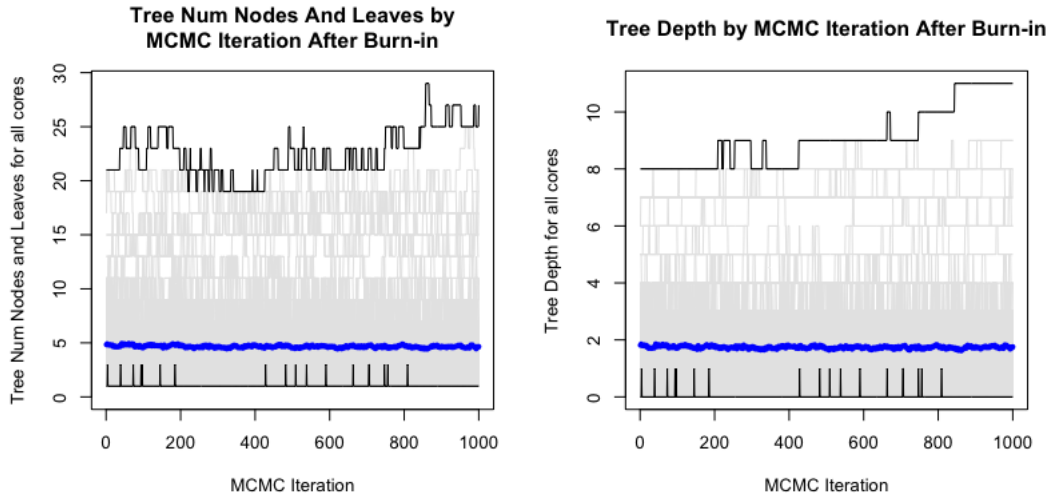
After the *burn-in* period acceptance rate is stable around 28% and the error variance is consistently bounded in the interval $0.279 < \sigma^2 < 0.285$. This indicates that the model has converged.

Also [Lutsko and Kuijpers, 1994] considered an MCMC-like search with simulated annealing, another type of search we can use for NP-Complete problems such as the Traveling Salesman Problem. In computer science NP-complete problems are problems that would require an infinite amount of computational power to be fully solved by a deterministic turing machine.

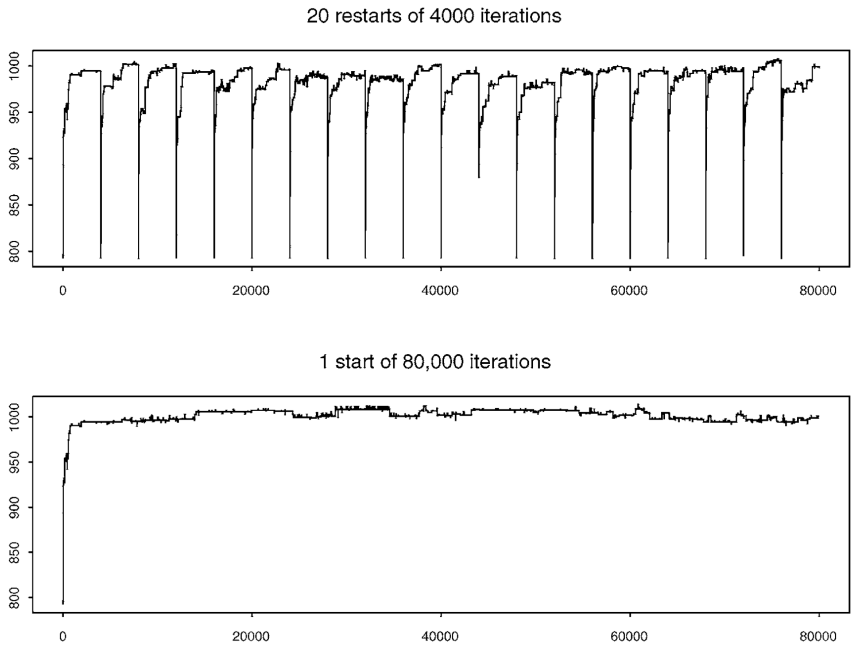
This is why their solution is given in polynomial time only by non-deterministic procedures.

4.3 Model space posterior

As the graph above shows, Bayesian Metropolis Hastings makes the search stabilize in a good region of regression tree weak models for BART, with consistent low mean square error. The region of the model space where $p(T|Y, X)$ is high, and consequently where the search stabilizes is quite evident in the figure below. Bayesian backfitting will gravitate with more and more local displacements for the rest of the iterations is a region of small trees.



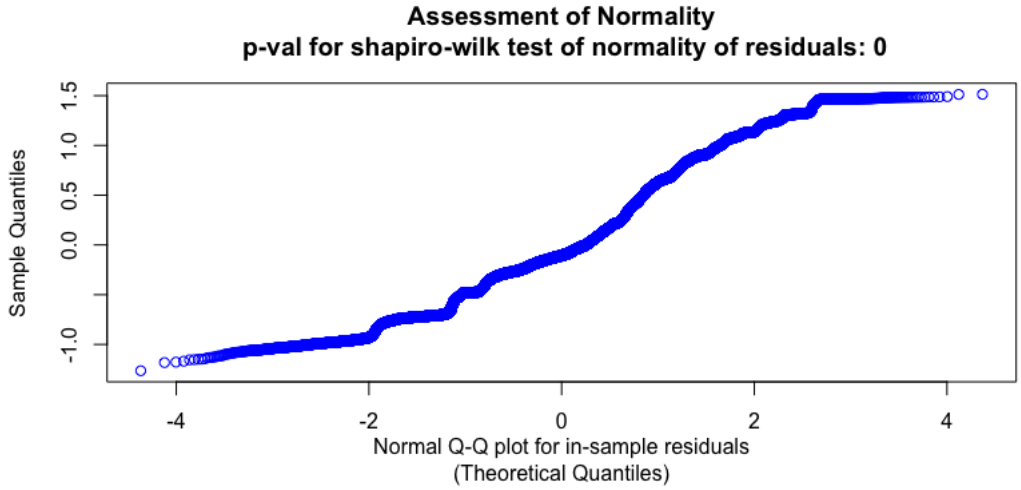
To improve the search, we can look and try to characterize the model displacements in this stable period, referred to as *after burn-in period*. We can characterize it by assuming that the posterior is a sharply peaked multimodal. This inter-modal moves will occur from time to time, but CGM98 agree that it is desirable to avoid wasting time waiting for mode-to-mode moves. That is why they recommend to restart the search multiple times, saving the most promising trees from each run will also be useful. We can see the model search with and without restarts below:



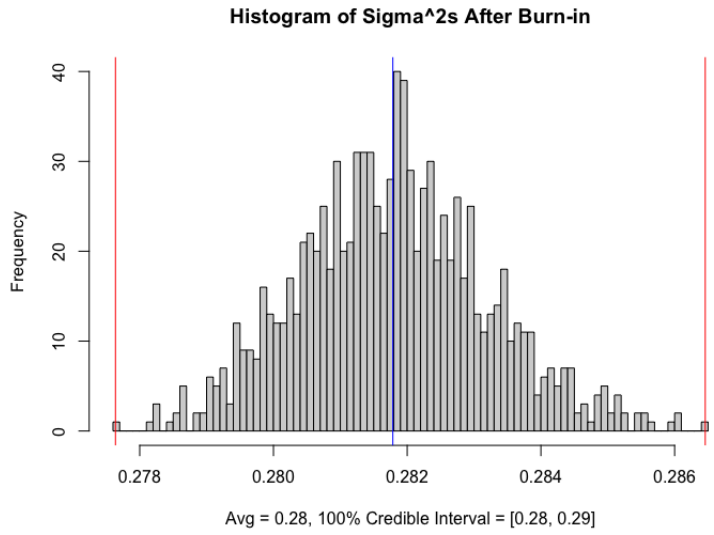
The decision of how many restarts and run per start will depend on sample size, signal to noise ratio and complexity of the data. In order to quantify how good is a tree, a very sensible is calculating the posterior $p(T_i|Y, X)$ for each posterior tree T_i and then select the sample MAP tree.

5 BART to predict Length of Stay

Length Of Stay (LOS) is the number of days from the initial admit date to the date the patient is discharged. Obviously, LOS varies depending on disease conditions, and LOS prediction can significantly enhance the quality of care and help organize the workload in the Hospital. Length of stay is similarly distributed to a log-normal random variable, so the variable we are predicting is log-LOS. When introducing BART as in equation (9) we assume that our data will be normally distributed, and so the error. Now we are going to test this assumption and draw a quantile-quantile plot:



The fact that points in the extremities are slightly curved off indicates that there are some extreme values and the distribution is not exactly normal, but it passes the Shapiro-Wilk test for residual normality with a very low p-value.



Also the histogram plot of the error variance of the estimates looks like a Normal distribution but with fat tails.

5.1 BART model in R

The very first development of the BART model in R [Chipman and McCulloch, 2016] is very sound from a theoretical point of view due to the authors, but it presented some computational complexity problems explained by the them in "Parallel Bayesian Additive Regression Trees" [Pratola et al., 2013]. BART is not fast enough to be interactive when the number of observations is greater than 1000, and according to them it is even unlikely to run with over 50000 observations.
In the Parallelized BART model paper some improvements for the BayesTree package were suggested and partially implemented in later version of the 'BayesTree' package leading to better runtime. On my database there were 80000 observations and BayesTree ran in 33 minutes, with the performance reported in section 5.5. The improvements suggested were: firstly, a simplification of the C++ representation of the regression tree model and a computational improvement of the Metropolis Hastings step presented in [Chipman and McCulloch, 2016]. Finally, the most important step to make BART model scalable was splitting up the data and allocating each portion to a different core, the workload can be parallelized and then recombined across portions to handle larger datasets in a reasonable time.

Even more simply, the lack of a "predict" function is a user-unfriendly drawback, since to make predictions on new unseen data the model has to be re-fitted, which takes a long time even with with the implementation of some of the improvements mentioned above. The newest package 'dbart' [Dorie, 2022] solved this and it still uses a C++ engine. A python library is also available: bartpy.

5.2 bartMachine

An earlier package is 'bartMachine' [Kapelner and Bleich, 2016], implemented in Java and integrated into R. From a run time perspective, this algorithm is much faster, thanks to parallelization on different cores it ran in 18 minutes on my machine 4 cores. It is also more user-friendly oriented, with the addition of new features such as a prediction function and interesting visualization tools that are also parallelized. This packages has many advantages and some drawbacks compared to the BayesTree package. It contains a very large set of tools for visualizations that can be very helpful to explore and understand the model.

5.2.1 Limitations

The default probability distribution over the type of move to be chosen by $p_{(\text{RULE})}$

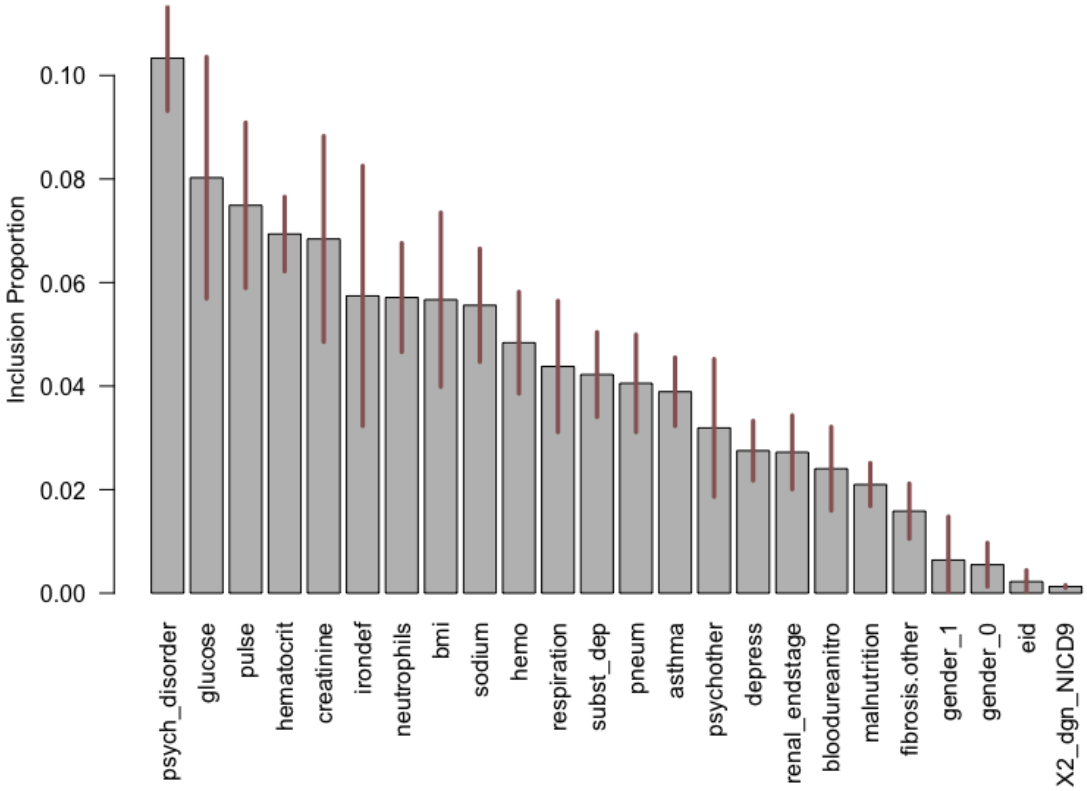
BayesTree		bartMachine	
p(GROW)	0.25	p(GROW)	0.278
p(PRUNE)	0.25	p(PRUNE)	0.278
p(CHANGE)	0.40	p(CHANGE)	0.444
p(SWAP)	0.10	p(SWAP)	$\rightarrow$ 0

The Metropolis Hastings used for iterating over of the next tree model does not support the 'SWAP' rule proposal that might have been useful for the last phase of the after *burn-in* model search, where it gets harder for the search to provide new information (see section 4.1.1). The reason precluding this "SWAP" randomization move from being implemented in bartMachine is that pointers rearrangement in Java is more complicated compared to C++. However, theoretically speaking the tree generating process described in section 3.1.1 is a reversible Markov Chain. Thus it can occur that two splitting rules are swapped, in very specific

combination instances of the other moves such as PRUNE ($p = 0.278$) followed by any other move (GROW or CHANGE) and the random sampling from p_{RULE} extracting exactly the pruned splitting rule. Consequently, it is pretty obvious that $p(\text{SWAP}) \to 0$, especially for large learning sets.

5.3 BART for variable selection

BART interpretability at the single variable level. we can see the histogram of the variables inclusion proportion: the posterior probability of the splitting rule to sample variable X_i as decision variable.

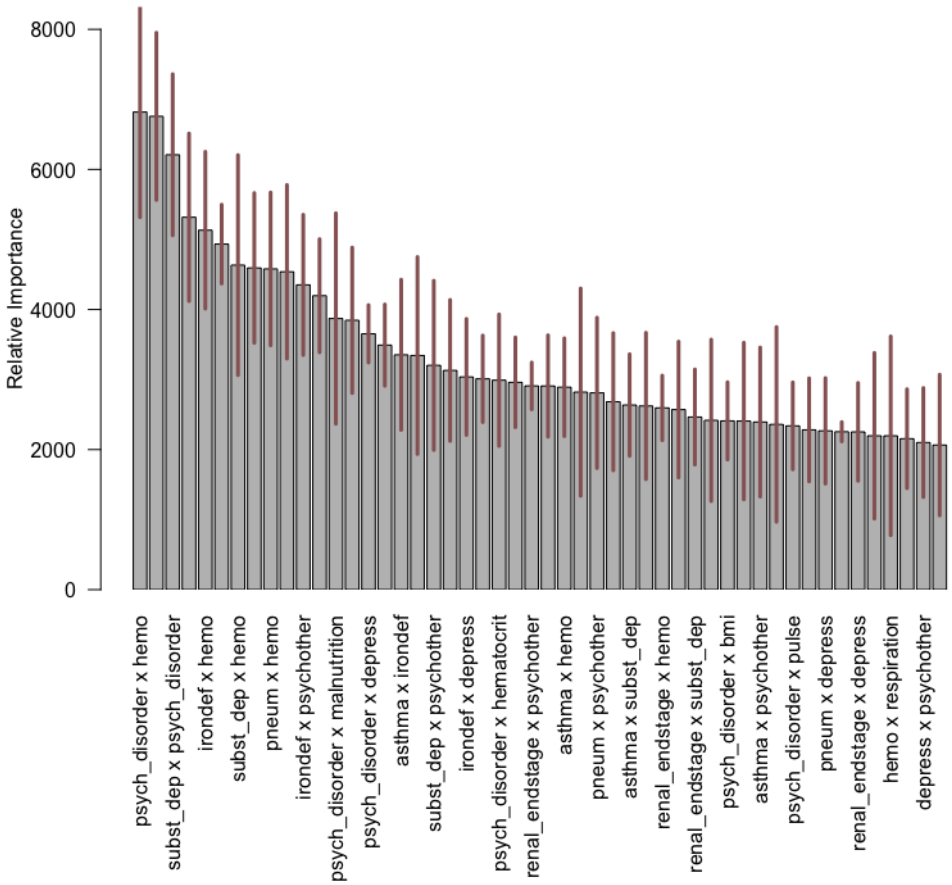


5.3.1 Informed prior information on covariates

In the bartMachine R package, we can incorporate informed prior in the splitting rules, by modifying the T_j prior in p_{RULE} as suggested by [Bleich et al., 2014]. This modification consists in doubling the split probability on influential variables.

5.4 Interaction effects

This can be extented to see to pairwise investigation to learn about the interactions fit in the model. For instance, the second most frequent pair of splitting variables affecting (LOS) is psychological disorder and substance dependence

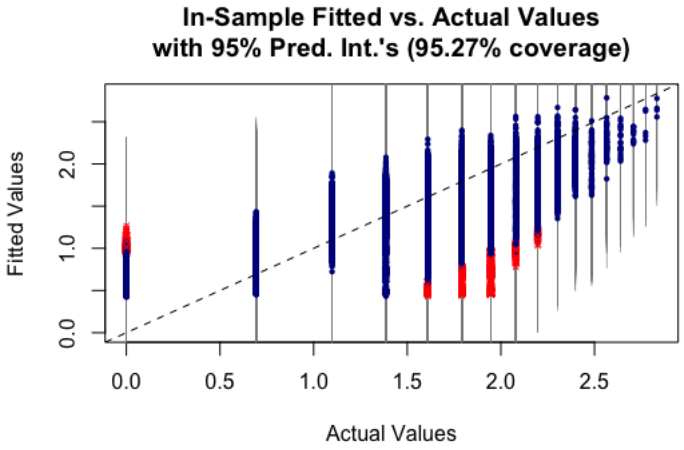


5.5 BART Results

As anticipated in section [Chipman et al., 2010], there are three configurations for the hyperparameters $(\nu, q) = (10, 0.75), (3, 0.90), (3, 0.99)$: conservative, default and aggressive. We report their performance in the table below.

BART	(ν, λ)	MSE
Default	$(3, 0.9)$	0.2851
Aggressive	$(3, 0.99)$	0.2846
Conservative	$(10, 0.75)$	0.2849

Different configurations of these values impact the prior mode by moving it towards smaller values of σ as q increases. Thus we can intepret q as an ulterior shrinkage parameter.
With these settings BART actual LOS values were in the 95% Confidence Interval of the fitted model with 95.27% probability.



5.6 Models compared: Ensembles and Linear Regression

BART performance is also very good compared to the other ensemble models we have presented in section 2.

Method	MSE
OLS Linear Regression	0.3805
Regression Tree	0.3641
Random Forest	0.2939
Gradient Boosted Model	0.2905
Extreme Gradient Boosting	0.2866
BART (Aggressive)	0.2846

The very good interpretability of this model is due to the strong Bayesian framework in which the model is specified. Usually in machine learning, model specification is dictated by performance oriented considerations, even without the theoretical basis being specified in detail.

6 Discussion

Bayesian Additive Regression Trees ensemble is not a machine learning model, it is a statistical model. Very basically, model parameters are considered and treated as random variables in a Bayesian Framework. This yields a very good interpretability-performance trade-off, since the Bayesian updates inform the posterior and help finding good models and most importantly, we can interpret model "parameters" as statistical objects and not just as computational objects, so they are different entities. And they have always been used to solve real world problems [Bleich et al., 2014] but potentially they could guide us in explaining computational world problems [Guo et al., 2018]. This explainability issue is getting legally important lately ([GDP, 2018] section 71) since machine learning algorithms make decisions that potentially affect people lives nowadays: legally or financially (credit scoring in the USA) and it is not always desirable to put them in the hands of black-box models.

References

[Gal, 1877] (1877). Typical laws of heredity1. *Nature*, 15(388):492–495.

[GDP, 2018] (2018). 2018 reform of eu data protection rules.

[Alexander and Grimshaw, 1996] Alexander, W. P. and Grimshaw, S. D. (1996). Treed regression. *Journal of Computational and Graphical Statistics*, 5(2):156–175.

[Bernardo, 1979] Bernardo, J. (1979). Reference posterior distributions for bayesian inference. *Journal of the Royal Statistical Society. Series B*, 41.

[Blanchart, 2021] Blanchart, P. (2021). An exact counterfactual-example-based approach to tree-ensemble models interpretability. *CoRR*, abs/2105.14820.

[Bleich et al., 2014] Bleich, J., Kapelner, A., George, E. I., and Jensen, S. T. (2014). Variable selection for BART: An application to gene regulation. *The Annals of Applied Statistics*, 8(3):1750 – 1781.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

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

[Buja et al., 1989] Buja, A., Hastie, T., and Tibshirani, R. (1989). Linear Smoothers and Additive Models. *The Annals of Statistics*, 17(2):453 – 510.

[Chipman and McCulloch, 2016] Chipman, H. and McCulloch, R. (2016). *BayesTree: Bayesian Additive Regression Trees*. R package version 0.3-1.4.

[Chipman and McCulloch, 2000] Chipman, H. and McCulloch, R. E. (2000). Hierarchical priors for bayesian cart shrinkage. *Statistics and Computing*, 10(1):17–24.

[Chipman et al., 2002] Chipman, H. A., George, E. I., and McCulloch, R. E. (2002). Bayesian treed models. *Machine Learning*, 48(1):299–320.

[Chipman et al., 2010] Chipman, H. A., George, E. I., and McCulloch, R. E. (2010). BART: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1).

[Diaconis and Ylvisaker, 1979] Diaconis and Ylvisaker (1979). *The Annals of Statistics*, pages 269–281.

[Dorie, 2022] Dorie, V. (2022). dbarts: Discrete bayesian additive regression trees sampler. R package version 0.9-22.

[Gelman, 2006] Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, 1(3):515 – 534.

[George and McCulloch, 1998] George, E. I. and McCulloch, R. E. (1998). Approaches for bayesian variable selection. *Statistica Sinica*, 7(2):339–373.

[Guo et al., 2018] Guo, W., Huang, S., Tao, Y., Xing, X., and Lin, L. (2018). Explaining deep learning models - a bayesian non-parametric approach.

[Hastie and Tibshirani, 1986] Hastie, T. and Tibshirani, R. (1986). Generalized Additive Models. *Statistical Science*, 1(3):297 – 310.

[Hastie and Tibshirani, 2000] Hastie, T. and Tibshirani, R. (2000). Bayesian backfitting. *Statistical Science*, 15(3):196–213.

[Ho, 1998] Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844.

[Hoff, 2009] Hoff, P. D. (2009). *A First Course in Bayesian Statistical Methods*. Springer Publishing Company, Incorporated, 1st edition.

[Kapelner and Bleich, 2016] Kapelner, A. and Bleich, J. (2016). bartMachine: Machine learning with Bayesian additive regression trees. *Journal of Statistical Software*, 70(4):1–40.

[Kearns and Valiant, 1989] Kearns, M. and Valiant, L. G. (1989). Crytographic limitations on learning boolean formulae and finite automata. In *STOC '89*.

[Lutsko and Kuijpers, 1994] Lutsko, J. F. and Kuijpers, B. (1994). Treed regression. *Selecting Models from Data*, pages 453–462.

[Merriman, 1877] Merriman, M. (1877). *A List of Writings Relating to the Method of Least Squares: With Historical and Critical Notes*. Transactions of the Connecticut Academy of Arts and Sciences. Academy.

[Pearson, 1930] Pearson, K. (1930). *The Life, Letters and Labours of Francis Galton*, volume 3 of *Cambridge Library Collection - Darwin, Evolution and Genetics*. Cambridge University Press.

[Pratola et al., 2013] Pratola, M. T., Chipman, H. A., Gattiker, J. R., Higdon, D. M., McCulloch, R., and Rust, W. N. (2013). Parallel bayesian additive regression trees.

[Schapire, 1990] Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

[Tierney, 1994] Tierney, L. (1994). Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728.